

Object Storage Service

Java SDK Developer Guide (Ally Region)

Issue	01
Date	2026-01-15



Copyright © Huawei Cloud Computing Technologies Co., Ltd. 2026. All rights reserved.

No part of this document may be reproduced or transmitted in any form or by any means without prior written consent of Huawei Cloud Computing Technologies Co., Ltd.

Trademarks and Permissions



HUAWEI and other Huawei trademarks are the property of Huawei Technologies Co., Ltd.

All other trademarks and trade names mentioned in this document are the property of their respective holders.

Notice

The purchased products, services and features are stipulated by the contract made between Huawei Cloud and the customer. All or part of the products, services and features described in this document may not be within the purchase scope or the usage scope. Unless otherwise specified in the contract, all statements, information, and recommendations in this document are provided "AS IS" without warranties, guarantees or representations of any kind, either express or implied.

The information in this document is subject to change without notice. Every effort has been made in the preparation of this document to ensure accuracy of the contents, but all statements, information, and recommendations in this document do not constitute a warranty of any kind, express or implied.

Huawei Cloud Computing Technologies Co., Ltd.

Address: Huawei Cloud Data Center Jiaoxinggong Road
Qianzhong Avenue
Gui'an New District
Gui Zhou 550029
People's Republic of China

Website: <https://www.huaweicloud.com/intl/en-us/>

Contents

1 Before You Start (SDK for Java)	1
2 API Overview (SDK for Java)	2
3 Preparations (SDK for Java)	16
4 SDK Download and Installation (SDK for Java)	18
5 Getting Started (SDK for Java)	19
6 Initialization (SDK for Java)	26
6.1 Overview (SDK for Java)	26
6.2 Creating and Configuring an OBS Client (SDK for Java)	28
6.3 Configuring SDK Logging (SDK for Java)	40
7 Bucket Management (SDK for Java)	42
7.1 Creating a Bucket (SDK for Java)	42
7.2 Obtaining a Bucket List (SDK for Java)	57
7.3 Deleting a Bucket (SDK for Java)	68
7.4 Checking Whether a Bucket Exists (SDK for Java)	71
7.5 Obtaining Bucket Metadata (SDK for Java)	74
7.6 Setting a Bucket ACL (SDK for Java)	80
7.7 Obtaining a Bucket ACL (SDK for Java)	94
7.8 Configuring a Bucket Policy (SDK for Java)	101
7.9 Obtaining the Policy of a Bucket (SDK for Java)	105
7.10 Deleting a Bucket Policy (SDK for Java)	108
7.11 Obtaining the Region of a Bucket (SDK for Java)	111
7.12 Obtaining Storage Information of a Bucket (SDK for Java)	114
7.13 Configuring a Storage Quota (SDK for Java)	117
7.14 Obtaining a Bucket Storage Quota (SDK for Java)	121
7.15 Configuring a Storage Class for a Bucket (SDK for Java)	124
7.16 Obtaining the Storage Class of a Bucket (SDK for Java)	128
7.17 Bucket Inventory (SDK for Java)	131
7.17.1 Configuring an Inventory Rule (SDK for Java)	131
7.17.2 Obtaining an Inventory Rule (SDK for Java)	139
7.17.3 Listing Inventory Rules (SDK for Java)	147
7.17.4 Deleting an Inventory Rule (SDK for Java)	155

7.18 Configuring a Custom Domain Name for a Bucket (SDK for Java).....	158
7.19 Obtaining the Custom Domain Names of a Bucket (SDK for Java).....	164
7.20 Deleting the Custom Domain Name of a Bucket (SDK for Java).....	168
8 Object Upload (SDK for Java).....	172
8.1 Overview (SDK for Java).....	172
8.2 Uploading an Object - Streaming (SDK for Java).....	172
8.3 Uploading an Object - File-Based (SDK for Java).....	195
8.4 Obtaining the Upload Progress (SDK for Java).....	216
8.5 Creating a Folder (SDK for Java).....	220
8.6 Configuring Object Metadata (SDK for Java).....	242
8.7 Multipart Upload (SDK for Java).....	267
8.7.1 Multipart Upload Overview (SDK for Java).....	267
8.7.2 Initiating a Multipart Upload (SDK for Java).....	267
8.7.3 Uploading a Part (SDK for Java).....	287
8.7.4 Assembling Parts (SDK for Java).....	299
8.7.5 Code Example of a Multipart Upload (SDK for Java).....	307
8.7.6 Aborting a Multipart Upload (SDK for Java).....	309
8.7.7 Listing Uploaded Parts (SDK for Java).....	313
8.7.8 Listing Multipart Uploads (SDK for Java).....	324
8.8 Configuring Lifecycle Rules (SDK for Java).....	339
8.9 Uploading an Object - Append (SDK for Java).....	360
8.10 Uploading an Object - Resumable (SDK for Java).....	382
8.11 Uploading an Object - Browser-Based (SDK for Java).....	404
8.12 Using CRC64 to Verify File Consistency (SDK for Java).....	411
8.12.1 Using CRC64 to Verify File Consistency During Uploads.....	411
8.12.2 Using CRC64 to Verify File Consistency During Downloads.....	416
8.12.3 Obtaining CRC64 of an Object.....	417
9 Object Download (SDK for Java).....	423
9.1 Overview (SDK for Java).....	423
9.2 Downloading an Object - Streaming (SDK for Java).....	424
9.3 Downloading an Object - Range-Based (SDK for Java).....	440
9.4 Obtaining the Download Progress (SDK for Java).....	456
9.5 Downloading an Object - Conditional (SDK for Java).....	459
9.6 Rewriting Response Headers (SDK for Java).....	475
9.7 Obtaining User-defined Metadata (SDK for Java).....	490
9.8 Restoring a Cold Object (SDK for Java).....	505
9.9 Downloading an Object - Resumable (SDK for Java).....	510
10 Object Management (SDK for Java).....	524
10.1 Configuring Object Metadata (SDK for Java).....	524
10.2 Obtaining Object Metadata (SDK for Java).....	539
10.3 Configuring an Object ACL (SDK for Java).....	550

10.4 Obtaining an Object ACL (SDK for Java).....	562
10.5 Listing Objects (SDK for Java).....	571
10.6 Deleting an Object (SDK for Java).....	596
10.7 Batch Deleting Objects (SDK for Java).....	601
10.8 Copying an Object (SDK for Java).....	612
10.9 Copying an Object - Multipart (SDK for Java).....	643
10.10 Checking Whether an Object Exists (SDK for Java).....	653
10.11 Setting an Object Expiration Time (SDK for Java).....	658
11 Temporarily Authorized Access (SDK for Java).....	661
11.1 Accessing OBS Using a Signed URL (SDK for Java).....	661
12 Versioning (SDK for Java).....	688
12.1 Overview (SDK for Java).....	688
12.2 Configuring Versioning for a Bucket (SDK for Java).....	688
12.3 Obtaining the Versioning Status of a Bucket (SDK for Java).....	692
12.4 Obtaining an Object Version (SDK for Java).....	696
12.5 Copying an Object Version (SDK for Java).....	712
12.6 Restoring a Specific Cold Object Version (SDK for Java).....	732
12.7 Listing Object Versions (SDK for Java).....	738
12.8 Setting an ACL for an Object Version (SDK for Java).....	768
12.9 Obtaining the ACL of an Object Version (SDK for Java).....	778
12.10 Deleting an Object Version (SDK for Java).....	786
12.11 Batch Deleting Object Versions (SDK for Java).....	790
13 Lifecycle Management (SDK for Java).....	799
13.1 Overview (SDK for Java).....	799
13.2 Setting Lifecycle Rules (SDK for Java).....	800
13.3 Obtaining Lifecycle Rules (SDK for Java).....	813
13.4 Deleting Lifecycle Rules (SDK for Java).....	825
14 Cross-Origin Resource Sharing (SDK for Java).....	831
14.1 Overview (SDK for Java).....	831
14.2 Configuring a CORS Rule (SDK for Java).....	831
14.3 Obtaining a CORS Rule (SDK for Java).....	838
14.4 Deleting a CORS Rule (SDK for Java).....	843
15 Logging (SDK for Java).....	847
15.1 Overview (SDK for Java).....	847
15.2 Configuring Logging for a Bucket (SDK for Java).....	847
15.3 Obtaining the Logging Configuration of a Bucket (SDK for Java).....	859
16 Static Website Hosting (SDK for Java).....	868
16.1 Overview (SDK for Java).....	868
16.2 Hosting Website Files in a Bucket (SDK for Java).....	868
16.3 Configuring Static Website Hosting (SDK for Java).....	893

16.4 Obtaining Static Website Hosting Configurations (SDK for Java).....	905
16.5 Deleting Static Website Hosting Configurations (SDK for Java).....	913
17 Client-Side Encryption (SDK for Java).....	917
17.1 Overview (SDK for Java).....	917
17.2 Client-Side Encryption APIs (SDK for Java).....	919
18 Fault Locating (SDK for Java).....	929
18.1 Methods (SDK for Java).....	929
18.2 Resolving Dependency Missing or Conflicts (SDK for Java).....	930
18.3 Other Notable Issues (SDK for Java).....	931
19 Troubleshooting (SDK for Java).....	935
19.1 HTTP Status Codes (SDK for Java).....	935
19.2 OBS Server-side Error Codes (SDK for Java).....	937
19.3 SDK Exceptions (SDK for Java).....	947
19.4 SDK Common Response Headers (SDK for Java).....	947
19.5 Log Analysis (SDK for Java).....	948
20 FAQs (SDK for Java).....	950
20.1 How Can I Set an Object to Be Accessible to Anonymous Users? (Java SDK).....	950
20.2 What Is the Retry Mechanism of SDK? (Java SDK).....	950
20.3 How Do I Obtain the Static Website Access Address of a Bucket? (Java SDK).....	951
20.4 How Do I Obtain an Object URL? (Java SDK).....	951
20.5 How to Improve the Speed of Uploading Large Files over the Public Network? (Java SDK).....	951
20.6 How Can I Perform a Multipart Upload? (Java SDK).....	952
20.7 How Can I Perform a Download in Multipart Mode? (Java SDK).....	952
20.8 How Do I Confirm that an Upload is Successful If I Upload an Object to Overwrite an Existing Object with the Same Name in a Bucket? (Java SDK).....	953
20.9 Does the SDK Support Uploading, Downloading, or Copying Objects in a Batch? (Java SDK).....	953
20.10 What Is Content-Type (MIME)? (Java SDK).....	954
20.11 How Do I Get My Account ID and User ID?.....	963
20.12 How Do I Limit the Rate During Object Upload? (SDK for Java).....	963
20.13 How Do I Limit the Rate During Object Download? (SDK for Java).....	964
20.14 How Do I Configure the referer Header? (SDK for Java).....	965

1 Before You Start (SDK for Java)

This section describes version compatibility and important notes about Object Storage Service (OBS) SDK for Java.

Compatibility

- Recommended JDK versions: JDK 8 or later
- Third-party dependencies: This version is not completely compatible with earlier versions (2.1.x). OkHttp 3 should be used to replace HttpClient4.x.
- Namespace: This version is compatible with earlier versions (2.1.x). All external APIs are contained in the **com.obs.services**, **com.obs.services.model**, and **com.obs.services.exception** packages.
- API functions: This version is compatible with earlier versions (2.1.x).
- The earlier 2.X versions are no longer maintained. You are advised to upgrade them to the latest version as soon as possible.

Important Notes

- You can learn about how to call an API through the OBS SDK for Java by referring to [Using an OBS Client](#).
- After an API is called using an instance of **ObsClient**, if no exception is thrown, the return value is valid. If an exception is thrown, the operation fails. For details about errors, see [SDK Exceptions \(SDK for Java\)](#).
- After an API is successfully called using an instance of **ObsClient**, a class or sub-class instance of [HeaderResponse](#) that contains response headers will be returned.
- Some features are available only in some regions. If an API call returns the 405 HTTP status code, check whether the region supports this feature.

2 API Overview (SDK for Java)

Bucket APIs

Table 2-1 lists the bucket-related APIs.

Table 2-1 Bucket APIs

API	Method	Function	Source Code Example
Creating a Bucket (SDK for Java)	obsClient.createBucket(CreateBucketRequest request)	Creates a bucket and configures the storage class, region, and ACL for the bucket.	Submit a service ticket.
Obtaining a Bucket List (SDK for Java)	obsClient.listBuckets(ListBucketsRequest request)	Lists all buckets that meet the specified conditions under the current account and returns them in alphabetical order.	Submit a service ticket.
Deleting a Bucket (SDK for Java)	obsClient.deleteBucket(String bucketName)	Deletes an empty bucket (the name of a deleted bucket can be reused at least 30 minutes after the deletion).	Submit a service ticket.

API	Method	Function	Source Code Example
Checking Whether a Bucket Exists (SDK for Java)	obsClient.headBucket(String bucketName)	Checks whether a bucket exists (HTTP status code 200 indicates the bucket exists, while 404 indicates it does not).	Submit a service ticket.
Obtaining Bucket Metadata (SDK for Java)	obsClient.getBucketMetadata(BucketMetadataInfoRequest request)	Returns information about a bucket, including the storage class, region, CORS rules, and redundancy policy.	Submit a service ticket.
Setting a Bucket ACL (SDK for Java)	obsClient.setBucketACL(String bucketName, AccessControlList acl)	Sets an ACL for a bucket.	Submit a service ticket.
Obtaining a Bucket ACL (SDK for Java)	obsClient.getBucketACL(String bucketName)	Returns the ACL of a bucket.	Submit a service ticket.
Configuring a Bucket Policy (SDK for Java)	obsClient.setBucketPolicy(String bucketName, String policy)	Sets a bucket policy.	Submit a service ticket.
Obtaining the Policy of a Bucket (SDK for Java)	obsClient.getBucketPolicy(String bucketName)	Returns a bucket policy.	Submit a service ticket.
Deleting a Bucket Policy (SDK for Java)	obsClient.deleteBucketPolicy(String bucketName)	Deletes a bucket policy (204 No Content is returned if the policy is deleted successfully or actually does not exist).	Submit a service ticket.
Obtaining the Region of a Bucket (SDK for Java)	obsClient.getBucketLocation(String bucketName)	Returns the region where the bucket is created.	Submit a service ticket.

API	Method	Function	Source Code Example
Obtaining Storage Information of a Bucket (SDK for Java)	<code>obsClient.getBucketStorageInfo(String bucketName)</code>	Returns the storage information about a bucket, including the storage usage and the object count in the bucket.	Submit a service ticket.
Configuring a Storage Quota (SDK for Java)	<code>obsClient.setBucketQuota(String bucketName, BucketQuota bucketQuota)</code>	Sets a limit on the capacity of a bucket.	Submit a service ticket.
Obtaining a Bucket Storage Quota (SDK for Java)	<code>obsClient.getBucketQuota(String bucketName)</code>	Returns the quota of a bucket (0 indicates there is no upper limit to the capacity of the bucket).	Submit a service ticket.
Configuring a Storage Class for a Bucket (SDK for Java)	<code>obsClient.setBucketStoragePolicy(String bucketName, BucketStoragePolicy-Configuration bucketStorage)</code>	Specifies the storage class for a bucket (after the bucket storage class is configured, if you do not specifically configure the storage class for objects in that bucket, those objects will inherit the storage class of the bucket by default).	Submit a service ticket.
Obtaining the Storage Class of a Bucket (SDK for Java)	<code>obsClient.getBucketStoragePolicy(String bucketName)</code>	Returns the storage class of a bucket.	Submit a service ticket.

API	Method	Function	Source Code Example
Configuring an Inventory Rule (SDK for Java)	obsClient.setInventoryConfiguration(SetInventoryConfigurationRequest request)	Configures an inventory rule for a bucket (you can specify the object attributes to include in inventories, such as the object version, size, storage class, and last modification).	Submit a service ticket.
Obtaining an Inventory Rule (SDK for Java)	obsClient.getInventoryConfiguration(GetInventoryConfigurationRequest request)	Returns a bucket inventory rule specified by the rule ID.	Submit a service ticket.
Listing Inventory Rules (SDK for Java)	obsClient.listInventoryConfiguration(ListInventoryConfigurationRequest request)	Returns all inventory rules of a bucket in a single response.	Submit a service ticket.
Deleting an Inventory Rule (SDK for Java)	obsClient.deleteInventoryConfiguration(DeleteInventoryConfigurationRequest request)	Deletes a bucket inventory rule specified by the rule ID.	Submit a service ticket.

Object APIs

[Table 2-2](#) lists object-related APIs.

Table 2-2 Object APIs

API	Method	Function	Source Code Example
Uploading an Object - Streaming (SDK for Java)	obsClient.putObject(PutObjectRequest request)	Uploads local files of any type that are smaller than 5 GB to a bucket in streaming mode.	Submit a service ticket.
Uploading an Object - File-Based (SDK for Java)	obsClient.putObject(PutObjectRequest request)	Uploads local files of any type to a bucket over the Internet.	-

API	Method	Function	Source Code Example
Obtaining the Upload Progress (SDK for Java)	PutObjectRequest.setProgressListener(ProgressListener progressListener)	Returns the progress of uploading an object.	-
Creating a Folder (SDK for Java)	obsClient.putObject(PutObjectRequest request)	Creates a folder in a bucket to categorize the data.	Submit a service ticket.
Configuring Object Metadata (SDK for Java)	obsClient.setObjectMetadata(SetObjectMetadataRequest request)	Sets object attributes (such as the size, MIME type, MD5 value, storage class, or user-defined metadata) when uploading an object in streaming, file-based, or multipart mode, or when copying an object.	Submit a service ticket.
Initiating a Multipart Upload (SDK for Java)	obsClient.initiateMultipartUpload(InitiateMultipartUploadRequest request)	Initiates a multipart upload and returns a globally unique upload ID.	Submit a service ticket.
Uploading a Part (SDK for Java)	obsClient.uploadPart(UploadPartRequest request)	Uploads parts to the bucket based on the upload ID returned by the preceding API.	Submit a service ticket.
Assembling Parts (SDK for Java)	obsClient.completeMultipartUpload(CompleteMultipartUploadRequest request)	Completes a multipart upload based on the multipart upload ID and information about the uploaded parts (including PartNumber and ETag).	Submit a service ticket.

API	Method	Function	Source Code Example
Aborting a Multipart Upload (SDK for Java)	<code>obsClient.abortMultipartUpload(AbortMultipartUploadRequest request)</code>	Aborts a multipart upload specified by the upload ID in a bucket.	-
Listing Uploaded Parts (SDK for Java)	<code>obsClient.listParts(ListPartsRequest request)</code>	Returns the uploaded parts in a bucket based on the specified multipart upload ID.	Submit a service ticket.
Listing Multipart Uploads (SDK for Java)	<code>obsClient.listMultipartUploads(ListMultipartUploadsRequest request)</code>	Lists ongoing multipart uploads.	-
Configuring Lifecycle Rules (SDK for Java)	<code>obsClient.putObject(PutObjectRequest request)</code>	Configures a lifecycle rule for objects to periodically delete objects in the bucket or transition object storage classes (the object expiration time set using this API takes precedence over that set in a bucket lifecycle rule).	-
Uploading an Object - Append (SDK for Java)	<code>obsClient.appendObject(AppendObjectRequest request)</code>	Appends content to an existing object.	-
Uploading an Object - Resumable (SDK for Java)	<code>obsClient.uploadFile(UploadFileRequest request)</code>	Provides the resumable function based on the API for multipart uploads to better respond to network disconnections or program crashes.	-

API	Method	Function	Source Code Example
Uploading an Object - Browser-Based (SDK for Java)	obsClient.createPostSignature(PostSignatureRequest request)	Uploads an object up to 5 GB in size to a bucket in the HTML form.	Submit a service ticket.
Downloading an Object - Streaming (SDK for Java)	obsClient.getObject(GetObjectRequest request)	Downloads an object from OBS to a local directory or memory (the returned results contain the object name, attributes, input stream, and bucket information).	Submit a service ticket.
Downloading an Object - Range-Based (SDK for Java)	obsClient.getObject(GetObjectRequest request)	Downloads the partial data of an object.	-
Obtaining the Download Progress (SDK for Java)	GetObjectRequest.setProgressListener(ProgressListener progressListener)	Returns the progress of downloading an object.	-
Downloading an Object - Conditional (SDK for Java)	obsClient.getObject(GetObjectRequest request)	Returns the objects that meet one or more specified conditions.	-
Rewriting Response Headers (SDK for Java)	obsClient.getObject(GetObjectRequest request)	Rewrites the following HTTP/HTTPS response headers when downloading an object: Content-Type , Content-Language , Expires , Cache-Control , Content-Disposition , and Content-Encoding .	-

API	Method	Function	Source Code Example
Obtaining User-defined Metadata (SDK for Java)	obsClient.getObject(GetObjectRequest request)	Returns the user-defined object metadata after the object is successfully downloaded.	Submit a service ticket.
Restoring a Cold Object (SDK for Java)	obsClient.restoreObject(RestoreObjectRequest request)	Restores and downloads a Cold object.	Submit a service ticket.
Downloading an Object - Resumable (SDK for Java)	obsClient.downloadFile(DownloadFileRequest request)	Adds the resumable function to the partial download API.	-
Configuring Object Metadata (SDK for Java)	obsClient.setObjectMetadata(SetObjectMetadataRequest request)	Configures the object metadata.	-
Obtaining Object Metadata (SDK for Java)	obsClient.getObjectMetadata(GetObjectMetadataRequest request)	Returns the object metadata.	-
Configuring an Object ACL (SDK for Java)	obsClient.setObjectAcl(SetObjectAclRequest request)	Sets an ACL for an object when uploading the object or modifies the ACL of an existing object by calling an ACL API.	Submit a service ticket.
Obtaining an Object ACL (SDK for Java)	obsClient.getObjectAcl(GetObjectAclRequest request)	Obtains the ACL of an object in a specified bucket.	Submit a service ticket.
Listing Objects (SDK for Java)	obsClient.listObjects(ListObjectsRequest request)	Lists some or all objects in a bucket based on the specified prefix, object count, and start position and returns the objects in alphabetical order.	Submit a service ticket.

API	Method	Function	Source Code Example
Deleting an Object (SDK for Java)	obsClient.deleteObject(deleteObjectRequest)	Deletes an object from a bucket.	Submit a service ticket.
Batch Deleting Objects (SDK for Java)	obsClient.deleteObjects(deleteObjectsRequest)	Deletes objects from a bucket in a batch. (Deleted objects cannot be restored.)	Submit a service ticket.
Copying an Object (SDK for Java)	obsClient.copyObject(copyObjectRequest)	Creates a copy up to 5 GB for an object.	Submit a service ticket.
Copying an Object - Multipart (SDK for Java)	obsClient.copyPart(copyPartRequest)	Copies parts to a specified bucket based on the multipart upload ID returned in initiating a multipart upload.	Submit a service ticket.
Checking Whether an Object Exists (SDK for Java)	doesObjectExist(final GetObjectMetadataRequest request)	Checks whether an object exists (HTTP status code 200 indicates the object exists, while 404 indicates the object or bucket does not exist).	-

Temporarily Authorized Access APIs

[Table 2-3](#) lists the APIs related to temporarily authorized access.

Table 2-3 Temporarily Authorized Access APIs

API	Method	Function	Source Code Example
Accessing OBS Using a Signed URL (SDK for Java)	obsClient.createTemporarySignature(TemporarySignatureRequest request)	Creates a signed URL and specifies the expiration time for the URL to grant visitors temporary access. (To allow other users to perform a temporary operation such as upload, you need to generate a URL for the corresponding operation and offer that to users.)	Submit a service ticket.

Versioning APIs

[Table 2-4](#) lists the APIs related to versioning.

Table 2-4 Versioning APIs

API	Method	Function	Source Code Example
Configuring Versioning for a Bucket (SDK for Java)	obsClient.setBucketVersioning(final SetBucketVersioningRequest request)	Configures versioning for objects in a bucket.	Submit a service ticket.
Obtaining the Versioning Status of a Bucket (SDK for Java)	obsClient.getBucketVersioning(final BaseBucketRequest request)	Returns the versioning status of a bucket.	Submit a service ticket.
Obtaining an Object Version (SDK for Java)	obsClient.getObject(GetObjectRequest request)	Returns a specified object version.	-
Copying an Object Version (SDK for Java)	obsClient.copyObject(CopyObjectRequest request)	Creates a copy up to 5 GB for a versioned object.	-

API	Method	Function	Source Code Example
Restoring a Specific Cold Object Version (SDK for Java)	obsClient.restoreObject(RestoreObjectRequest request)	Restores a Cold object version. (To download a Cold object version, you need to restore it first.)	-
Listing Object Versions (SDK for Java)	obsClient.listVersions(ListVersionsRequest request)	Lists some or all object versions in a bucket based on the specified prefix, version count, and start position and returns the versions in alphabetical order.	Submit a service ticket.
Setting an ACL for an Object Version (SDK for Java)	obsClient.setObjectAcl(SetObjectAclRequest request)	Sets an ACL for an object version.	-
Obtaining the ACL of an Object Version (SDK for Java)	obsClient.getObjectAcl(GetObjectAclRequest request)	Returns the ACL of an object version.	-
Deleting an Object Version (SDK for Java)	obsClient.deleteObject(DeleteObjectRequest request)	Deletes the ACL of an object version.	-
Batch Deleting Object Versions (SDK for Java)	obsClient.deleteObjects(DeleteObjectsRequest deleteRequest)	Deletes object versions from a bucket in a batch. (Deleted object versions cannot be restored.)	Submit a service ticket.

Lifecycle Rule APIs

[Table 2-5](#) lists the APIs related to lifecycle rules.

Table 2-5 Lifecycle Rule APIs

API	Method	Function	Source Code Example
Setting Lifecycle Rules (SDK for Java)	obsClient.setBucketLifecycle(final SetBucketLifecycleRequest request)	Sets a lifecycle rule for a bucket to periodically delete objects in the bucket or transition object storage classes.	Submit a service ticket.
Obtaining Lifecycle Rules (SDK for Java)	obsClient.getBucketLifecycle(final BaseBucketRequest request)	Returns a lifecycle rule of a bucket.	Submit a service ticket.
Deleting Lifecycle Rules (SDK for Java)	obsClient.deleteBucketLifecycle(final BaseBucketRequest request)	Deletes a lifecycle rule of a bucket.	Submit a service ticket.

CORS rule APIs

[Table 2-6](#) lists the APIs related to CORS rules.

Table 2-6 CORS rule APIs

API	Method	Function	Source Code Example
Configuring a CORS Rule (SDK for Java)	obsClient.setBucketCors(final SetBucketCorsRequest request)	Sets a CORS rule for a bucket (the existing rules will be overwritten by the new ones).	Submit a service ticket.
Obtaining a CORS Rule (SDK for Java)	obsClient.getBucketCors(final BaseBucketRequest request)	Returns a CORS rule of a bucket.	Submit a service ticket.
Deleting a CORS Rule (SDK for Java)	obsClient.deleteBucketCors(final BaseBucketRequest request)	Deletes a CORS rule of a bucket.	Submit a service ticket.

Bucket Logging APIs

[Table 2-7](#) lists the logging APIs.

Table 2-7 Bucket logging APIs

API	Method	Function	Source Code Example
Configuring Logging for a Bucket (SDK for Java)	obsClient.setBucketLogging(final SetBucketLoggingRequest request)	Enables logging for a source bucket, specifies a target bucket for storing log files, and configures the name prefixes and access permissions for log files.	Submit a service ticket.
Obtaining the Logging Configuration of a Bucket (SDK for Java)	obsClient.getBucketLogging(final BaseBucketRequest request)	Returns the logging configuration of a bucket.	Submit a service ticket.

Static Website Hosting APIs

[Table 2-8](#) lists the APIs related to static website hosting.

Table 2-8 Static website hosting APIs

API	Method	Function	Source Code Example
Hosting Website Files in a Bucket (SDK for Java)	- obsClient.putObject(PutObjectRequest request) - obsClient.setObjectAcl(SetObjectAclRequest acl)	Uploads the files of a static website to a specified bucket in OBS, configures the public read permission for the files, and enables static website hosting for the bucket.	-
Configuring Static Website Hosting (SDK for Java)	obsClient.setBucketWebsite(final SetBucketWebsiteRequest request)	Configures website hosting for a bucket.	Submit a service ticket.
Obtaining Static Website Hosting Configurations (SDK for Java)	obsClient.getBucketWebsite(final BaseBucketRequest request)	Returns the website configuration of a bucket.	Submit a service ticket.

API	Method	Function	Source Code Example
Deleting Static Website Hosting Configurations (SDK for Java)	obsClient.deleteBucketWebsite(final BaseBucketRequest request)	Deletes the website configuration of a bucket.	Submit a service ticket.

3 Preparations (SDK for Java)

Before using OBS SDK for Java to access OBS, you need to prepare the service and development environments. To prepare the service environment, you will need an account and access keys. Both of them are necessary for interaction between OBS SDK and OBS. To ensure successful SDK installation and SDK-based code development and running, you should also set up a local development environment, for example, installing dependencies and development tools.

Preparing Access Keys

Access keys consist of two parts: an access key ID (AK) and a secret access key (SK). OBS uses access keys to sign requests to make sure that only authorized accounts can access specified OBS resources. Programmatic access must be enabled for an IAM user before the IAM user can get access keys. Access keys are explained as follows:

- One AK maps to only one user but one user can have multiple AKs. OBS authenticates users by their AKs.
- An SK is required for accessing OBS. Authentication information is generated based on the SK and request headers. AKs and SKs are in one-to-one match.

Access keys are classified into permanent access keys (AK/SK) and temporary access keys (AK/SK and security token). Each user can create at most two permanent access keys. Temporary access keys must be used within a given validity period. Once expired, they must be requested again. For security purposes, you are advised to use temporary access keys to access OBS. If you want to use permanent access keys, periodically update them. The following describes how to obtain two types of access keys.

- To get permanent access keys, do as follows:
 - a. Log in to the management console.
 - b. In the upper right corner, hover over the username and choose **My Credentials**.
 - c. On the **My Credentials** page, click **Access Keys** in the navigation pane.
 - d. On the **Access Keys** page, click **Create Access Key**.
 - e. In the displayed dialog box, enter the login password and verification code.

 NOTE

- If you have not bound an email address or a mobile number yet, only the login password is required.
 - If you have bound both an email address and a mobile number, you can use either of them for verification.
- f. Click **OK**.
- g. Click **Download**. The access key file is automatically saved to your browser's default download path.
- h. Open the downloaded **credentials.csv** file to obtain the access keys (AK and SK).

 NOTE

- Each user can create a maximum of two valid access key pairs.
 - Keep AKs and SKs properly to prevent information leakage. If you click **Cancel** in the download dialog box, the access keys will not be downloaded and cannot be downloaded later. You can create a new AK/SK pair if needed.
- To get temporary access keys, refer to the following:
Temporary access keys are issued by the system and are only valid for 15 minutes to 24 hours. Once expired, they must be requested again. They follow the principle of least privilege. When a temporary AK/SK pair is used for authentication, a security token must be used at the same time.

Setting Up a Development Environment

- Download a recommended version of JDK from the [Oracle's official website](#) and install it. Recommended JDK versions: JDK 8 or later
- Download the latest version of Eclipse IDE for Java Developers from the [Eclipse's official website](#) and install it.

4 SDK Download and Installation (SDK for Java)

Downloading OBS SDK for Java

You can obtain the latest version of Java SDK and Java SDK API document as follows:

OBS Java SDK API document: Submit a service ticket.

Java SDK API document: Submit a service ticket.

5 Getting Started (SDK for Java)

This section introduces how to use OBS SDK for Java to perform some basic actions, such as creating a bucket, and uploading, downloading, listing, and deleting objects.

Preparations

Ensure you have made the following preparations before using the SDK:

1. [Before You Start \(SDK for Java\)](#): Select a proper SDK version.
2. [Preparations \(SDK for Java\)](#): Prepare the service and development environments.
3. [SDK Download and Installation \(SDK for Java\)](#): Download and install the OBS SDK for Java.

Creating a Bucket

The example below shows how to create a bucket named **examplebucket**, and configure access, storage class, region, and redundancy type for the bucket. For more details about how to create a bucket, see [Creating a Bucket \(SDK for Java\)](#).

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.AvailableZoneEnum;
import com.obs.services.model.CreateBucketRequest;
import com.obs.services.model.ObsBucket;
import com.obs.services.model.StorageClassEnum;

public class CreateBucket001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");

        // Enter the endpoint corresponding to the region where the bucket is to be created.
```

```
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    CreateBucketRequest request = new CreateBucketRequest();
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Example bucket location
    request.setBucketName(exampleBucket);
    // Set the bucket ACL to Private (the default value).
    request.setAcl(AccessControlList.REST_CANNED_PRIVATE);
    // Set the bucket storage class to Standard.
    request.setBucketStorageClass(StorageClassEnum.STANDARD);
    request.setLocation(exampleLocation);
    // Create a bucket.
    ObsBucket bucket = obsClient.createBucket(request);
    // The bucket is created.
    System.out.println("CreateBucket successfully");
    System.out.println("RequestId:" + bucket.getRequestId());

} catch (ObsException e) {
    System.out.println("CreateBucket failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message: " + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("CreateBucket failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Uploading an Object

The example below shows how to upload two local files **localfile** and **localfile2** to the bucket **examplebucket**, and specify the names of the objects created as **objectkey** and **objectkey2** respectively. For more details about uploading an object, see [Object Upload \(SDK for Java\)](#).

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
```

```
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk,endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Upload files.
    // localfile indicates the path of the local file to be uploaded, in which the file name must be
specified.
    PutObjectRequest request = new PutObjectRequest();
    request.setBucketName("examplebucket");
    request.setObjectKey("objectkey");
    request.setFile(new File("localfile"));
    obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Downloading an Object

The example below shows how to download the object **objectname** from the bucket **examplebucket**. For more details about downloading an object, see [Object Download \(SDK for Java\)](#).

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsObject;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
public class GetObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
```

```
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Download the object using streaming.
    ObsObject obsObject = obsClient.getObject("examplebucket", "objectname");
    // Read the object content.
    System.out.println("Object content:");
    InputStream input = obsObject.getObjectContent();
    byte[] b = new byte[1024];
    ByteArrayOutputStream bos = new ByteArrayOutputStream();
    int len;
    while ((len = input.read(b)) != -1) {
        bos.write(b, 0, len);
    }
    System.out.println("getObjectContent successfully");
    System.out.println(new String(bos.toByteArray()));
    bos.close();
    input.close();
} catch (ObsException e) {
    System.out.println("getObjectContent failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObjectContent failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Listing Objects

The example below shows how to list objects in the bucket **examplebucket**. For more details about object listing, see [Listing Objects \(SDK for Java\)](#).

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectListing;
import com.obs.services.model.ObsObject;
public class ListObjects001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
```

```
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk,endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Listing objects.
    ObjectListing result = obsClient.listObjects("examplebucket");
    for (ObsObject obsObject : result.getObjects()) {
        System.out.println("listObjects successfully");
        System.out.println("ObjectKey:" + obsObject.getObjectKey());
        System.out.println("Owner:" + obsObject.getOwner());
    }
} catch (ObsException e) {
    System.out.println("listObjects failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listObjects failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Deleting an Object

The example below shows how to delete the object **objectname** from the bucket **examplebucket**. For more details about deleting an object, see [Deleting an Object \(SDK for Java\)](#).

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class DeleteObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
    }
}
```

```
try {
    // Delete the object.
    obsClient.deleteObject("examplebucket", "objectname");
    System.out.println("deleteObject successfully");
} catch (ObsException e) {
    System.out.println("deleteObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Using an OBS Client

The example below shows how to use an ObsClient instance.

```
// Make sure there is only one ObsClient instance in the whole project.
// ObsClient is thread-safe and can be used in concurrency scenarios.
ObsClient obsClient = null;
try
{
    String endPoint = "https://your-endpoint";
    // Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes, encrypt your access keys
    and store them in the configuration file or environment variables. In this example, access keys are stored in
    environment variables for identity authentication. Before running the code in this example, configure
    environment variables ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID.
    // Obtain an AK/SK pair on the management console.
    String ak = System.getenv("ACCESS_KEY_ID");
    String sk = System.getenv("SECRET_ACCESS_KEY_ID");
    // Create an ObsClient instance.
    obsClient = new ObsClient(ak, sk, endPoint);
    // Call an API to perform an operation, for example, uploading an object.
    HeaderResponse response = obsClient.putObject("bucketname", "objectname", new File("localfile")); //
    localfile indicates the path of the local file to be uploaded. Use the file path in your case.
    System.out.println(response);
}
catch (ObsException e)
{
    System.out.println("HTTP Code: " + e.getResponseCode());
    System.out.println("Error Code:" + e.getErrorCode());
    System.out.println("Error Message: " + e.getErrorMessage());

    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    Map<String, String> headers = e.getResponseHeaders();// Check all map entries and print all headers
    with errors reported.
    if(headers != null){
        for (Map.Entry<String, String> header : headers.entrySet()) {
            System.out.println(header.getKey()+"."+header.getValue());
        }
    }
    e.printStackTrace();
}finally{
    // Close the ObsClient instance. If the instance is used globally, skip this step.
    // After the ObsClient instance is closed by calling ObsClient.close, it cannot be used again.
    if(obsClient != null){
        try
```

```
    {  
        // obsClient.close();  
    }  
    catch (IOException e)  
    {  
    }  
}
```

6 Initialization (SDK for Java)

6.1 Overview (SDK for Java)

After the SDK is installed, you need to create and configure an OBS client and configure the logging for the SDK to complete the initialization.

Table 6-1 Initialization

Task	Sub-task	M a n d a t o r y (Y e s / N o)	Description
Creating and Configuring an OBS Client (SDK for Java)	Creating an OBS client	Yes	OBS clients offer a series of APIs for you to use OBS SDK for Java to interact with OBS. Before sending a request to OBS, you need to create and initialize an OBS client (an ObsClient instance).
	Configuring access credentials	Yes	For security purposes, you need to configure access credentials for the client so that the server can verify the validity of requests.

Task	Sub-task	M a n d a t o r y (Y e s / N o)	Description
	Configuring the endpoint and other parameters	Yes for the endpoint; No for other parameters	• To quickly create and retrieve your resources, you need to configure an endpoint for the client so that the server can know where to create or retrieve resources. • You can configure other parameters for the client based on your needs.
Configuring SDK Logging (SDK for Java)	Configuring a log level	No	OBS SDK for Java offers logging based on the open-source library of Apache Log4j 2. The Log4j 2 framework classifies log files into different levels based on their priorities. You can specify a log level to customize the log storage scope. By default, OBS SDK for Java saves log files of the WARN and higher levels to local directories.
	Configuring a storage path for log files	No	Configure a local path to store logs.

6.2 Creating and Configuring an OBS Client (SDK for Java)

Scenarios

This section describes how to create and configure an OBS client. If you already have a client, skip this section and obtain the API details by referring to [API Overview \(SDK for Java\)](#).

Prerequisites

Ensure you have completed the following preparations:

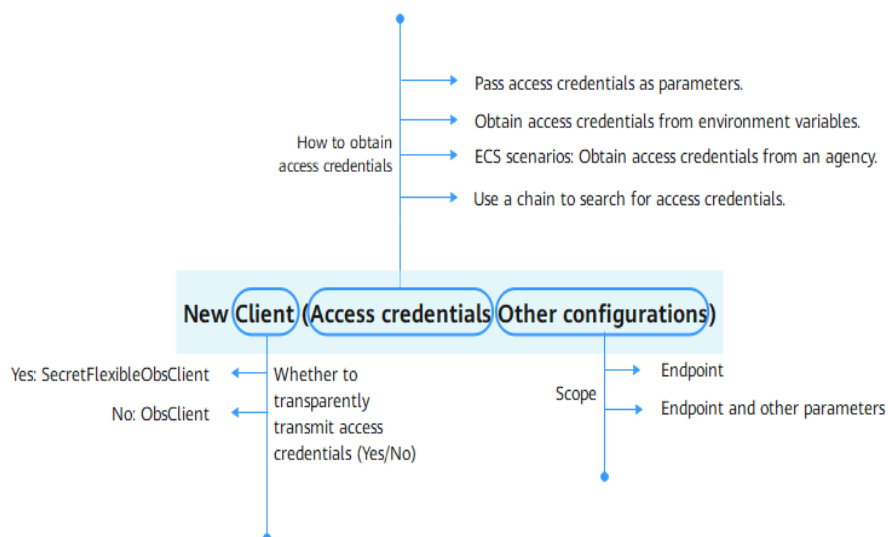
1. Select a proper SDK version by referring to [Before You Start \(SDK for Java\)](#).
2. Prepare the service and development environments by referring to [Preparations \(SDK for Java\)](#).
3. Download and install the OBS SDK for Java by referring to [SDK Download and Installation \(SDK for Java\)](#).

Important Notes

- Make sure to configure the client at the time you create it. After the client is created, you will not be able to configure it.
- If you use a temporary AK/SK pair, you can call **ObsClient.refresh("yourAccessKey", "yourSecretKey", "yourSecurityToken")** to refresh to keep the pair valid, without the need to create a new ObsClient instance.
- You are advised to create an ObsClient instance at the initialization and use it for the whole project. Creating multiple clients affects high-concurrency performance.
- ObsClient is thread-safe and can be used in concurrency scenarios.
- If a client is closed by calling **ObsClient.close**, it cannot be used again. To ensure the whole project uses only one client, you are advised not to proactively close the client.

Overview

Figure 6-1 Creating and configuring a client



You can select a client class, a method of obtaining access credentials, and other parameters by referring to [Figure 6-1](#).

Table 6-2 Creating and configuring an OBS client

Cat eg ory	Option	Description	Sample Code
Clie nt	Client that does not transparently transmit access credentials	If you want to configure access credentials only once and use them for all API calls, use ObsClient .	ObsClient
	Client that transparently transmits access credentials	If you need to configure access credentials for each API call, use SecretFlexibleObsClient , which is inherited from ObsClient .	SecretFlexibleObsClient
Acc ess cre de nti als	Passing access credentials as parameters	Create an ObsClient instance and pass access credentials as parameters.	• Passing Access Credentials as Parameters • BasicObsCredentialsProvider

Cat eg ory	Option	Description	Sample Code
	Obtaining access credentials from environment variables	Create an <code>ObsClient</code> instance and use <code>EnvironmentVariableObsCredentialsProvider</code> to obtain access credentials from environment variables.	EnvironmentVariableObsCredentialsProvider
	Obtaining access credentials from an agency in ECS scenarios	Create an <code>ObsClient</code> instance and use <code>EcsObsCredentialsProvider</code> to automatically obtain temporary access credentials from the ECS. The access credentials are periodically updated automatically. NOTICE 1. To use <code>EcsObsCredentialsProvider</code>, ensure that the application runs on an ECS that has an IAM agency with OBS permissions. 2. Ensure that the server and the environment where the application is deployed have the same UTC time, or temporary access keys may fail to be updated in a timely manner.	EcsObsCredentialsProvider
	Using a chain to search for access credentials	You can use this method to search for access keys in a chain in a predefined sequence. The first pair of access keys obtained is used to create an <code>ObsClient</code> instance. By default, environment variables are checked first and then the ECS agencies are checked to obtain access credentials. You cannot customize the search method or sequence.	OBSCredentialsProviderChain
Other configurations	Configuring only the endpoint	Other than access credentials, only the endpoint is configured.	Configuring Only the Endpoint

Cat eg ory	Option	Description	Sample Code
	Configuring the endpoint and other parameters	In addition to the endpoint, you need to configure the HTTP proxy, Socket timeout, and other parameters. You can use the ObsConfiguration class to configure the ObsClient instance. For details about the supported parameters, see Table 6-3 .	ObsConfiguration

Code Examples

- **Using the client class that does not transparently transmit access credentials to directly configure access credentials and configuring the endpoint**
 - Using permanent access keys (AKs/SKs)

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-endpoint";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes, encrypt your access keys and store them in the configuration file or environment variables. In this example, access keys are stored in the environment variables for identity authentication. Before running the code in this example, configure environment variables ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID.// Obtain an AK/SK pair on the management console.String ak = System.getenv("ACCESS_KEY_ID");String sk = System.getenv("SECRET_ACCESS_KEY_ID");// Create an ObsClient instance.ObsClient obsClient = new ObsClient(ak, sk, endPoint); // Use the instance to access OBS. // Close the instance.obsClient.close();
```
 - Using temporary access credentials (AKs/SKs and security tokens)

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-endpoint";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes, encrypt your access keys and store them in the configuration file or environment variables. In this example, access keys are stored in the environment variables for identity authentication. Before running the code in this example, configure environment variables ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID.// Obtain an AK/SK pair on the management console.String ak = System.getenv("ACCESS_KEY_ID");String sk = System.getenv("SECRET_ACCESS_KEY_ID");String securityToken = System.getenv("SECRET_TOKEN");// Create an ObsClient instance.ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint); // Use the instance to access OBS. // Close the instance.obsClient.close();
```
 - BasicObsCredentialsProvider

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-endpoint";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes, encrypt your access keys and store them in the configuration file or environment variables. In this example, access keys are stored in the environment variables for identity authentication. Before running the code in this example, configure environment variables ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID.// Obtain an AK/SK pair on the management console.String ak = System.getenv("ACCESS_KEY_ID");String sk = System.getenv("SECRET_ACCESS_KEY_ID");// Create an ObsClient instance.ObsClient obsClient = new ObsClient(new BasicObsCredentialsProvider(ak, sk), endPoint); // Use the instance to access OBS. // Close the instance.obsClient.close();
```
- **Using the SecretFlexibleObsClient client class that transparently transmits access credentials**

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-endpoint";// Create an instance of the ObsConfiguration class.ObsConfiguration config = new ObsConfiguration();config.setEndPoint(endPoint);// Create an instance of SecretFlexibleObsClient.SecretFlexibleObsClient obsClient = new SecretFlexibleObsClient(config);// Use the instance of SecretFlexibleObsClient to access OBS.// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes, encrypt your access keys and store them in the configuration file or environment variables. In this example, access keys are stored in the environment variables for
```

```
identity authentication. Before running the code in this example, configure environment variables
ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID.// Obtain an AK/SK pair on the management
console.String ak1 = System.getenv("ACCESS_KEY_ID");String sk1 =
System.getenv("SECRET_ACCESS_KEY_ID");obsClient.listBuckets(ak1, sk1);String ak2 =
System.getenv("ACCESS_KEY_ID");String sk2 =
System.getenv("SECRET_ACCESS_KEY_ID");obsClient.listBuckets(ak2, sk2);// Close the
instance.obsClient.close();
```

- **Using EnvironmentVariableObsCredentialsProvider to obtain access credentials from environment variables**

In this method, you need to define **OBS_ACCESS_KEY_ID** and **OBS_SECRET_ACCESS_KEY** in the environment variables to represent the permanent AK and SK respectively.

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-
endpoint";// Create an ObsClient instance.ObsClient obsClient = new ObsClient(new
EnvironmentVariableObsCredentialsProvider(), endPoint); // Use the instance to access OBS.    //
Close the instance.obsClient.close();
```

- **Using EcsObsCredentialsProvider to obtain access credentials from an agency in the ECS scenarios**

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-
endpoint";// Create an ObsClient instance.ObsClient obsClient = new ObsClient(new
EcsObsCredentialsProvider(), endPoint); // Use the instance to access OBS.    // Close the
instance.obsClient.close();
```

- **Using OBSCredentialsProviderChain to obtain access credentials from a chain**

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-
endpoint";// Create an ObsClient instance.ObsClient obsClient = new ObsClient(new
OBSCredentialsProviderChain(), endPoint); // Use the instance to access OBS.    // Close the
instance.obsClient.close();
```

- **Using ObsConfiguration to configure parameters for the client**

- **KeyManagerFactory:** By configuring **KeyManagerFactory**, you can save a certificate on your local PC and check whether the certificate returned by the server is correct.

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-
endpoint";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes,
encrypt your access keys and store them in the configuration file or environment variables. In
this example, access keys are stored in the environment variables for identity authentication.
Before running the code in this example, configure environment variables ACCESS_KEY_ID and
SECRET_ACCESS_KEY_ID.// Obtain an AK/SK pair on the management console.String ak =
System.getenv("ACCESS_KEY_ID");String sk = System.getenv("SECRET_ACCESS_KEY_ID");String
jksPassword = "you-jks-password";String jksPath = "/path/to/your/keystore/file";KeyStore ks =
KeyStore.getInstance("JKS");char[] passArray = jksPassword.toCharArray();FileInputStream
inputStream = new FileInputStream(jksPath);ks.load(inputStream,
passArray);KeyManagerFactory kmf =
KeyManagerFactory.getInstance(KeyManagerFactory.getDefaultAlgorithm());kmf.init(ks,
passArray);String trustJKSPassword = "you-trustJKS-password";String trustJKSPath = "/path/to/
your/trustKeyStore/file";KeyStore trustKeyStore = KeyStore.getInstance("JKS");char[]
trustPassArray = trustJKSPassword.toCharArray();FileInputStream trustInputStream = new
FileInputStream(trustJKSPath);trustKeyStore.load(trustInputStream,
trustPassArray);TrustManagerFactory tmf =
TrustManagerFactory.getInstance(TrustManagerFactory.getDefaultAlgorithm());tmf.init(trustKeyS
tore);ObsConfiguration config = new
ObsConfiguration();config.setEndPoint(endPoint);config.setKeyManagerFactory(kmf);config.setTr
ustManagerFactory(tmf);ObsClient obsClient = new ObsClient(ak, sk, config);
```

NOTE

The local certificate must be stored as a .jks file. You can run the following command to call the Java keytool to convert a .cer certificate into a .jks one:

```
keytool -import -file your-cer-file.cer -keystore your-keystore-file.jks
```

- **validateCertificate:** By configuring **validateCertificate**, you can use the default trust store to check whether the certificate returned by the server is correct.

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-  
endpoint";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes,  
encrypt your access keys and store them in the configuration file or environment variables. In  
this example, access keys are stored in the environment variables for identity authentication.  
Before running the code in this example, configure environment variables ACCESS_KEY_ID and  
SECRET_ACCESS_KEY_ID./ Obtain an AK/SK pair on the management console.String ak =  
System.getenv("ACCESS_KEY_ID");String sk =  
System.getenv("SECRET_ACCESS_KEY_ID");config.setEndPoint(endPoint);config.setValidateCertific  
ate(true);ObsClient obsClient = new ObsClient(ak, sk,config);
```

- HTTP proxy: After the HTTP proxy is configured, the SDK uses the proxy to access the server.

```
// Enter the endpoint of the region where the bucket is located.String endPoint = "https://your-  
endpoint";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes,  
encrypt your access keys and store them in the configuration file or environment variables. In  
this example, access keys are stored in the environment variables for identity authentication.  
Before running the code in this example, configure environment variables ACCESS_KEY_ID and  
SECRET_ACCESS_KEY_ID./ Obtain an AK/SK pair on the management console.String ak =  
System.getenv("ACCESS_KEY_ID");String sk = System.getenv("SECRET_ACCESS_KEY_ID");// The  
URL cannot contain the protocol header http:// or https://.String proxyUrl = "proxy.com";int  
proxyPort = 8080;String proxyUser = "userName";String proxyPassword =  
"password";ObsConfiguration config = new  
ObsConfiguration();config.setEndPoint(endPoint);config.setHttpProxy(proxyUrl, proxyPort,  
proxyUser, proxyPassword);ObsClient obsClient = new ObsClient(ak, sk,config);
```

Configuring a User-Defined Domain Name to Access OBS

```
// Specify the user-defined domain name that has been configured on the console.String endPoint = "http://  
your-domain";// Hard-coded or plaintext access keys (AK/SK) are risky. For security purposes, encrypt your  
access keys and store them in the configuration file or environment variables. In this example, access keys  
are stored in the environment variables for identity authentication. Before running the code in this example,  
configure environment variables ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID./ Obtain an AK/SK pair on  
the management console.String ak = System.getenv("ACCESS_KEY_ID");String sk =  
System.getenv("SECRET_ACCESS_KEY_ID");// Create an instance of the ObsConfiguration  
class.ObsConfiguration config = new  
ObsConfiguration();config.setEndPoint(endPoint);config.setCname(true);// Create an ObsClient  
instance.ObsClient obsClient = new ObsClient(ak, sk, config);// Create an instance of ObsClient using  
Provider./ ObsClient obsClient = new ObsClient(new EnvironmentVariableObsCredentialsProvider(),  
config);// ObsClient obsClient = new ObsClient(new EcsObsCredentialsProvider(), config);// Use the  
instance to access OBS./ Close the instance.obsClient.close();
```

ObsConfiguration Parameters

NOTE

- If the network bandwidth is sufficient, you can tune the **socketWriteBufferSize**, **socketReadBufferSize**, **readBufferSize**, and **writeBufferSize** parameters to improve upload and download performance.
- If the network condition is poor, you are advised to increase the values of **connectionTimeout** and **socketTimeout**.

Table 6-3 ObsConfiguration parameters

Parameter	Description	Method	Recommended Value
connectionTimeout	Explanation: The amount of time to wait when initially establishing an HTTP/HTTPS connection before giving up and timing out. Default value: 60000 , in milliseconds.	ObsConfiguration.setConnectionTimeout	[10000, 60000]
socketTimeout	Explanation: The amount of time to wait for data to be transferred by the socket layer before the transfer times out. Default value: 60000 , in milliseconds.	ObsConfiguration.setSocketTimeout	[10000, 60000]
idleConnectionTime	Explanation: If the idle time of a connection exceeds the value of this parameter, the connection is closed. To ensure your services run smoothly, you are advised to set this parameter to a value less than or equal to 29000 ms. Default value: 28000 , in ms.	ObsConfiguration.setIdleConnectionTime	[0, 29000]
maxIdleConnections	Explanation: The allowed maximum number of pooled idle connections. Default value: 1000	ObsConfiguration.setMaxIdleConnections	Configure this parameter based on your needs.
maxConnections	Explanation: The allowed maximum number of concurrent HTTP requests. Default value: 1000	ObsConfiguration.setMaxConnections	Default

Parameter	Description	Method	Recommended Value
maxErrorRetry	Explanation: The allowed maximum number of retry attempts for failed requests (for example, request exceptions, or 500 or 503 error responses from the server). Restrictions: This parameter is invalid if an exception occurs when the data stream of the object to upload or download is being processed. Default value: 3	ObsConfiguration.setMaxErrorRetry	[0, 5]
endPoint	Explanation: OBS server address. It consists of a protocol type, domain name, and port number, for example, https://your-endpoint:443. For security purposes, you are advised to use HTTPS. Restrictions: For DNS resolution and OBS reliability reasons, you are not allowed to specify an IP address for this parameter because this will lead to a path-style request being used, which is prohibited by OBS. You need to use a domain name to specify this parameter. Default value: None	ObsConfiguration.setEndPoint	Configure this parameter based on your needs.
httpProxy	Explanation: HTTP proxy configuration. Default value: This parameter is left blank by default.	ObsConfiguration.setHttpProxy	Configure this parameter based on your needs.

Parameter	Description	Method	Recommended Value
validateCertificate	Explanation: Whether to verify the server certificate. Value range: • true: The server certificate is verified. • false: The server certificate is not verified. Default value: false	ObsConfiguration.setValidateCertificate	Configure this parameter based on your needs.
verifyResponseContentType	Explanation: Whether to verify the ContentType header in the response. Value range: • true: The ContentType header is verified. • false: The ContentType header is not verified. Default value: true	ObsConfiguration.setVerifyResponseContentType	Default
readBufferSize	Explanation: Size of the buffer used for downloading objects from socket streams. The value -1 indicates that no buffer is configured. Default value: -1, in bytes.	ObsConfiguration.setReadBufferSize	Configure this parameter based on your needs.
writeBufferSize	Explanation: Size of the buffer used for uploading objects to socket streams. The value -1 indicates that no buffer is configured. Default value: -1, in bytes.	ObsConfiguration.setWriteBufferSize	Configure this parameter based on your needs.

Parameter	Description	Method	Recommended Value
socketWriteBufferSize	Explanation: Size of the sending buffer of the socket. This parameter corresponds to java.net.SocketOptions.SO_SNDBUF. Default value: -1, in bytes, indicating that this parameter is not configured.	ObsConfiguration.setSocketWriteBufferSize	Default
socketReadBufferSize	Explanation: Size of the receiving buffer of the socket. This parameter corresponds to java.net.SocketOptions.SO_RCVBUF. Default value: -1, in bytes, indicating that this parameter is not configured.	ObsConfiguration.setSocketReadBufferSize	Default
keyManagerFactory	Explanation: Factory used for generating javax.net.ssl.KeyManager. Default value: This parameter is left blank by default.	ObsConfiguration.setKeyManagerFactory	Configure this parameter based on your needs.
trustManagerFactory	Explanation: Factory used for generating javax.net.ssl.TrustManager. Default value: This parameter is left blank by default.	ObsConfiguration.setTrustManagerFactory	Configure this parameter based on your needs.

Parameter	Description	Method	Recommended Value
isStrictHostnameVerification	Explanation: Whether to strictly verify the server-side host name. If this parameter is set to true, javax.net.ssl.HttpsURLConnection.setDefaultHostnameVerifier needs to be used to create an object as an implementation of javax.net.ssl.HostnameVerifier to verify the host name. Value range: • true: The host name of the server is strictly verified. • false: The host name of the server is not strictly verified. Default value: false	ObsConfiguration.setIsStrictHostnameVerification	Configure this parameter based on your needs.
keepAlive	Explanation: Whether to use persistent connections to access OBS. Value range: • true: Persistent connections are used. • false: Persistent connections are not used. Default value: true	ObsConfiguration.setKeepAlive	Configure this parameter based on your needs.

Parameter	Description	Method	Recommended Value
cname	Explanation: Whether to use a user-defined domain name to access OBS. NOTE To protect your services from being affected by the takeover of Huawei Cloud public domain names by relevant organizations, you are advised to use a user-defined domain name to access a bucket. Value range: true: A user-defined domain name is used. false: A user-defined domain name is not used. Default value: false	ObsConfiguration.setCname	Configure this parameter based on your needs.
sslProvider	Explanation: Provider of the SSLContext. Default value: The SSLContext provided by the JDK is used.	ObsConfiguration.setSslProvider	Configure this parameter based on your needs.
httpProtocolType	Explanation: HTTP protocol type used for accessing the OBS server. Default value: HTTP1.1 NOTE If the value of endPoint does not contain any protocol, HTTPS is used by default.	ObsConfiguration.setHttpProtocolType	Configure this parameter based on your needs.
httpDispatcher	Explanation: User-defined dispatcher. Default value: None	ObsConfiguration.setHttpDispatcher	Configure this parameter based on your needs.

Parameter	Description	Method	Recommended Value
secureRandom	Explanation: User-defined random number generator. Default value: new SecureRandom() NOTE - On some platforms, the implementation of new SecureRandom() may be insecure. For security purposes, you are advised to use ObsConfiguration.setSecureRandom() to get a SecureRandom instance from a true random seed. - If an operating system does not have sufficient entropy to generate random numbers (for example, when the system is just started), SecureRandom that generates true random numbers may be blocked until there is enough entropy available. To address this issue, you can take measures (such as using haveged in Linux) to supplement entropy.	ObsConfiguration.setSecureRandom	Configure this parameter based on your needs.

6.3 Configuring SDK Logging (SDK for Java)

OBS SDK for Java offers logging based on the open-source Apache Log4j 2 library. By default, the SDK stores WARN log files to the directory specified by the JDK system variable **user.dir**. You can modify the log configuration file based on your needs.

Procedure

- Step 1** Obtain the file **log4j2.xml** from the OBS SDK for Java package.
- Step 2** Modify the log level and storage path in the file as required.
- Step 3** Save the file to the **classpath** root directory, or call **Log4j2Configurator.setLogConfig** to specify the storage path of the file.

----End

 NOTE

- For details about SDK logging, see [Log Analysis \(SDK for Java\)](#).
- You can modify the **log4j2.xml** file to configure access permissions for log files.

7 Bucket Management (SDK for Java)

7.1 Creating a Bucket (SDK for Java)

Function

This API creates an OBS bucket. Buckets are containers for storing objects (files uploaded to OBS) in OBS.

When creating a bucket, you can also configure parameters such as the storage class, region, and access control list (ACL) as needed.

Restrictions

- To create a bucket, you must have the **obs:bucket:CreateBucket** permission. IAM is recommended for granting permissions. For details, see .
- A maximum of 100 buckets and parallel file systems in total can be created globally for an account. There is no limit on the number or size of objects stored in a bucket.
- The created bucket name must be unique in OBS. If you repeatedly create buckets with the same name in the same region, an HTTP status code **200** will be returned. In other cases, creating a bucket with an existing bucket name will have an HTTP status code **409** returned, indicating that such a bucket already exists.
- The name of a deleted bucket can be reused for another bucket or a parallel file system at least 30 minutes after the deletion.

Method

- Method 1: Create a bucket with a specified name.
`createBucket(String bucketName)`
- Method 2: Create a bucket with a specified name in a specified region.
`createBucket(String bucketName, String location)`
- Method 3: Create a bucket with a specified name in a specified region.
`createBucket(ObsBucket bucket)`

- Method 4: Create a bucket and configure access permissions, storage class, location, cluster type, bucket type (object bucket or parallel file system), enterprise project ID, and extension permissions for the bucket.
createBucket([CreateBucketRequest request](#))

Request Parameters

Table 7-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	CreateBucketRequest	Yes	Explanation: Request parameters for creating a bucket. For details, see Table 7-2 .

Table 7-2 CreateBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
location	String	Yes if the region where the OBS service resides is not the default region	Explanation: Region where a bucket will be located

Parameter	Type	Mandatory (Yes/No)	Description
acl	AccessControlList	No	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 7-3 for the available options. - To use a user-defined ACL, see Table 7-5 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE
storageClasses	StorageClassEnum	No	Explanation: Bucket storage class that can be specified at bucket creation. Value range: See Table 7-12. Default value: STANDARD
extensionPermissionMap	Map< ExtensionBucketPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionBucketPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 7-4. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
epid	String	No	Explanation: Enterprise project ID that can be specified at bucket creation. If you have enabled Enterprise Project Management Service (EPS), you can obtain the project ID from the EPS console. Restrictions: The value of epid is a Universally Unique Identifier (UUID). epid is not required if you have not enabled EPS yet. Example: 9892d768-2d13-450f-aac7-ed0e44c2585f Default value: None

Table 7-3 Pre-defined ACL

Constant	Description
AccessControllist.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.

Constant	Description
AccessControllist.REST_CAN_NED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 7-4 ExtensionBucketPermissionEnum

Constant	Description
GRANT_READ	Grants the READ permission to an account ID. The account with the READ permission can list objects, multipart uploads, and object versions in the bucket you are creating, and can obtain bucket metadata.

Constant	Description
GRANT_WRITE	Grants the WRITE permission to an account ID. The account with the WRITE permission can create, delete, and overwrite objects in the bucket you are creating, and can initiate or abort multipart uploads, as well as upload, copy, and assemble parts.
GRANT_READ_ACP	Grants the READ_ACP permission to an account ID. The account with the READ_ACP permission can read the ACL of the bucket you are creating.
GRANT_WRITE_ACP	Grants the WRITE_ACP permission to an account ID. The account with the WRITE_ACP permission can modify the ACL of the bucket you are creating.
GRANT_FULL_CONTROL	Grants the FULL_CONTROL permission to an account ID. The account with the FULL_CONTROL permission can perform any operation on the bucket you are creating.
GRANT_READ_DELIVERED	Grants the READ permission to an account ID. By default, this READ permission applies to all objects in the bucket.
GRANT_FULL_CONTROL_DELIVERED	Grants the FULL_CONTROL permission to an account ID. By default, this FULL_CONTROL permission applies to all objects in the bucket.

Table 7-5 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Description
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 7-6 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 7-7 .

Table 7-6 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-7 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 7-8 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: For details, see Table 7-11 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 7-8 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 7-9 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: For details, see Table 7-10 . Default value: None

Table 7-9 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-10 GroupGrantee

Constant	Description
ALL_USERS	All users.

Constant	Description
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 7-11 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACL	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACL	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 7-12 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class.
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Responses

Table 7-13 ObsBucket

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
owner	Owner	Explanation: Bucket owner information. For details, see Table 7-6.

Parameter	Type	Description
creationDate	java.util.Date	Explanation: Time when the bucket was created. Default value: None
location	String	Explanation: Region where a bucket will be located
storageClass	StorageClass Enum	Explanation: Bucket storage class that can be specified at bucket creation. Value range: See Table 7-12 . Default value: STANDARD
acl	AccessControlList	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: • To use a pre-defined ACL, see Table 7-3 for the available options. • To use a user-defined ACL, see Table 7-5 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE

Code Example: Creating a Bucket with Complex Configurations

This example creates a bucket and configures its ACL, storage class, and region.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.AvailableZoneEnum;
import com.obs.services.model.CreateBucketRequest;
import com.obs.services.model.ObsBucket;
import com.obs.services.model.StorageClassEnum;

public class CreateBucket001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
```

```
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways.
// String securityToken = System.getenv("SECURITY_TOKEN");

// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    CreateBucketRequest request = new CreateBucketRequest();
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Example bucket location
    request.setBucketName(exampleBucket);
    // Set the bucket ACL to Private (the default value).
    request.setAcl(AccessControlList.REST_CANNED_PRIVATE);
    // Set the bucket storage class to Standard.
    request.setBucketStorageClass(StorageClassEnum.STANDARD);
    request.setLocation(exampleLocation);
    // Create a bucket.
    ObsBucket bucket = obsClient.createBucket(request);
    // The bucket is created.
    System.out.println("CreateBucket successfully");
    System.out.println("RequestId:" + bucket.getRequestId());

} catch (ObsException e) {
    System.out.println("CreateBucket failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message: " + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("CreateBucket failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Creating a Bucket with Simple Configurations

This example creates a bucket and specifies its name and endpoint.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsBucket;

public class CreateBucket002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
```

```
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Create a bucket.
    ObsBucket bucket = obsClient.createBucket(exampleBucket);
    System.out.println("CreateBucket successfully");
    System.out.println("StatusCode: " + bucket.getStatusCode());
    System.out.println("RequestId: " + bucket.getRequestId());
} catch (ObsException e) {
    System.out.println("CreateBucket failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code: " + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message: " + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID: " + e.getErrorRequestId());
    System.out.println("Host ID: " + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("CreateBucket failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.2 Obtaining a Bucket List (SDK for Java)

Function

OBS buckets are containers for storing objects you upload to OBS. This API returns a list of all buckets that meet the specified conditions in all regions of the current account. Returned buckets are listed in alphabetical order by bucket name.

Restrictions

- To obtain a bucket list, you must have the **obs:bucket:ListAllMyBuckets** permission.
- To query the bucket location when listing buckets, you need to set **ListBucketsRequest.setQueryLocation** to **true**.

Method

obsClient.listBuckets([ListBucketsRequest request](#))

Request Parameters

Table 7-14 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	ListBucketsRequest	Yes	Explanation: Request parameters for listing buckets. For details, see Table 7-15 .

Table 7-15 ListBucketsRequest

Parameter	Type	Mandatory (Yes/No)	Description
queryLocation	boolean	No	Explanation: Whether to query the bucket location Value range: true: The bucket location is queried. false: The bucket location is not queried. Default value: false

Responses

Table 7-16 ListBucketsResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
owner	Owner	Explanation: Bucket owner information. For details, see Table 7-17 . Default value: None
buckets	List< ObsBucket >	Explanation: Bucket list information. For details, see Table 7-26 . Default value: None

Table 7-17 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-18 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class.
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Table 7-19 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Description
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 7-17 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 7-20 .

Table 7-20 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 7-21 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 7-24 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 7-21 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 7-22 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 7-23 . Default value: None

Table 7-22 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-23 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 7-24 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 7-25 Pre-defined ACL

Constant	Description
AccessControllist.REST_CAN_NED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CAN_NED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN_NED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 7-26 ObsBucket

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
owner	Owner	Explanation: Bucket owner information. For details, see Table 7-17.
creationDate	java.util.Date	Explanation: Time when the bucket is created. Default value: None
location	String	Explanation: Region where a bucket is located

Parameter	Type	Description
storageClass	StorageClass Enum	Explanation: Bucket storage class that can be specified at bucket creation. Value range: See Table 7-18 . Default value: STANDARD
acl	AccessControlList	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: • To use a pre-defined ACL, see Table 7-25 for the available options. • To use a user-defined ACL, see Table 7-19 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE

Code Examples

This example lists all buckets and queries the bucket regions.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListBucketsRequest;
import com.obs.services.model.ObsBucket;
import java.util.List;
public class ListBucket001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
```

```
try {
    // List buckets.
    ListBucketsRequest request = new ListBucketsRequest();
    request.setQueryLocation(true);
    List<ObsBucket> buckets = obsClient.listBuckets(request);
    for (ObsBucket bucket : buckets) {
        System.out.println("ListBuckets successfully");
        System.out.println("BucketName:" + bucket.getBucketName());
        System.out.println("CreationDate:" + bucket.getCreationDate());
        System.out.println("Location:" + bucket.getLocation());
    }
} catch (ObsException e) {
    System.out.println("ListBuckets failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("ListBuckets failed");
    // Print other error information.
    e.printStackTrace();
}
```

7.3 Deleting a Bucket (SDK for Java)

Function

This API deletes an empty bucket. You can delete buckets you no longer use to free up space. The name of a deleted bucket can be reused for another bucket at least 30 minutes after the deletion.

Restrictions

- Only empty buckets can be deleted. An empty bucket means that:
 - The bucket does not contain any object (including noncurrent versions and delete markers).
 - The bucket does not contain any fragments, which means that there are no multipart uploads that have not been completed in the bucket.
- To delete a bucket, you must be the bucket owner or have the required permission (**obs:bucket:DeleteBucket** in IAM or **DeleteBucket** in a bucket policy).
- Only empty buckets (without objects and part fragments) can be deleted.
- Bucket deletion is a non-idempotence operation and an error will be reported if the bucket you want to delete does not exist.

Method

obsClient.deleteBucket(String [bucketName](#))

Request Parameters

Table 7-27 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-28 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HeaderResponse;
public class DeleteBucket001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
```

```
// Example bucket name
String exampleBucket = "examplebucket";
// Delete the bucket.
HeaderResponse response = obsClient.deleteBucket(exampleBucket);
System.out.println("DeleteBucket successfully");
System.out.println("StatusCode:"+response.getStatusCode());
System.out.println("RequestId:"+response.getRequestId());
} catch (ObsException e) {
    System.out.println("DeleteBucket failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message: " + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("DeleteBucket failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.4 Checking Whether a Bucket Exists (SDK for Java)

Function

This API checks whether a bucket exists. If an HTTP status code **200** is returned, the bucket exists. If **404** is returned, the bucket does not exist.

Restrictions

- To check whether a bucket exists, you must be the bucket owner or have the required permission (**obs:bucket:HeadBucket** in IAM or **HeadBucket** in a bucket policy).

Method

obsClient.headBucket(String **bucketName**)

Request Parameters

Table 7-29 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-30 List of returned results

Type	Description
boolean	Explanation: Whether the bucket exists Value range: true: The bucket exists. false: The bucket does not exist. Default value: None

Code Examples

This example checks whether bucket **examplebucket** exists.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class HeadBucket {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            // Check whether the bucket exists.
            boolean exists = obsClient.headBucket(exampleBucket);
            System.out.println("HeadBucket successfully");
            System.out.println(exists);
        } catch (ObsException e) {
            System.out.println("HeadBucket failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code :" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
        }
    }
}
```

```
        System.out.println("Host ID:" + e.getErrorHostId());
    } catch (Exception e) {
        System.out.println("HeadBucket failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

7.5 Obtaining Bucket Metadata (SDK for Java)

Function

This API returns information about a bucket, including the storage class, region, CORS rules, and redundancy policy.

Restrictions

- To obtain bucket metadata, you must be the bucket owner or have the required permission (**obs:bucket:HeadBucket** in IAM or **HeadBucket** in a bucket policy).
- For details about values for methods such as **BucketMetadataInfoResult.getAllowMethods**, see the [CORS](#) configurations of the bucket.

Method

obsClient.getBucketMetadata([BucketMetadataInfoRequest request](#))

Request Parameters

Table 7-31 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BucketMetadataInfoRequest	Yes	Explanation: Request parameters for obtaining bucket metadata. For details, see Table 7-32 .

Table 7-32 BucketMetadataInfoRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
origin	String	No	Explanation: Origin of the cross-domain request specified by the preflight request. Generally, it is a domain name. Restrictions: You can configure one or more rules and use at most one wildcard character (*) in each rule. If you want to configure multiple rules, separate them using a line breaker. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
requestHeaders	List<String>	No	Explanation: HTTP headers in a cross-origin request. Only CORS requests matching the allowed headers are valid. Restrictions: You can enter multiple allowed headers, with one separated from another using a line break. Each header can contain one wildcard character (*) at most. Spaces, ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None

Responses

Table 7-33 BucketMetadataInfoResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
location	String	Explanation: Region where a bucket is located.

Parameter	Type	Description
obsVersion	String	Explanation: OBS version of the bucket. Value range: 3.0 indicates the latest OBS version. -- indicates any version earlier than 3.0. Default value: None
storageClass	StorageClassEnum	Explanation: Bucket storage class that can be specified at bucket creation. Value range: See Table 7-34. Default value: STANDARD
allowOrigin	String	Explanation: If Origin in the request meets the CORS rules of the bucket, AllowedOrigin specified in the CORS rules is returned. Restrictions: You can configure one or more rules and use at most one wildcard character (*) in each rule. If you want to configure multiple rules, separate them using a line breaker. Default value: None
allowHeaders	List<String>	Explanation: If RequestHeader in the request meets the CORS rules of the bucket, AllowedHeader specified in the CORS rules is returned. Restrictions: You can enter multiple allowed headers, with one separated from another using a line break. Each header can contain one wildcard character (*) at most. Spaces, ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None

Parameter	Type	Description
allowMethods	List<String>	Explanation: The allowed HTTP methods for a cross-origin request, indicating the operation types of buckets and objects. Value range: The following HTTP methods are supported: • GET • PUT • HEAD • POST • DELETE Default value: None
exposeHeaders	List<String>	Explanation: The CORS-allowed additional headers in the response. These headers provide additional information to clients. By default, your browser can only access headers Content-Length and Content-Type. If your browser needs to access other headers, add them to the list of the allowed additional headers. Restrictions: Spaces, wildcard characters (*), ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None
maxAge	int	Explanation: Time your client can cache the response for a cross-origin request Restrictions: Each CORS rule can contain only one maxAge. Value range: An integer greater than 0, in seconds Default value: 100

Parameter	Type	Description
epid	String	Explanation: Enterprise project ID that can be specified at bucket creation. If you have enabled EPS, you can obtain the project ID from the EPS console. Restrictions: The value of epid is a UUID. If the default enterprise project is used, 0 is passed here or epid is not included. epid is not required for those who have not enabled the EPS. Example: 9892d768-2d13-450f-aac7-ed0e44c2585f Default value: None

Table 7-34 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class.
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Code Examples

This example obtains the metadata of bucket **examplebucket**. The origin of the cross-origin request is **http://www.exampleorigin.com**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketMetadataInfoRequest;
import com.obs.services.model.BucketMetadataInfoResult;
public class GetBucketMetadata001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
```

```
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Example origin
    String exampleOrigin = "http://www.exampleorigin.com";
    BucketMetadataInfoRequest request = new BucketMetadataInfoRequest(exampleBucket);
    // Specify setOrigin only when cross origin resource sharing (CORS) is used.
    request.setOrigin(exampleOrigin);
    // Obtain the bucket metadata.
    BucketMetadataInfoResult result = obsClient.getBucketMetadata(request);
    System.out.println("GetBucketMetadata successfully");
    System.out.println("GetBucketType:" + result.getBucketType());
    System.out.println("GetLocation:" + result.getLocation());
    System.out.println("GetBucketStorageClass:" + result.getBucketStorageClass());
    System.out.println("GetObsVersion:" + result.getObsVersion());
    System.out.println("GetAllowOrigin:" + result.getAllowOrigin());
    System.out.println("GetMaxAge:" + result.getMaxAge());
    System.out.println("GetAllowHeaders:" + result.getAllowHeaders());
    System.out.println("GetAllowMethods:" + result.getAllowMethods());
    System.out.println("GetExposeHeaders:" + result.getExposeHeaders());
} catch (ObsException e) {
    System.out.println("GetBucketMetadata failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("GetBucketMetadata failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.6 Setting a Bucket ACL (SDK for Java)

Function

Access control lists (ACLs) allow resource owners to grant other accounts the permissions to access resources. By default, only the resource owner has full control over resources when a bucket or object is created. That is, the bucket creator has full control over the bucket, and the object uploader has full control over the object. Other accounts do not have the permissions to access resources. If resource owners want to grant other accounts the read and write permissions on resources, they can use ACLs. ACLs grant permissions to accounts. After an account is granted permissions, both the account and its IAM users can access the resources.

You can configure a bucket ACL by referring to [Method](#).

Restrictions

- A bucket can have up to 100 ACL rules.
- To configure an ACL for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketAcl** in IAM or **PutBucketAcl** in a bucket policy).

Method

- Method 1: Set a pre-defined ACL when creating a bucket.

```
obsBucket.setBucketName(exampleBucket);  
// Set the bucket ACL to private read and write.  
obsBucket.setAcl(AccessControlList.REST_CANNED_PRIVATE);  
// Create a bucket.  
obsClient.createBucket(obsBucket);
```
- Method 2: After the bucket is created, set a pre-defined ACL.

```
// Set the bucket ACL to private read and write.  
obsClient.setBucketAcl(String exampleBucket, AccessControlList.REST\_CANNED\_PRIVATE);
```
- Method 3: After the bucket is created, set a user-defined ACL.

```
// Set a user-defined ACL for the bucket.  
obsClient.setBucketAcl(String bucketName, AccessControlList acl);
```

Request Parameters

Table 7-35 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
acl	AccessControlList	Yes	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 7-37 for the available options. - To use a user-defined ACL, see Table 7-36 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE

Table 7-36 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	Yes	Explanation: Bucket owner information. For details, see Table 7-38 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Parameter	Type	Mandatory (Yes/No)	Type
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 7-39 .

Table 7-37 ACL

Constant	Description
AccessControlList.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControlList.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.

Constant	Description
AccessControlList.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Table 7-38 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-39 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 7-40 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 7-43 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 7-40 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 7-41 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 7-42 . Default value: None

Table 7-41 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-42 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 7-43 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.

Constant	Default Value	Description
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Responses

Table 7-44 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Example: Setting a Pre-defined ACL When Creating a Bucket

This example configures a pre-defined ACL during the creation of bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.ObsBucket;
public class SetBucketAcl001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            ObsBucket obsBucket = new ObsBucket();
            // Example bucket name
            String exampleBucket = "examplebucket";
            obsBucket.setBucketName(exampleBucket);
            // Set the bucket ACL to private read and write.
            obsBucket.setAcl(AccessControlList.REST_CANNED_PRIVATE);
            // Create a bucket.
            obsClient.createBucket(obsBucket);
            System.out.println("SetBucketAcl successfully");
        } catch (ObsException e) {
            System.out.println("SetBucketAcl failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
        }
```



```
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("SetBucketAcl failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

Code Example: Setting a Pre-defined ACL for an Existing Bucket

This example configures a pre-defined ACL for bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
public class SetBucketAcl002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            // Set the bucket ACL to private read and write.
            obsClient.setBucketAcl(exampleBucket, AccessControlList.REST_CANNED_PRIVATE);
            System.out.println("SetBucketAcl successfully");
        } catch (ObsException e) {
            System.out.println("SetBucketAcl failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("SetBucketAcl failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

Code Example: Setting a User-defined ACL for an Existing Bucket

This example defines an ACL for the **exampleBucket** bucket.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.CanonicalGrantee;
import com.obs.services.model.GroupGrantee;
import com.obs.services.model.Owner;
import com.obs.services.model.Permission;

public class SetBucketAcl003 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            // Example user ID
            String exampleUserId = "userid";
            // Example user ID
            String exampleOwnerId = "ownerid";
            AccessControlList acl = new AccessControlList();
            Owner owner = new Owner();
            owner.setId(exampleOwnerId);
            acl.setOwner(owner);
            // Grant the full control permission to a specified user.
            acl.grantPermission(new CanonicalGrantee(exampleUserId),
                Permission.PERMISSION_FULL_CONTROL);
            // Grant the read permission to all users.
            acl.grantPermission(GroupGrantee.ALL_USERS, Permission.PERMISSION_READ);
            // Set the bucket ACL.
            obsClient.setBucketAcl(exampleBucket, acl);
            System.out.println("SetBucketAcl successfully");
        } catch (ObsException e) {
            System.out.println("SetBucketAcl failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("SetBucketAcl failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

```
    }  
  }  
}
```

This example directly configures an ACL for bucket **exampleBucket** and applies this ACL to the objects in the bucket.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.AccessControlList;  
import com.obs.services.model.CanonicalGrantee;  
import com.obs.services.model.GroupGrantee;  
import com.obs.services.model.Owner;  
import com.obs.services.model.Permission;  
public class SetBucketAcl004 {  
    public static void main(String[] args) {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using  
        // hard coding may result in leakage.  
        // Obtain an AK/SK pair on the management console.  
        String ak = System.getenv("ACCESS_KEY_ID");  
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");  
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are  
        // advised not to use hard coding, which may result in information leakage.  
        // Obtain an AK/SK pair and a security token using environment variables or import them in other  
        // ways.  
        // String securityToken = System.getenv("SECURITY_TOKEN");  
        // Enter the endpoint corresponding to the region where the bucket is located.  
        String endPoint = "https://your-endpoint";  
        // Obtain an endpoint using environment variables or import it in other ways.  
        //String endPoint = System.getenv("ENDPOINT");  
  
        // Create an instance of ObsClient.  
        // Use a permanent AK/SK pair to initialize the client.  
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
        // Use a temporary AK/SK pair and security token to initialize the client.  
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);  
  
        try {  
            //Example bucket name  
            String exampleBucket = "examplebucket";  
            //Example user ID  
            String exampleUserid = "userid";  
            //Example owner ID  
            String exampleOwnerId = "ownerid";  
            AccessControlList acl = new AccessControlList();  
            Owner owner = new Owner();  
            owner.setId(exampleOwnerId);  
            acl.setOwner(owner);  
            // Grant the full control permission to a specific user. The bucket ACL also applies to the objects in  
            // the bucket.  
            acl.grantPermission(new CanonicalGrantee(exampleUserid),  
            Permission.PERMISSION_FULL_CONTROL,true);  
            // Grant the read permission to all users. The bucket ACL also applies to the objects in the bucket.  
            acl.grantPermission(GroupGrantee.ALL_USERS, Permission.PERMISSION_READ,true);  
            // Set the bucket ACL.  
            obsClient.setBucketAcl(exampleBucket, acl);  
            System.out.println("SetBucketAcl successfully");  
        } catch (ObsException e) {  
            System.out.println("SetBucketAcl failed");  
            // Request failed. Print the HTTP status code.  
            System.out.println("HTTP Code:" + e.getResponseCode());  
            // Request failed. Print the server-side error code.  
            System.out.println("Error Code:" + e.getErrorCode());  
            // Request failed. Print the error details.  
            System.out.println("Error Message:" + e.getErrorMessage());  
            // Request failed. Print the request ID.  
            System.out.println("Request ID:" + e.getErrorRequestId());  
            System.out.println("Host ID:" + e.getErrorHostId());  
            e.printStackTrace();  
        } catch (Exception e) {
```

```
        System.out.println("SetBucketAcl failed");  
        // Print other error details.  
        e.printStackTrace();  
    }  
}
```

7.7 Obtaining a Bucket ACL (SDK for Java)

Function

Access control lists (ACLs) allow resource owners to grant other accounts the permissions to access resources. By default, only the resource owner has full control over resources when a bucket or object is created. That is, the bucket creator has full control over the bucket, and the object uploader has full control over the object. Other accounts do not have the permissions to access resources. If resource owners want to grant other accounts the read and write permissions on resources, they can use ACLs. ACLs grant permissions to accounts. After an account is granted permissions, both the account and its IAM users can access the resources.

This API returns the ACL of a bucket.

Restrictions

- To obtain the ACL of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketAcl** in IAM or **GetBucketAcl** in a bucket policy).

Method

obsClient.getBucketAcl(String [bucketName](#))

Request Parameters

Table 7-45 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-46 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	Yes	Explanation: Bucket owner information. For details, see Table 7-47 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 7-48 .

Table 7-47 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-48 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 7-49 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 7-52 . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 7-49 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 7-50 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 7-51 . Default value: None

Table 7-50 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 7-51 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 7-52 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Code Examples

This example obtains the ACL of the **examplebucket** bucket.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.AccessControlList;  
public class GetBucketAcl001 {
```

```
public static void main(String[] args) {
    // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
    // hard coding may result in leakage.
    // Obtain an AK/SK pair on the management console.
    String ak = System.getenv("ACCESS_KEY_ID");
    String sk = System.getenv("SECRET_ACCESS_KEY_ID");
    // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
    // advised not to use hard coding, which may result in information leakage.
    // Obtain an AK/SK pair and a security token using environment variables or import them in other
    // ways.
    // String securityToken = System.getenv("SECURITY_TOKEN");
    // Enter the endpoint corresponding to the region where the bucket is to be created.
    String endPoint = "https://your-endpoint";
    // Obtain an endpoint using environment variables or import it in other ways.
    //String endPoint = System.getenv("ENDPOINT");

    // Create an ObsClient instance.
    // Use the permanent AK/SK pair to initialize the client.
    ObsClient obsClient = new ObsClient(ak, sk, endPoint);
    // Use the temporary AK/SK pair and security token to initialize the client.
    // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

    try {
        // Example bucket name
        String exampleBucket = "examplebucket";
        //Obtain the bucket ACL.
        AccessControlList acl = obsClient.getBucketAcl(exampleBucket);
        System.out.println("GetBucketAcl successfully");
        System.out.println(acl);
    } catch (ObsException e) {
        System.out.println("GetBucketAcl failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("GetBucketAcl failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

7.8 Configuring a Bucket Policy (SDK for Java)

Function

OBS provides access control over buckets. You can use an access policy to define whether a user can perform certain operations on a specific bucket. OBS access control can be implemented using IAM permissions, bucket policies, and ACLs.

A bucket policy applies to both the bucket and objects therein. You can use a bucket policy to grant permissions for a bucket and objects therein to IAM users or other accounts. If you want IAM users to have different permissions for different buckets, you need to configure different bucket policies for those users.

Besides bucket ACLs, bucket owners can use bucket policies to centrally control access to buckets and objects in buckets.

You can call **ObsClient.setBucketPolicy** to configure a bucket policy.

For more information, see "Bucket Policies and Object Policies" in the *User Guide (Ally Region)*.

Restrictions

- To configure a bucket policy, you must be the bucket owner or the bucket owner's IAM user with the required permission (**obs:bucket:PutBucketPolicy** in IAM or **PutBucketPolicy** in a bucket policy).
- The latest policy configurations will overwrite the preceding configurations when you call the API for configuring a bucket policy. For example, if you have configured bucket policies A, B, C, and D and you want to add a new bucket policy E, you must add policy E to the file containing the existing four policies and then upload the file with all policies contained. Likewise, if you want to delete bucket policy D, you must remove it from the file containing policies A, B, C, and D, and then upload the file without policy D contained.
- For details about the bucket policy format (JSON), see the Object Storage Service API Reference.
- OBS returns **404 NoSuchBucketPolicy** when you call this API in the following scenarios:
 - The specified bucket policy does not exist.
 - The standard policy of the specified bucket is set to **Private** and no advanced policies are configured.

Method

obsClient.setBucketPolicy(String **bucketName**, String **policy**)

Request Parameters

Table 7-53 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.). - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
policy	String	Yes	Explanation: Policy information in JSON format Restrictions: The bucket name contained in the Resource parameter of the policy must be the same as the one specified for the current bucket policy. Default value: None

Responses

Table 7-54 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures a policy for bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class SetBucketPolicy001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
```

```
String exampleBucket = "examplebucket";
// Example bucket policy
String examplePolicy =
    "{\"Statement\": [{\"Principal\": \"*\", \"Effect\": \"Allow\", \"Action\": \"ListBucket\", \"Resource\": \"s3://\" + exampleBucket + \"/*\"}]}";
obsClient.setBucketPolicy(exampleBucket, examplePolicy);
System.out.println("SetBucketAcl successfully");
} catch (ObsException e) {
    System.out.println("SetBucketAcl failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("SetBucketAcl failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.9 Obtaining the Policy of a Bucket (SDK for Java)

Function

OBS provides access control over buckets. You can use an access policy to define whether a user can perform certain operations on a specific bucket. OBS access control can be implemented using IAM permissions, bucket policies, and ACLs. For more information, see .

This API returns the policy of a bucket.

Restrictions

- OBS returns **404 NoSuchBucketPolicy** when you call this API in the following scenarios:
 - The specified bucket policy does not exist.
 - The standard policy of the specified bucket is set to **Private** and no advanced policies are configured.
- To obtain the policy of a bucket, you must be the bucket owner or the bucket owner's IAM user with the required permission (**obs:bucket:GetBucketPolicy** in IAM or **GetBucketPolicy** in a bucket policy).

Method

obsClient.getBucketPolicy(String **bucketName**)

Request Parameters

Table 7-55 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-56 List of returned results

Parameter	Type	Description
policy	String	Explanation: Policy information in JSON format Restrictions: The bucket name contained in the Resource parameter of the policy must be the same as the one specified for the current bucket policy. Default value: None

Code Examples

This example returns the policy of bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class GetBucketPolicy001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            String policy = obsClient.getBucketPolicy(exampleBucket);
            System.out.println("\t" + policy);
        } catch (ObsException e) {
            System.out.println("GetBucketAcl failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
        }
    }
}
```

```
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("GetBucketAcl failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.10 Deleting a Bucket Policy (SDK for Java)

Function

OBS provides access control over buckets. You can use an access policy to define whether a user can perform certain operations on a specific bucket. OBS access control can be implemented using IAM permissions, bucket policies, and ACLs. For more information, see .

You can call **ObsClient.deleteBucketPolicy** to delete the policy of a bucket. OBS returns **204 No Content** if the deletion is successful or the requested bucket policy does not exist.

Restrictions

- To delete the policy of a bucket, you must be the bucket owner or the bucket owner's IAM user with the required permission (**obs:bucket:DeleteBucketPolicy** in IAM or **DeleteBucketPolicy** in a bucket policy).

Method

obsClient.deleteBucketPolicy(String **bucketName**)

Request Parameters

Table 7-57 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-58 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes the policy of bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class DeleteBucketPolicy001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
```

```
String exampleBucket = "examplebucket";
obsClient.deleteBucketPolicy(exampleBucket);
} catch (ObsException e) {
    System.out.println("DeleteBucketAcl failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("DeleteBucketAcl failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.11 Obtaining the Region of a Bucket (SDK for Java)

Function

This API returns the region where a bucket is located.

Restrictions

- To obtain the region of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketLocation** in IAM or **GetBucketLocation** in a bucket policy).
- When creating a bucket, you can specify its location. For details, see [Creating a Bucket](#).

Method

obsClient.getBucketLocation(String [bucketName](#))

Request Parameters

Table 7-59 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-60 BucketLocationResponse

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
location	String	Explanation: Region where a bucket is located.

Code Examples

This example returns the region where bucket **exampleBucket** locates.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class GetBucketLocation001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");
```

```
// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Obtain the bucket region.
    String location = obsClient.getBucketLocation(exampleBucket);
    System.out.println("GetBucketLocation successfully");
    System.out.println("Location:" + location);
} catch (ObsException e) {
    System.out.println("GetBucketLocation failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("GetBucketLocation failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.12 Obtaining Storage Information of a Bucket (SDK for Java)

Function

This API returns the storage information of a bucket, including the number of objects and the space occupied by the objects in the bucket.

NOTE

OBS measures bucket storage statistics in the background and does not update the storage information in real time. So, you are advised not to perform real-time verification on the storage information.

Restrictions

- To obtain the storage information of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketStorageInfo** in IAM or **GetBucketStorage** in a bucket policy).

Method

obsClient.getBucketStorageInfo(String **bucketName**)

Request Parameters

Table 7-61 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (.), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-62 BucketStorageInfo

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
size	long	Explanation: Size of the space occupied by objects in the bucket. Value range: A non-negative integer, in bytes. Default value: None
objectNum	long	Explanation: Number of objects in the bucket. Default value: None

Code Examples

This example returns the storage information of bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketStorageInfo;
public class GetBucketStorageInfo001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
```

```
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Obtain the bucket storage information.
    BucketStorageInfo storageInfo = obsClient.getBucketStorageInfo(exampleBucket);
    System.out.println("GetBucketStorageInfo successfully");
    System.out.println("GetObjectNumber:" + storageInfo.getObjectNumber());
    System.out.println("GetStorageSize:" + storageInfo.getSize() + " bytes");
} catch (ObsException e) {
    System.out.println("GetBucketStorageInfo failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("GetBucketStorageInfo failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.13 Configuring a Storage Quota (SDK for Java)

Function

A quota limits the maximum capacity allowed in a bucket. By default, there is no limit on the storage capacity of the entire OBS system or a single bucket, and any number of objects can be stored. You can set a storage quota to control the total size of objects that can be uploaded to the bucket. After the storage quota has been reached, object upload will fail.

A quota limit does not apply to the objects uploaded before the quota is configured. If the specified quota is already smaller than the total size of existing objects in the bucket, the existing objects in the bucket will not be deleted, but no more object can be uploaded to the bucket later. In this case, to upload new objects, you must delete some of the existing objects to make the used space below the quota limit.

Restrictions

- A bucket quota must be a non-negative integer in bytes.
- OBS does not provide an API for deleting bucket storage quotas. You can set the bucket storage quota to **0** to cancel the limit.
- To configure a storage quota for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketQuota** in IAM or **PutBucketQuota** in a bucket policy).

Method

obsClient.setBucketQuota(String **bucketName**, **BucketQuota bucketQuota**)

Request Parameters

Table 7-63 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.). - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
bucketQuota	BucketQuota	Yes	Explanation: Bucket quota information Restrictions: See Table 7-64.

Table 7-64 BucketQuota

Parameter	Type	Mandatory (Yes/No)	Description
bucketQuota	long	Yes	Explanation: Bucket quota. Value range: A non-negative integer, in bytes. Default value: 0 , indicating that there is no limit on the bucket quota.

Responses

Table 7-65 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures a quota for bucket **exampleBucket**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.BucketQuota;
```

```
public class SetBucketQuota001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            // Example bucket quota
            long exampleBucketQuota = 1024 * 1024 * 100L;
            // Set the bucket quota to 100 MB.
            BucketQuota quota = new BucketQuota(exampleBucketQuota);
            obsClient.setBucketQuota(exampleBucket, quota);
            System.out.println("SetBucketQuota successfully");
        } catch (ObsException e) {
            System.out.println("SetBucketQuota failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("SetBucketQuota failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

7.14 Obtaining a Bucket Storage Quota (SDK for Java)

Function

This API returns the storage quota (upper limit of the storage capacity) of a bucket. If the quota is 0, there is no upper limit on the bucket capacity.

Restrictions

- A bucket quota must be a non-negative integer in bytes.
- A frozen bucket owner (due to account in arrears) is not allowed to query the bucket storage quota.

- To obtain the storage quota of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketQuota** in IAM or **GetBucketQuota** in a bucket policy).

Method

```
obsClient.getBucketQuota(String bucketName)
```

Request Parameters

Table 7-66 List of request parameters

Parameter	Type	Mandator y (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-67 BucketQuota

Parameter	Type	Mandatory (Yes/No)	Description
bucketQuota	long	Yes	Explanation: Bucket quota. Value range: A non-negative integer, in bytes. Default value: 0 , indicating that there is no limit on the bucket quota.

Code Examples

This example returns the quota of bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketQuota;
public class GetBucketQuota001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            // Obtain the bucket quota.
            BucketQuota quota = obsClient.getBucketQuota(exampleBucket);
            System.out.println("GetBucketQuota successfully");
            System.out.println("GetBucketQuota:" + quota.getBucketQuota() + " bytes");
        } catch (ObsException e) {
            System.out.println("GetBucketQuota failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
```

```
System.out.println("Error Code:" + e.getErrorCode());  
// Request failed. Print the error details.  
System.out.println("Error Message:" + e.getErrorMessage());  
// Request failed. Print the request ID.  
System.out.println("Request ID:" + e.getErrorRequestId());  
System.out.println("Host ID:" + e.getErrorHostId());  
e.printStackTrace();  
} catch (Exception e) {  
    System.out.println("GetBucketQuota failed");  
    // Print other error information.  
    e.printStackTrace();  
}  
}
```

7.15 Configuring a Storage Class for a Bucket (SDK for Java)

Function

This API configures a storage class for a bucket. If you do not specify a storage class when uploading or copying an object, or initiating a multipart upload, the object inherits the bucket's storage class.

The following table lists the three available storage classes.

Storage Class	Description	Value in OBS Java SDK
Standard	Features low access latency and high throughput and is used for storing massive, frequently accessed (multiple times a month) or small objects (< 1 MB) requiring quick response.	StorageClassEnum.STANDARD

For more information, see "Storage Classes" in the *User Guide (Ally Region)*.

Restrictions

- To configure a storage class for a bucket, you must be the bucket owner or have the required permission (**obs:PutBucketStoragePolicy** in IAM or **PutBucketStoragePolicy** in a bucket policy).
- The bucket storage class is independent from the storage classes of objects in the bucket. When uploading an object, if you do not specify a storage class for the object, the object will use the same storage class that the bucket uses. However, after the object is uploaded, its storage class will not change as that of the bucket changes. Likewise, the storage class of the bucket will also not change if the storage class of any objects therein changes.

Method

```
obsClient.setBucketStoragePolicy(String bucketName,  
    BucketStoragePolicyConfiguration bucketStorage)
```

Request Parameters

Table 7-68 Request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value:

Parameter	Type	Mandatory (Yes/No)	Description
			None
bucketStorage	BucketStoragePolicyConfiguration (Inheriting HeaderResponse)	Yes	Explanation: Storage class of the bucket. Value range: See Table 7-69 . Default value: None

Table 7-69 BucketStoragePolicyConfiguration

Parameter	Type	Description
storageClass	StorageClassEnum	Explanation: Storage class of the bucket. Value range: See Table 7-70 . Default value: None

Table 7-70 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class.
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Responses

Table 7-71 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures a storage class for bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.AvailableZoneEnum;
import com.obs.services.model.CreateBucketRequest;
import com.obs.services.model.ObsBucket;
import com.obs.services.model.StorageClassEnum;

public class CreateBucket001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");

        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
```

```
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    CreateBucketRequest request = new CreateBucketRequest();
    // Example bucket name
    String exampleBucket = "examplebucket";
    // Example bucket location
    request.setBucketName(exampleBucket);
    // Set the bucket ACL to Private (the default value).
    request.setAcl(AccessControlList.REST_CANNED_PRIVATE);
    // Set the bucket storage class to Standard.
    request.setBucketStorageClass(StorageClassEnum.STANDARD);
    request.setLocation(exampleLocation);
    // Create a bucket.
    ObsBucket bucket = obsClient.createBucket(request);
    // The bucket is created.
    System.out.println("CreateBucket successfully");
    System.out.println("RequestId:" + bucket.getRequestId());

} catch (ObsException e) {
    System.out.println("CreateBucket failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message: " + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("CreateBucket failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.16 Obtaining the Storage Class of a Bucket (SDK for Java)

Function

This API returns the storage class of a bucket.

Restrictions

- To obtain a bucket's storage class, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketStoragePolicy** in IAM or **GetBucketStoragePolicy** in a bucket policy).

Method

obsClient.getBucketStoragePolicy(String **bucketName**)

Request Parameters

Table 7-72 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-73 BucketStoragePolicyConfiguration

Parameter	Type	Description
storageClass	StorageClassEnum	Explanation: Storage class of the bucket. For details, see Table 7-74 .

Table 7-74 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class.
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Code Examples

This example configures a storage class for bucket **exampleBucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketStoragePolicyConfiguration;
public class GetBucketStoragePolicy001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Example bucket name
            String exampleBucket = "examplebucket";
            // Obtain the bucket's storage class.
            BucketStoragePolicyConfiguration storagePolicy =
            obsClient.getBucketStoragePolicy(exampleBucket);
```



```
        System.out.println("GetBucketStoragePolicy successfully");
        System.out.println("GetBucketStorageClass:" + storagePolicy.getBucketStorageClass());
    } catch (ObsException e) {
        System.out.println("GetBucketStoragePolicy failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("GetBucketStoragePolicy failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

7.17 Bucket Inventory (SDK for Java)

7.17.1 Configuring an Inventory Rule (SDK for Java)

Function

This API configures an inventory rule for the bucket. With this API, object information in a bucket (source) is regularly listed and saved as CSV files. These files are then stored in a specified bucket (destination). In this manner, you can easily manage objects in a bucket. A source bucket can also be the destination bucket. A bucket inventory file contains the following object-related information: object versions, sizes, storage classes, and last modification time.

Restrictions

- To configure an inventory rule for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketInventoryConfiguration** in IAM or **PutBucketInventoryConfiguration** in a bucket policy).
- **Bucket versions**
 - Bucket inventories can be generated only for OBS 3.0 buckets, but they can be stored in either OBS 3.0 or OBS 2.0.
- **Number of bucket inventory rules**
 - A bucket can have a maximum of 10 inventory rules.
- **Source and destination buckets**
 - The source bucket (for which a bucket inventory rule is configured) and the destination bucket (where the generated inventory files are stored) must belong to the same account.
 - The source bucket and the destination bucket must be in the same region.
- **Functions**

- Inventory files must be in the CSV format.
- OBS can generate inventory files for all objects in a bucket or a group of objects whose names begin with the same prefix.
- If a bucket has multiple inventory rules, the filtering criteria in these rules must not overlap.
 - If a bucket already has an inventory rule for the entire bucket, new inventory rules that filter objects by prefixes cannot be created. If you need an inventory rule that covers only a subset of objects in the bucket, first delete the inventory rule configured for the entire bucket.
 - If an inventory rule that filters objects by a specified prefix already exists, you cannot create an inventory rule for the entire bucket. To create an inventory rule for the entire bucket, make sure that the bucket has no inventory rules that filter objects by specified prefixes.
 - If a bucket already has an inventory rule that filters objects by the object name prefix, such as **ab**, the filter of a new inventory rule cannot be any prefix that contains **a**, **b**, or **ab**. If you still need to create such rules, delete the inventory rule that uses the prefix **ab** first.
- Bucket inventory files only support SSE-KMS encryption.
- **Permissions**
 - Inventory files are uploaded to the destination bucket by an OBS system user. Therefore, you need to grant the system user the permission to write objects to the destination bucket.
- **Others**
 - The bucket inventory function is offered for free, but inventory files are billed for the storage they use.

Method

obsClient.setInventoryConfiguration([SetInventoryConfigurationRequest request](#))

Request Parameters

Table 7-75 SetInventoryConfigurationRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
inventoryConfiguration	InventoryConfiguration	Yes	Explanation: Inventory configurations for a bucket. For details, see Table 7-76.

Table 7-76 InventoryConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
configurationId	String	Yes	Explanation: ID of a bucket inventory rule. Restrictions: The rule ID allows letters (a–z, A–Z), digits (0–9), hyphens (-), underscores (_), and periods (.). Value range: The value can be up to 64 characters long. Default value: None
isEnabled	boolean	Yes	Explanation: Whether the bucket inventory rule is enabled. Value range: true: The rule is enabled, and an inventory file is generated. false: The rule is disabled. No inventory file is generated. Default value: true
objectPrefix	String	No	Explanation: Used to filter objects. Only objects with the specified name prefix are included in the inventory. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
frequency	String	Yes	Explanation: Intervals when inventories are generated. You can set this parameter to Daily or Weekly. An inventory is generated within one hour from when it is configured for the first time. Then it is generated at the specified intervals. Value range: Daily: Inventories are generated once a day. Weekly: Inventories are generated once a week. Default value: None
inventory Format	String	Yes	Explanation: Inventory file format. Only the CSV format is supported. Value range: CSV Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
destinationBucket	String	Yes	Explanation: Name of the bucket for storing inventories. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
inventoryPrefix	String	No	Explanation: The prefix of the inventory file name. Value range: The value must contain 1 to 1,024 characters. Default value: If you do not specify this parameter, BucketInventory is used as the prefix by default.

Parameter	Type	Mandatory (Yes/No)	Description
includedObjectVersions	String	Yes	Explanation: Whether versions of objects are included in an inventory. Value range: - If this parameter is set to All, all the versions of objects are included in the inventory, and version-related fields are added to the inventory, including: VersionId, IsLatest, and DeleteMarker. - If this parameter is set to Current, the inventory only lists information about the current object version and does not include any version-related fields. Default value: None
optionalFields	ArrayList<String>	No	Explanation: Additional object metadata fields that are contained in an inventory file. Value range: Size: Object size. LastModifiedDate: Last time when the object was modified. StorageClass: The storage class of the object. ETag: The ETag value of the object. IsMultipartUploaded: Whether the object was uploaded in a multipart upload. ReplicationStatus: The cross-region replication status of the object. EncryptionStatus: The encryption status of the object. Default value: None

Responses

Table 7-77 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures an inventory for bucket **example-bucket** to list objects with the name prefix **exampleObjectPrefix**. The inventory files are stored in bucket **example-target-bucket** and the name prefix of the inventory files is **exampleInventoryPrefix**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HeaderResponse;
import com.obs.services.model.inventory.InventoryConfiguration;
import com.obs.services.model.inventory.SetInventoryConfigurationRequest;
public class SetInventoryConfiguration001
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");
```



```
// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the following parameters.
    String exampleBucketName = "example-bucket";
    String exampleTargetBucketName = "example-target-bucket";
    String exampleConfigurationId = "exampleConfigId001";
    String exampleInventoryPrefix = "exampleInventoryPrefix";
    String exampleObjectPrefix = "exampleObjectPrefix";
    // Set parameters for the inventory rule.
    InventoryConfiguration exampleConfiguration = new InventoryConfiguration();
    exampleConfiguration.setDestinationBucket(exampleTargetBucketName);
    exampleConfiguration.setConfigurationId(exampleConfigurationId);
    exampleConfiguration.setInventoryFormat(InventoryConfiguration.InventoryFormatOptions.CSV);
    exampleConfiguration.setFrequency(InventoryConfiguration.FrequencyOptions.DAILY);
    exampleConfiguration.setEnabled(true);
    exampleConfiguration.setIncludedObjectVersions(
        InventoryConfiguration.IncludedObjectVersionsOptions.CURRENT);
    exampleConfiguration.setInventoryPrefix(exampleInventoryPrefix);
    exampleConfiguration.setObjectPrefix(exampleObjectPrefix);
    // Set additional metadata fields that will be contained in the inventory file.
    exampleConfiguration
        .getOptionalFields()
        .add(InventoryConfiguration.OptionalFieldOptions.IS_MULTIPART_UPLOADED);
    exampleConfiguration.getOptionalFields().add(InventoryConfiguration.OptionalFieldOptions.ETAG);
    exampleConfiguration
        .getOptionalFields()
        .add(InventoryConfiguration.OptionalFieldOptions.REPLICATION_STATUS);
    SetInventoryConfigurationRequest request =
        new SetInventoryConfigurationRequest(exampleBucketName, exampleConfiguration);
    // Set the inventory rule.
    HeaderResponse response = obsClient.setInventoryConfiguration(request);
    System.out.println("SetInventoryConfiguration succeeded");
    System.out.println("HTTP Code: " + response.getStatusCode());
} catch (ObsException e) {
    System.out.println("SetInventoryConfiguration failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("SetInventoryConfiguration failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.17.2 Obtaining an Inventory Rule (SDK for Java)

Function

This API returns an inventory rule (identified by **configurationId**) for the bucket.

Restrictions

- To obtain a bucket inventory rule, you must be the bucket owner or have the required permission (**obs:bucket:GetInventoryConfiguration** in IAM or **GetInventoryConfiguration** in a bucket policy).

Method

obsClient.getInventoryConfiguration([GetInventoryConfigurationRequest request](#))

Request Parameters

Table 7-78 GetInventoryConfigurationRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.). - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
configurationId	String	Yes	Explanation: ID of the inventory rule. Restrictions: The ID can be up to 64 bytes long and allows letters (a-z, A-Z), digits (0-9), hyphens (-), underscores (_), and periods (.). Default value: None

Responses

Table 7-79 GetInventoryConfigurationResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Parameter	Type	Description
inventoryConfiguration	InventoryConfiguration	Explanation: Bucket inventory configurations. For details, see Table 7-80 .

Table 7-80 InventoryConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
configurationId	String	Yes	Explanation: ID of a bucket inventory rule. Restrictions: The rule ID allows letters (a-z, A-Z), digits (0-9), hyphens (-), underscores (_), and periods (.). Value range: The value can be up to 64 characters long. Default value: None
isEnabled	boolean	Yes	Explanation: Whether the bucket inventory rule is enabled. Value range: true: The rule is enabled, and an inventory file is generated. false: The rule is disabled. No inventory file is generated. Default value: true
objectPrefix	String	No	Explanation: Used to filter objects. Only objects with the specified name prefix are included in the inventory. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
frequency	String	Yes	Explanation: Intervals when inventories are generated. You can set this parameter to Daily or Weekly. An inventory is generated within one hour from when it is configured for the first time. Then it is generated at the specified intervals. Value range: Daily: Inventories are generated once a day. Weekly: Inventories are generated once a week. Default value: None
inventory Format	String	Yes	Explanation: Inventory file format. Only the CSV format is supported. Value range: CSV Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
destinationBucket	String	Yes	Explanation: Name of the bucket for storing inventories. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
inventoryPrefix	String	No	Explanation: The prefix of the inventory file name. Value range: The value must contain 1 to 1,024 characters. Default value: If you do not specify this parameter, BucketInventory is used as the prefix by default.

Parameter	Type	Mandatory (Yes/No)	Description
includedObjectVersions	String	Yes	Explanation: Whether versions of objects are included in an inventory. Value range: - If this parameter is set to All, all the versions of objects are included in the inventory, and version-related fields are added to the inventory, including: VersionId, IsLatest, and DeleteMarker. - If this parameter is set to Current, the inventory only lists information about the current object version and does not include any version-related fields. Default value: None
optionalFields	ArrayList<String>	No	Explanation: Additional object metadata fields that are contained in an inventory file. Value range: Size: Object size. LastModifiedDate: Last time when the object was modified. StorageClass: The storage class of the object. ETag: The ETag value of the object. IsMultipartUploaded: Whether the object was uploaded in a multipart upload . ReplicationStatus: The cross-region replication status of the object. EncryptionStatus: The encryption status of the object. Default value: None

Code Examples

This example returns the inventory rule whose ID is **exampleConfigId001** for bucket **example-bucket**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.inventory.GetInventoryConfigurationRequest;
import com.obs.services.model.inventory.GetInventoryConfigurationResult;
import com.obs.services.model.inventory.InventoryConfiguration;
public class GetInventoryConfiguration001
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the following parameters.
            String exampleBucketName = "example-bucket";
            String exampleConfigurationId = "exampleConfigId001";
            GetInventoryConfigurationRequest request =
                new GetInventoryConfigurationRequest(exampleBucketName, exampleConfigurationId);
            // Obtain the inventory rule.
            GetInventoryConfigurationResult result = obsClient.getInventoryConfiguration(request);
            InventoryConfiguration inventoryConfiguration = result.getInventoryConfiguration();
            // Print all parameters in the inventory rule.
            System.out.println("ConfigurationId:" + inventoryConfiguration.getConfigurationId());
            System.out.println("DestinationBucket:" + inventoryConfiguration.getDestinationBucket());
            System.out.println("InventoryFormat:" + inventoryConfiguration.getInventoryFormat());
            System.out.println("Enabled:" + inventoryConfiguration.getEnabled());
            System.out.println("Frequency:" + inventoryConfiguration.getFrequency());
            System.out.println("IncludedObjectVersions:" + inventoryConfiguration.getIncludedObjectVersions());
            System.out.println("InventoryPrefix:" + inventoryConfiguration.getInventoryPrefix());
            System.out.println("ObjectPrefix:" + inventoryConfiguration.getObjectPrefix());
            System.out.println("OptionalFields:" + inventoryConfiguration.getOptionalFields());
            // Print the HTTP status code.
            System.out.println("HTTP Code: " + result.getStatusCode());
            System.out.println("GetInventoryConfiguration succeeded");
        } catch (ObsException e) {
            System.out.println("GetInventoryConfiguration failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code: " + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
        } catch (Exception e) {
            System.out.println("GetInventoryConfiguration failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```


7.17.3 Listing Inventory Rules (SDK for Java)

Function

This API returns a list of all inventory rules for the bucket.

Restrictions

- To list inventory rules for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetInventoryConfiguration** in IAM or **GetInventoryConfiguration** in a bucket policy).

Method

obsClient.listInventoryConfiguration([ListInventoryConfigurationRequest request](#))

Request Parameters

Table 7-81 ListInventoryConfigurationRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 7-82 ListInventoryConfigurationResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
inventoryConfigurations	List< InventoryConfiguration >	Explanation: All inventory configurations of the bucket. For details, see Table 7-83 .

Table 7-83 InventoryConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
configurationId	String	Yes	Explanation: ID of a bucket inventory rule. Restrictions: The rule ID allows letters (a–z, A–Z), digits (0–9), hyphens (-), underscores (_), and periods (.). Value range: The value can be up to 64 characters long. Default value: None
isEnabled	boolean	Yes	Explanation: Whether the bucket inventory rule is enabled. Value range: true: The rule is enabled, and an inventory file is generated. false: The rule is disabled. No inventory file is generated. Default value: true
objectPrefix	String	No	Explanation: Used to filter objects. Only objects with the specified name prefix are included in the inventory. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
frequency	String	Yes	Explanation: Intervals when inventories are generated. You can set this parameter to Daily or Weekly. An inventory is generated within one hour from when it is configured for the first time. Then it is generated at the specified intervals. Value range: Daily: Inventories are generated once a day. Weekly: Inventories are generated once a week. Default value: None
inventory Format	String	Yes	Explanation: Inventory file format. Only the CSV format is supported. Value range: CSV Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
destinationBucket	String	Yes	Explanation: Name of the bucket for storing inventories. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
inventoryPrefix	String	No	Explanation: The prefix of the inventory file name. Value range: The value must contain 1 to 1,024 characters. Default value: If you do not specify this parameter, BucketInventory is used as the prefix by default.

Parameter	Type	Mandatory (Yes/No)	Description
includedObjectVersions	String	Yes	Explanation: Whether versions of objects are included in an inventory. Value range: - If this parameter is set to All, all the versions of objects are included in the inventory, and version-related fields are added to the inventory, including: VersionId, IsLatest, and DeleteMarker. - If this parameter is set to Current, the inventory only lists information about the current object version and does not include any version-related fields. Default value: None
optionalFields	ArrayList<String>	No	Explanation: Additional object metadata fields that are contained in an inventory file. Value range: Size: Object size. LastModifiedDate: Last time when the object was modified. StorageClass: The storage class of the object. ETag: The ETag value of the object. IsMultipartUploaded: Whether the object was uploaded in a multipart upload . ReplicationStatus: The cross-region replication status of the object. EncryptionStatus: The encryption status of the object. Default value: None

Code Examples

This example lists all inventory files of bucket **example-bucket**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.inventory.InventoryConfiguration;
import com.obs.services.model.inventory.ListInventoryConfigurationRequest;
import com.obs.services.model.inventory.ListInventoryConfigurationResult;
import java.util.List;
public class ListInventoryConfigurations001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the following parameters.
            String exampleBucketName = "example-bucket";
            ListInventoryConfigurationRequest request = new
ListInventoryConfigurationRequest(exampleBucketName);
            // List all inventory rules.
            ListInventoryConfigurationResult result = obsClient.listInventoryConfiguration(request);
            List<InventoryConfiguration> inventoryConfigurations = result.getInventoryConfigurations();
            //Print all parameters in all inventory rules.
            for (InventoryConfiguration inventoryConfiguration : inventoryConfigurations) {
                System.out.println("ConfigurationId:" + inventoryConfiguration.getConfigurationId());
                System.out.println("DestinationBucket:" + inventoryConfiguration.getDestinationBucket());
                System.out.println("InventoryFormat:" + inventoryConfiguration.getInventoryFormat());
                System.out.println("Enabled:" + inventoryConfiguration.getEnabled());
                System.out.println("Frequency:" + inventoryConfiguration.getFrequency());
                System.out.println("IncludedObjectVersions:" +
inventoryConfiguration.getIncludedObjectVersions());
                System.out.println("InventoryPrefix:" + inventoryConfiguration.getInventoryPrefix());
                System.out.println("ObjectPrefix:" + inventoryConfiguration.getObjectPrefix());
                System.out.println("OptionalFields:" + inventoryConfiguration.getOptionalFields());
                System.out.println("-----");
            }
            // Print the HTTP status code.
            System.out.println("HTTP Code: " + result.getStatusCode());
            System.out.println("ListInventoryConfiguration succeeded");
        } catch (ObsException e) {
            System.out.println("ListInventoryConfiguration failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code: " + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
        } catch (Exception e) {
            System.out.println("ListInventoryConfiguration failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```



```
}  
}
```

7.17.4 Deleting an Inventory Rule (SDK for Java)

Function

This API deletes an inventory rule (identified by **configurationId**) from the bucket.

Restrictions

- To delete a bucket inventory rule, you must be the bucket owner or have the required permission (**obs:bucket:DeleteBucketInventoryConfiguration** in IAM or **DeleteBucketInventoryConfiguration** in a bucket policy).

Method

obsClient.deleteInventoryConfiguration([DeleteInventoryConfigurationRequest request](#))

Request Parameters

Table 7-84 DeleteInventoryConfigurationRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.) - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
configurationId	String	Yes	Explanation: ID of the inventory rule to be deleted. The ID can be up to 64 bytes long and allows letters (a-z, A-Z), digits (0-9), hyphens (-), underscores (_), and periods (.) Default value: None

Responses

Table 7-85 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes the inventory rule whose ID is **exampleConfigId001** for bucket **example-bucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HeaderResponse;
import com.obs.services.model.inventory.DeleteInventoryConfigurationRequest;
import com.obs.services.model.inventory.GetInventoryConfigurationRequest;
import com.obs.services.model.inventory.GetInventoryConfigurationResult;
import com.obs.services.model.inventory.InventoryConfiguration;
public class DeleteInventoryConfiguration001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
```

```
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the following parameters.
    String exampleBucketName = "example-bucket";
    String exampleConfigurationId = "exampleConfigId001";
    DeleteInventoryConfigurationRequest request =
        new DeleteInventoryConfigurationRequest(exampleBucketName, exampleConfigurationId);
    // Delete the inventory rule.
    HeaderResponse result = obsClient.deleteInventoryConfiguration(request);
    // Print the HTTP status code.
    System.out.println("HTTP Code: " + result.getStatusCode());
    System.out.println("DeleteInventoryConfiguration succeeded");
} catch (ObsException e) {
    System.out.println("DeleteInventoryConfiguration failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("DeleteInventoryConfiguration failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.18 Configuring a Custom Domain Name for a Bucket (SDK for Java)

Function

This API sets a custom domain name for an OBS bucket.

Restrictions

- To use this API, you must be the bucket owner or have the required permission (**obs:bucket:SetBucketCustomDomain** in IAM or **SetBucketCustomDomain** in a bucket policy).
-

Method

`ObsClient.setBucketCustomDomain(SetBucketCustomDomainRequest request)`

Request Parameters

Table 7-86 SetBucketCustomDomainRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets with the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Value range: None Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
domainName	String	Yes	Explanation: Custom domain name. Restrictions: None Value range: None Default value: None
customDomainCertificateConfig	CustomDomainCertificateConfig	No	Explanation: Custom domain name certificate configuration. Restrictions: Only HTTPS requests are supported. Value range: CustomDomainCertificateConfig Default value: None

Table 7-87 CustomDomainCertificateConfig

Parameter	Type	Mandatory (Yes/No)	Description
name	String	Yes	Explanation: Certificate name Restrictions: None Value range: The value can contain 3 to 63 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
certificateId	String	No	Explanation: Certificate ID on the CCM. Restrictions: None Value range: The value must contain 16 characters. Default value: None
certificate	String	Yes	Explanation: Certificate content, which can contain intermediate and root certificates. If certificateChain contains a certificate chain, only the certificate itself will be extracted from certificate. Use escape characters \n or \r\n to replace line breaks. Restrictions: None Value range: None Default value: None
certificateChain	String	No	Explanation: Certificate chain, which is optional and can be passed through the certificate parameter. Use escape characters \n or \r\n to replace line breaks. Restrictions: None Value range: None Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
privateKey	String	Yes	Explanation: Private key of a certificate. An encrypted private key cannot be uploaded. Use escape characters <code>\n</code> or <code>\r\n</code> to replace line breaks. Restrictions: None Value range: None Default value: None

Responses

Type	Description
HeaderResponse	Explanation: SDK common results

Table 7-88 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be <code>2xx</code> (indicating successes) or <code>4xx</code> or <code>5xx</code> (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures a custom domain name for a bucket.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CustomDomainCertificateConfig;
import com.obs.services.model.HeaderResponse;
import com.obs.services.model.SetBucketCustomDomainRequest;

import java.util.Map;

public class SetBucketCustomDomain {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        String securityToken = System.getenv("SECURITY_TOKEN");

        // Enter the endpoint corresponding to the bucket. CN North-Beijing4 is used here as an example.
        // Replace it with the one currently in use.
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT_S");
        String endPoint = "https://obs.cn-north-4.myhuaweicloud.com";

        // Create an ObsClient instance.
        ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
        try {
            // Example bucket name
            String exampleBucket = "example-bucket";
            // Example custom domain name
            String exampleDomainName = "example.domain.name.com";
            // Example name of the certificate configured for the custom domain name
            String exampleDomainCertificateName = "exampleCertName";
            // Example ID of the certificate configured for the custom domain name
            String exampleDomainCertificateId = "1234512345123450";
            // Example custom domain name certificate
            String exampleDomainCertificate = "exampleDomainCertificate";
            // Example certificate chain of the custom domain name
            String exampleDomainCertificateChain = "exampleDomainCertificateChain";
            // Example private key of the custom domain name certificate
            String exampleDomainCertificatePrivateKey = "exampleDomainCertificatePrivateKey";
            CustomDomainCertificateConfig customDomainCertificateConfig = new
CustomDomainCertificateConfig();
            customDomainCertificateConfig.setName(exampleDomainCertificateName);
            customDomainCertificateConfig.setCertificateId(exampleDomainCertificateId);
```

```
customDomainCertificateConfig.setCertificate(exampleDomainCertificate);
customDomainCertificateConfig.setCertificateChain(exampleDomainCertificateChain);
customDomainCertificateConfig.setPrivateKey(exampleDomainCertificatePrivateKey);
SetBucketCustomDomainRequest setBucketCustomDomainRequest =
    new SetBucketCustomDomainRequest(exampleBucket, exampleDomainName,
customDomainCertificateConfig);
// Configure the custom domain name for the bucket.
HeaderResponse response = obsClient.setBucketCustomDomain(setBucketCustomDomainRequest);
System.out.println("StatusCode:" + response.getStatusCode());
System.out.println("RequestId:" + response.getRequestId());
System.out.println("SetBucketCustomDomain successfully");
} catch (ObsException e) {
    System.out.println("SetBucketCustomDomain failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    // Check all Map entries and print all headers associated with the error.
    Map<String, String> headers = e.getResponseHeaders();
    if(headers != null){
        for (Map.Entry<String, String> header : headers.entrySet()) {
            System.out.println(header.getKey()+"."+header.getValue());
        }
    }
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("SetBucketCustomDomain failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.19 Obtaining the Custom Domain Names of a Bucket (SDK for Java)

Function

This API obtains the custom domain names of a bucket.

Restrictions

- To use this API, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketCustomDomain** in IAM or **GetBucketCustomDomain** in a bucket policy).
-

Method

ObsClient.getBucketCustomDomain(GetBucketCustomDomainRequest request)

Request Parameters

Table 7-89 GetBucketCustomDomainRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets with the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Value range: None Default value: None

Responses

Type	Description
BucketCustomDomainInfo	Explanation: Information about the custom domain names of a bucket.

Table 7-90 BucketCustomDomainInfo

Parameter	Type	Description
domains	List< Domains >	Explanation: Custom domain name list.
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits indicating the status of a response. It ranges from 2xx (indicating successes) to 4xx or 5xx (indicating errors). Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Table 7-91 Domains

Parameter	Type	Description
domainName	String	Explanation Custom domain name.

Parameter	Type	Description
createTime	java.util.Date	Explanation: Time when a custom domain name was created.
certificateId	String	Explanation Certificate ID on the CCM.

Code Examples

This example obtains the custom domain name of a bucket.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketCustomDomainInfo;
import com.obs.services.model.GetBucketCustomDomainRequest;

import java.util.Map;

public class GetBucketCustomDomain {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        String securityToken = System.getenv("SECURITY_TOKEN");

        // Enter the endpoint corresponding to the bucket. CN North-Beijing4 is used here as an example.
        // Replace it with the one currently in use.
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");
        String endPoint = "https://obs.cn-north-4.myhuaweicloud.com";

        // Create an ObsClient instance.
        try (ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint)) {
            // Example bucket name
            String exampleBucket = "example-bucket";
            // Obtain the custom domain name of the bucket.
            GetBucketCustomDomainRequest getBucketCustomDomainRequest = new
            GetBucketCustomDomainRequest(exampleBucket);
            BucketCustomDomainInfo bucketCustomDomainInfo =
            obsClient.getBucketCustomDomain(getBucketCustomDomainRequest);

            System.out.println("StatusCode:" + bucketCustomDomainInfo.getStatusCode());
            System.out.println("RequestId:" + bucketCustomDomainInfo.getRequestId());
            for (BucketCustomDomainInfo.Domains domain : bucketCustomDomainInfo.getDomains()) {
                System.out.println("DomainName:" + domain.getDomainName());
                System.out.println("CertificateId:" + domain.getCertificateId());
                System.out.println("CreateTime:" + domain.getCreateTime());
                System.out.println();
            }
            System.out.println("GetBucketCustomDomain successfully");
        } catch (ObsException e) {
            System.out.println("GetBucketCustomDomain failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
        }
    }
}
```

```
// Request failed. Print the server-side error code.
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
// Check all Map entries and print all headers associated with the error.
Map<String, String> headers = e.getResponseHeaders();
if(headers != null){
    for (Map.Entry<String, String> header : headers.entrySet()) {
        System.out.println(header.getKey()+"."+header.getValue());
    }
}
e.printStackTrace();
} catch (Exception e) {
    System.out.println("GetBucketCustomDomain failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

7.20 Deleting the Custom Domain Name of a Bucket (SDK for Java)

Function

If encryption is applied to a bucket, objects uploaded to it will be encrypted based on the specified configuration before being stored in OBS. When you download these objects, OBS decrypts them first before returning them to you.

This API deletes the encryption configuration of a bucket.

Restrictions

- To use this API, you must be the bucket owner or have the required permission (**obs:bucket:PutEncryptionConfiguration** in IAM or **PutEncryptionConfiguration** in a bucket policy).
-

Method

ObsClient.deleteBucketCustomDomain(DeleteBucketCustomDomainRequest request)

Request Parameters

Table 7-92 DeleteBucketCustomDomainRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	str	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets with the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Value range: None Default value: None

Responses

Table 7-93 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes the custom domain name of a bucket.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.DeleteBucketCustomDomainRequest;
import com.obs.services.model.HeaderResponse;

import java.util.Map;

public class DeleteBucketCustomDomain {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        String securityToken = System.getenv("SECURITY_TOKEN");

        // Enter the endpoint corresponding to the bucket. CN North-Beijing4 is used here as an example.
        // Replace it with the one currently in use.
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");
        String endPoint = "https://obs.cn-north-4.myhuaweicloud.com";

        // Create an ObsClient instance.
        try (ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint)) {
```



```
// Example bucket name
String exampleBucket = "example-bucket";
// Example custom domain name
String exampleDomainName = "example.domain.name.com";
DeleteBucketCustomDomainRequest deleteBucketCustomDomainRequest =
    new DeleteBucketCustomDomainRequest(exampleBucket,exampleDomainName);
// Delete the custom domain name of the bucket.
HeaderResponse response =
    obsClient.deleteBucketCustomDomain(deleteBucketCustomDomainRequest);

System.out.println("StatusCode:" + response.getStatusCode());
System.out.println("RequestId:" + response.getRequestId());
System.out.println("DeleteBucketCustomDomain successfully");
} catch (ObsException e) {
    System.out.println("DeleteBucketCustomDomain failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    // Check all Map entries and print all headers associated with the error.
    Map<String, String> headers = e.getResponseHeaders();
    if(headers != null){
        for (Map.Entry<String, String> header : headers.entrySet()) {
            System.out.println(header.getKey()+":"+header.getValue());
        }
    }
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("DeleteBucketCustomDomain failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8 Object Upload (SDK for Java)

8.1 Overview (SDK for Java)

In OBS, objects are basic data units that users can perform operations on. OBS Java SDK provides abundant APIs for object upload in the following methods:

- [Streaming](#)
- [File-based](#)
- [Multipart](#)
- [Appendable](#)
- [Resumable](#)
- [Browser-based](#)

The SDK supports the upload of objects whose size ranges from 0 KB to 5 GB. For streaming upload, appendable upload, and file-based upload, data to be uploaded cannot be larger than 5 GB. If the file is larger than 5 GB, multipart upload (where each part is smaller than 5 GB) is suitable. Browser-based upload allows files to be uploaded through a browser.

If you grant anonymous users the read permission for an object during the upload, anonymous users can access the object through a URL after the upload is complete. The object URL is in the format of **`https://bucket name.domain name/directory level/object name`**. If the object resides in the root directory of the bucket, its URL does not contain directory levels.

8.2 Uploading an Object - Streaming (SDK for Java)

Function

This API uploads a local file to OBS over the Internet. These files can be texts, images, videos, or any other type of files.

You can upload texts, images, videos, or any other types of files smaller than 5 GB. In a streaming upload, **`java.io.InputStream`** is used as the object data source. You can call **`ObsClient.putObject`** to upload data streams to OBS.

Restrictions

- To upload an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- The object size in a single upload ranges from 0 to 5 GB.
- To upload files larger than 5 GB, refer to [Multipart Upload \(SDK for Java\)](#).
- To upload a local file, you are advised to use a file-based upload. For details, see [Uploading an Object - File-Based \(SDK for Java\)](#).

Method

obsClient.putObject([PutObjectRequest request](#))

Request Parameters

Table 8-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	PutObjectRequest	Yes	Explanation: Request parameters for uploading an object. For details, see Table 8-2 .

Table 8-2 PutObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-4. Default value: None
acl	AccessControlList	No	Explanation: ACL that can be pre-defined when an object is created. Refer to Table 8-6 to choose the option you need. Value range: - To use a pre-defined ACL, see Table 8-3 for the available options. - To use a user-defined ACL, see Table 8-6 to configure the required parameters. Default value: None
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-5. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call. Value range: An integer greater than 0, in days Default value: None
progressListener	ProgressListener	No	Explanation: Upload progress. For details, see Table 8-13.

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-3 Pre-defined ACL

Constant	Description
AccessControllist.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-4 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Size of the object. Restrictions: • The object size in a single upload ranges from 0 to 5 GB. • To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: Multipurpose Internet Mail Extensions (MIME) type of the object to be uploaded. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentLanguage	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vqI4K0vnWpog== Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
storageClasses	StorageClassEnum	No	Explanation: Object storage class that can be specified at object creation. If this parameter is not specified, the object inherits the storage class of the bucket where it is to be uploaded by default. Value range: See Table 8-16. Default value: None
websiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map , the String key indicates the name of the user-defined metadata that starts with x-obs-meta- , and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata . For details, see Obtaining Object Metadata (SDK for Java) . Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-5 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.

Constant	Description
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Table 8-6 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-11 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-7 .

Table 8-7 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-8 .

Parameter	Type	Mandatory (Yes/No)	Description
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-12 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-8 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 8-9 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 8-10 . Default value: None

Table 8-9 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-10 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 8-11 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-12 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.

Constant	Default Value	Description
PERMISSION_READ_ACL	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACL	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 8-13 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used to obtain the upload progress. For details, see Table 8-14 . Default value: None

Table 8-14 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-15 . Default value: None

Table 8-15 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.

Method	Return Value Type	Description
getTransferredBytes() ()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-16 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Responses

Table 8-17 PutObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
storageClass	StorageClassEnum	Explanation: Object storage class. If the storage class is Standard, leave this parameter blank. Value range: See Table 8-16. Default value: None
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version ID will be returned. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Code Example 1: Uploading a Byte Array

The following code is used to upload the **Hello OBS** character string to the **examplebucket** bucket and name it **objectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import java.io.ByteArrayInputStream;
public class PutObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
```

```
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk,endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Upload a string and convert it to a byte array.
    String content = "Hello OBS";
    obsClient.putObject("examplebucket", "objectname", new
ByteArrayInputStream(content.getBytes()));
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example 2: Uploading a Network Stream

The following code is used to upload the **http://www.a.com** network stream to the **examplebucket** bucket and name it **objectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import java.io.InputStream;
import java.net.URL;
public class PutObject002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Upload a network stream.
```

```
        InputStream inputStream = new URL("http://www.a.com").openStream();
        obsClient.putObject("examplebucket", "objectname", inputStream);
        System.out.println("putObject successfully");
    } catch (ObsException e) {
        System.out.println("putObject failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("putObject failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

Code Example 3: Uploading a File Stream

The following code is used to upload the **localfile** file stream to the **objectname** object in the **dir** directory of **bucketname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
import java.io.FileInputStream;

public class PutObject003 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Path of the local file to be uploaded, in which the file name must be specified.
            FileInputStream fis = new FileInputStream(new File("localfile"));
            obsClient.putObject("examplebucket", "dir/objectname", fis);
            // Path of the local file to be uploaded, in which the file name must be specified.
            FileInputStream fis2 = new FileInputStream(new File("localfile2"));
            PutObjectRequest request = new PutObjectRequest();
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname2");
            request.setInput(fis2);
            obsClient.putObject(request);
            System.out.println("putObject successfully");
        } catch (ObsException e) {
```



```
        System.out.println("putObject failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("putObject failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

8.3 Uploading an Object - File-Based (SDK for Java)

Function

This API uploads local files to OBS over the Internet. The file to be uploaded can be of any type.

Restrictions

- To upload an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- The object size in a single upload ranges from 0 to 5 GB.
- To upload files larger than 5 GB, **multipart uploads** should be used.

Method

obsClient.putObject(**PutObjectRequest request**)

Request Parameters

Table 8-18 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	PutObjectRequest	Yes	Explanation: Request parameters for uploading an object. For details, see Table 8-19 .

Table 8-19 PutObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-31. Default value: None
acl	AccessControlList	No	Explanation: ACL that can be pre-defined when an object is created. Refer to Table 8-24 to choose the option you need. Value range: - To use a pre-defined ACL, see Table 8-33 for the available options. - To use a user-defined ACL, see Table 8-24 to configure the required parameters. Default value: None
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-23. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call. Value range: An integer greater than 0, in days Default value: None

Parameter	Type	Man dato ry (Yes /No)	Description
needCalculateCRC64	boolean	No	Explanation: Whether to automatically calculate the CRC64 value of the data to be uploaded and submit the CRC64 value to the server for verification. Restrictions: This parameter is not supported for POSIX or SFS objects. This parameter is supported only for file uploads. Value range: true: The SDK calculates CRC64 and submits it to the server for verification. false: CRC64 is not calculated. Default value: false
progressListener	Progress Listener	No	Explanation: Upload progress. For details, see Table 8-20.
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-20 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-21 . Default value: None

Table 8-21 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress status . For details, see Obtaining the Upload Progress (SDK for Java) . Default value: None

Table 8-22 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneous-Speed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-23 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 8-24 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-25 .

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-26 .

Table 8-25 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-26 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-28 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-27 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-27 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 8-28 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 8-29 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 8-30 . Default value: None

Table 8-29 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-30 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 8-31 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Size of the object. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object to be uploaded. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Man dator y (Yes/ No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: • The MD5 value of the object must be Base64 encoded. • If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClass	StorageClassEnum	No	Explanation: Object storage class that can be specified at object creation. If this parameter is not specified, the object inherits the storage class of the bucket where it is to be uploaded by default. Value range: See Table 8-32. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-32 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-33 Pre-defined ACL

Constant	Description
AccessControllist.REST_CAN NED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CAN NED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN NED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN NED_PUBLIC_READ_DELIVER ED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN NED_PUBLIC_READ_WRITE_ DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Responses

Table 8-34 PutObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Parameter	Type	Description
storageClass	StorageClassEnum	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-32. Default value: None
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version ID will be returned. Value range: The value must contain 32 characters. Default value: None
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Code Examples

This example uploads local files to bucket **examplebucket**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Upload files.
            // localfile indicates the path of the local file to be uploaded, in which the file name must be
            // specified.
            PutObjectRequest request = new PutObjectRequest();
            request.setBucketName("examplebucket");
            request.setObjectKey("objectkey");
            request.setFile(new File("localfile"));
            obsClient.putObject(request);
            System.out.println("putObject successfully");
        } catch (ObsException e) {
            System.out.println("putObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("putObject failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

8.4 Obtaining the Upload Progress (SDK for Java)

Function

This API returns the upload progress of a specified object.

You can call **PutObjectRequest.setProgressListener** to obtain the upload progress.

Restrictions

- To obtain the upload progress, you must have the **obs:object:PutObject** permission.
- You can query the upload progress when uploading an object in streaming, file-based, multipart, appendable, or resumable mode.
- If the value of **ProgressStatus.getTransferPercentage()** is **-1**, the content is uploaded in streaming mode. In this case, you must configure the **Content-Length** parameter.

Method

PutObjectRequest.setProgressListener([ProgressListener progressListener](#))

Request parameters

Table 8-35 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
progressListener	ProgressListener	Yes	Explanation: Data transmission listener for obtaining the upload progress. For details, see Table 8-36 .

Table 8-36 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-37 .

Table 8-37 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-38 .

Table 8-38 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneous-Speed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Code Examples

This example returns the progress of uploading local file **exampleLocalFilePath** to bucket **examplebucket** as object **objectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ProgressListener;
import com.obs.services.model.ProgressStatus;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject005 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
```



```
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Upload a file.
    PutObjectRequest request = new PutObjectRequest("examplebucket", "exampleobject");
    request.setFile(new File("exampleLocalFilePath"));
    request.setProgressListener(
        new ProgressListener() {
            @Override
            public void progressChanged(ProgressStatus status) {
                // Obtain the average upload rate.
                System.out.println("AverageSpeed:" + status.getAverageSpeed());
                // Obtain the upload progress in percentage.
                System.out.println("TransferPercentage:" + status.getTransferPercentage());
            }
        }
    );
    // Refresh the upload progress each time 1 MB data is uploaded.
    request.setProgressInterval(1024 * 1024L);
    obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

This example returns the progress of uploading local file **exampleFileName** in a stream to bucket **examplebucket** as object **objectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.ObsConfiguration;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.ProgressListener;
import com.obs.services.model.ProgressStatus;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
import java.io.FileInputStream;

public class PutObjectByInputStreamWithProgress {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
```

```
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the bucket.
// Obtain an endpoint using environment variables or import it in other ways.
// String endPoint = System.getenv("ENDPOINT");
String endPoint = "https://obs.region.example.com";
ObsConfiguration obsConfiguration = new ObsConfiguration();
obsConfiguration.setEndPoint(endPoint);
// Create an ObsClient instance.
try (ObsClient obsClient = new ObsClient(ak, sk, securityToken, obsConfiguration)) {
    String exampleBucket = "examplebucket";
    String exampleObject = "objectname";
    String fileToUpload = "exampleFileName";
    long contentLength = new File(fileToUpload).length();
    PutObjectRequest request = new PutObjectRequest(exampleBucket, exampleObject, new
FileInputStream(fileToUpload));
    ObjectMetadata objectMetadata = new ObjectMetadata();
    // Streaming uploads require the object attribute Content-Length must be configured.
    // Otherwise, ProgressStatus.getTransferPercentage () returns -1.
    objectMetadata.setContentLength(contentLength);
    request.setMetadata(objectMetadata);
    request.setProgressListener(
        new ProgressListener() {
            @Override
            public void progressChanged(ProgressStatus status) {
                // Obtain the average upload rate.
                System.out.println("AverageSpeed:" + status.getAverageSpeed());
                // Obtain the upload progress in percentage.
                System.out.println("TransferPercentage:" + status.getTransferPercentage());
            }
        });
    // Refresh the upload progress each time 1 MB data is uploaded.
    request.setProgressInterval(1024 * 1024L);
    obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.5 Creating a Folder (SDK for Java)

Function

This API creates a folder in an existing bucket to manage data in OBS.

OBS does not involve folders like in a file system. All elements in buckets are objects. To create a folder in OBS is essentially to create an object whose size is 0 and whose name ends with a slash (/). Such objects are no different from other

objects (you can perform normal operations on them such as download and delete), except that they are displayed as folders in OBS console.

Restrictions

- To create a folder, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- To create a multi-level folder, you only need to create the folder with the last level. For example, if you want to create a folder named **src1/src2/src3/**, create it directly. You do not need to create folders **src1/** and **src1/src2/** first.

Method

obsClient.putObject([PutObjectRequest request](#))

Request Parameters

Table 8-39 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	PutObjectRequest	Yes	Explanation: Request parameters for uploading an object. For details, see Table 8-40 .

Table 8-40 PutObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-47 . Default value: None
acl	AccessControlList	No	Explanation: ACL that can be pre-defined when an object is created. Refer to Table 8-48 to choose the option you need. Value range: - To use a pre-defined ACL, see Table 8-41 for the available options. - To use a user-defined ACL, see Table 8-48 to configure the required parameters. Default value: None
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-45. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call. Value range: An integer greater than 0, in days Default value: None
progressListener	ProgressListener	No	Explanation: Upload progress. For details, see Table 8-42.

Parameter	Type	Man dato ry (Yes /No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-41 Pre-defined ACL

Constant	Description
AccessControllist.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-42 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-43. Default value: None

Table 8-43 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-44 . Default value: None

Table 8-44 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-45 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
AES256	AES256

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-46 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-47 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Size of the object. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object to be uploaded. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If this header is not set, the default storage class of the bucket is used as the storage class of the object. Value range: See Table 8-46. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-48 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-49.

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-50 .

Table 8-49 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-50 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-52 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-51 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-51 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 8-52 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 8-53 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 8-54 . Default value: None

Table 8-53 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-54 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Responses

Table 8-55 PutObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
storageClass	StorageClassEnum	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-16 . Default value: None
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version ID will be returned. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Value range: The value must contain 32 characters. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my-.bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Code Examples

This example uses the object upload operation to create folder **parent_directory/** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import java.io.ByteArrayInputStream;
public class PutObject006 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
```

```
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Create a folder.
    final String keySuffixWithSlash = "parent_directory/";
    obsClient.putObject("examplebucket", keySuffixWithSlash, new ByteArrayInputStream(new
byte[0]));
    // Create an object in the folder.
    obsClient.putObject(
        "examplebucket", keySuffixWithSlash + "objectname", new ByteArrayInputStream("Hello
OBS".getBytes()));
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.6 Configuring Object Metadata (SDK for Java)

Function

This API configures metadata for an object when uploading it. Object metadata include the object length, MIME type, MD5 value (for verification), storage class, and custom object metadata. You can configure metadata for an object that is being uploaded in streaming, file-based, or multipart mode or when copying an object.

Restrictions

- To configure the metadata of an object when uploading it, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- An object can have multiple pieces of metadata. The metadata size cannot exceed 8 KB in total.
- Currently, metadata names cannot contain non-ASCII characters. If such characters are contained in a metadata value, the value must be Base64-encoded.

Method

```
obsClient.putObject(PutObjectRequest request)
```

Request Parameters

Table 8-56 List of request parameters

Parameter	Type	Mandator y (Yes/No)	Description
request	PutObject Request	Yes	Explanation: Request parameters for uploading an object. For details, see Table 8-57 .

Table 8-57 PutObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-64. Default value: None
acl	AccessControlList	No	Explanation: ACL that can be pre-defined when an object is created. Refer to Table 8-65 to choose the option you need. Value range: - To use a pre-defined ACL, see Table 8-58 for the available options. - To use a user-defined ACL, see Table 8-65 to configure the required parameters. Default value: None
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-62. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call. Value range: An integer greater than 0, in days. Default value: None
progressListener	ProgressListener	No	Explanation: Upload progress. For details, see Table 8-59.

Parameter	Type	Man dato ry (Yes /No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-58 Pre-defined ACL

Constant	Description
AccessControllist.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-59 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-60. Default value: None

Table 8-60 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-61 . Default value: None

Table 8-61 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-62 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
AES256	AES256

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-63 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-64 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the file.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the file. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClass	StorageClassEnum	No	Explanation: Object storage class that can be specified at object creation. If this parameter is not specified, the object inherits the storage class of the bucket where it is to be uploaded by default. Value range: See Table 8-63. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The metadata size cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-65 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-66.

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-67 .

Table 8-66 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-67 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-69 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-68 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-68 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Table 8-69 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 8-70 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 8-71 . Default value: None

Table 8-70 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-71 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Responses

Table 8-72 PutObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
storageClass	StorageClassEnum	Explanation: Object storage class that can be specified at object creation. If this parameter is not specified, the object inherits the storage class of the bucket where it is to be uploaded by default. Value range: See Table 8-63 . Default value: None

Parameter	Type	Description
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version ID will be returned. Value range: The value must contain 32 characters. Default value: None
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An uploaded object or copied object has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Code Example: Setting the Length for an Object

This example uploads local file **localfile** to bucket **examplebucket** as object **objectname** and sets its length to **1024 * 1024L**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import java.io.File;
import java.io.IOException;
public class PutObject007 {
    public static void main(String[] args) throws IOException {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
```

```
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the length for an object.
    ObjectMetadata metadata = new ObjectMetadata();
    metadata.setContentLength(1024 * 1024L); // 1MB
    obsClient.putObject("examplebucket", "objectname", new File("localfile"), metadata);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Setting the MIME Type for an Object

This example uploads local file **localimage.jpg** to bucket **examplebucket** as object **objectname.jpg** and sets the MIME type of the object to **image/jpeg**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import java.io.File;

public class PutObject008 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
```

```
try {
    // Set the MIME type for an object.
    // Upload an image.
    ObjectMetadata metadata = new ObjectMetadata();
    metadata.setContentType("image/jpeg");
    obsClient.putObject("examplebucket", "objectname.jpg", new File("localimage.jpg"), metadata);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Setting the MD5 Value for an Object

This example uploads local file **localimage.jpg** to bucket **examplebucket** as object **objectname** and calls **ObsClient.base64Md5** to calculate the **Content-MD5** header.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import java.io.File;
import java.nio.file.Files;
import java.nio.file.Paths;

public class PutObject009 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the MD5 value for an object.
            // Upload an image.
            ObjectMetadata metadata = new ObjectMetadata();
            // Your file's Base64-encoded MD5
            String base64Md5 = obsClient.base64Md5(Files.newInputStream(Paths.get("localimage.jpg")));
```

```
        metadata.setContentMd5(base64Md5);
        obsClient.putObject("examplebucket", "objectname", new File("localimage.jpg"), metadata);
        System.out.println("putObject successfully");
    } catch (ObsException e) {
        System.out.println("putObject failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("putObject failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

Code Example: Customizing Metadata for an Object

This example uploads local file **localfile** to object **objectname** in bucket **examplebucket**, customizes two pieces of metadata named **property1** and **property2**, and sets their respective values to **property-value1** and **property-value2**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import java.io.File;
public class PutObject011 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set user-defined metadata for the object.
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.addUserMetadata("property1", "property-value1");
            metadata.addUserMetadata("property2", "property-value2");
            obsClient.putObject("examplebucket", "objectname", new File("localfile"), metadata);
            System.out.println("putObject successfully");
        } catch (ObsException e) {
            System.out.println("putObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
        }
    }
}
```

```
// Request failed. Print the server-side error code.
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Setting the Storage Class for an Object

This example uploads local file **localfile** to bucket **exampleBucket** as object **objectname** and sets the object storage class to Warm.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.StorageClassEnum;
import java.io.File;
public class PutObject010 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the storage class for an object.
            ObjectMetadata metadata = new ObjectMetadata();
            // Set the storage class of the object to Warm.
            metadata.setObjectStorageClass(StorageClassEnum.WARM);
            obsClient.putObject("examplebucket", "objectname", new File("localfile"), metadata);
            System.out.println("putObject successfully");
        } catch (ObsException e) {
            System.out.println("putObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
```



```
        System.out.println("putObject failed");  
        // Print other error information.  
        e.printStackTrace();  
    }  
}
```

8.7 Multipart Upload (SDK for Java)

8.7.1 Multipart Upload Overview (SDK for Java)

Function

You can upload large files using multipart upload. Multipart upload is applicable to many scenarios, including:

- Files to be uploaded are larger than 100 MB.
- The network condition is poor. Connection to the OBS server is constantly down.
- Sizes of files to be uploaded are uncertain.

A multipart upload has the following advantages:

- Higher throughput: You can upload parts in parallel to improve throughput.
- Quick recovery from network failures: Uploading an object in parts helps minimize the impact of upload interruptions caused by network failures.
- Flexible suspension and resumption of object uploads: You can upload parts at any time. A multipart upload will not expire after being initiated. You must explicitly complete or abort a multipart upload.
- No need to know the size before uploading an object: You can upload an object while creating it.

A multipart upload consists of the following steps:

Step 1 [Initiate a multipart upload](#) (`ObsClient.initiateMultipartUpload`).

Step 2 [Upload parts one by one or concurrently](#) (`ObsClient.uploadPart`).

Step 3 [Assemble parts](#) (`ObsClient.completeMultipartUpload`) or [abort the multipart upload](#) (`ObsClient.abortMultipartUpload`).

----End

8.7.2 Initiating a Multipart Upload (SDK for Java)

Function

This API initiates a multipart upload and returns a globally unique upload ID. You can use this upload ID in your subsequent requests including **`abortMultipartUpload`**, **`listMultipartUploads`**, and **`listParts`**.

Restrictions

- To initiate a multipart upload, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- After initiating a multipart upload and uploading one or more parts, you must assemble the parts or abort the multipart upload.

Method

obsClient.initiateMultipartUpload([InitiateMultipartUploadRequest request](#))

Request Parameters

Table 8-73 List of request parameters

Parameter	Type	Man dato ry (Yes/ No)	Description
request	InitiateMultipartUploadRequest	Yes	Explanation: Request parameters for initiating a multipart upload. For details, see Table 8-74 .

Table 8-74 InitiateMultipartUploadRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-78. Default value: None
acl	AccessControlList	No	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 8-75 for the available options. - To use a user-defined ACL, see Table 8-79 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-76. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call after the object has been uploaded. Value range: An integer greater than 0, in days. Default value: None
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-75 Pre-defined ACL

Constant	Description
AccessControlList.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.

Constant	Description
AccessControllist.REST_CAN NED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN NED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN NED_PUBLIC_READ_DELIVER ED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN NED_PUBLIC_READ_WRITE_ DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-76 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
AES256	AES256

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-77 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-78 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the file.

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None
storageClass	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-77. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The metadata size cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-79 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-80.

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-81 .

Table 8-80 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-81 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-83 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-82 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-82 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 8-83 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 8-84.
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 8-85. Default value: None

Table 8-84 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-85 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

List of returned results

Table 8-86 InitiateMultipartUploadResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
uploadId	String	Explanation: The ID created by the OBS server to identify the multipart upload globally. Value range: The value must contain 1 to 32 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Name of the bucket involved in the multipart upload. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Description
objectKey	String	Explanation: Name of the object to be uploaded. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Code Examples

This example initiates a multipart upload for object **objectname** in bucket **examplebucket** and obtains the upload ID.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.InitiateMultipartUploadRequest;
import com.obs.services.model.InitiateMultipartUploadResult;
import com.obs.services.model.ObjectMetadata;
public class InitiateMultipartUpload001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest("examplebucket",
"objectname");
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.addUserMetadata("property", "property-value");
            metadata.setContentType("text/plain");
            request.setMetadata(metadata);
            InitiateMultipartUploadResult result = obsClient.initiateMultipartUpload(request);
            String uploadId = result.getUploadId();
```

```
        System.out.println("initiateMultipartUpload successfully");
        System.out.println("uploadId:" + uploadId);
    } catch (ObsException e) {
        System.out.println("initiateMultipartUpload failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("initiateMultipartUpload failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

8.7.3 Uploading a Part (SDK for Java)

Function

After a multipart upload is initiated, this API uploads a part to a specified bucket. In the upload request, the multipart upload ID must be included. Except for the part lastly being uploaded whose size ranges from 0 to 5 GB, sizes of the other parts range from 100 KB to 5 GB. Part numbers can be any number from 1 to 10,000.

When uploading a part, you must specify its upload ID and part number. A part number uniquely identifies a part and its position in the object you are uploading. If you upload a new part with the same part number as that of a previous part, the previously uploaded part will be overwritten. Whenever you upload a part, OBS returns the ETag header in the response. For each part upload task, you must record the part number and ETag value. These values are required in subsequent requests for you to complete a multipart upload.

Restrictions

- To upload a part, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- After initiating a multipart upload and uploading one or more parts, you must assemble the parts or abort the multipart upload.
- **partNumber** in a multipart upload must be unique. When the same **partNumber** of the same object is concurrently uploaded, last write wins policy is applied. The time of last write is defined as the time when the part metadata is created. To ensure data accuracy, the client must be locked to ensure concurrent uploads of the same part of the same object. Concurrent uploads for different parts of the same object do not need to be locked.

Method

obsClient.uploadPart([UploadPartRequest request](#))

Request Parameters

Table 8-87 List of request parameters

Parameter	Type	Man dato ry (Yes/ No)	Description
request	UploadPartRequ est	Yes	Explanation: Request parameters for uploading a part. For details, see Table 8-88 .

Table 8-88 UploadPartRequest

Parameter	Type	Mandato ry (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
partNumber	int	Yes	Explanation: Part number. Value range: The value ranges from 1 to 10,000. If a part number is not within the range, OBS returns error code 400 Bad Request. Default value: None
uploadId	String	Yes	Explanation: Multipart upload ID, which can be returned by initiating a multipart upload, for example, 000001648453845DBB78F2340DD460D8. Restrictions: The value must contain 32 characters. Default value: None
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Restrictions: You must specify either file or input. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Restrictions: You must specify either file or input. If one of the two parameters is left blank, the other must be specified. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
offset	long	No	Explanation: Start offset of a part in the source file. This parameter is not supported for streams. Value range: A non-negative integer not exceeding the size of the object to be uploaded, in bytes. Default value: 0
partSize	Long	No	Explanation: Part size. Restrictions: • Except the part last uploaded, each of the other parts must be larger than 100 KB. The size of each part is not verified during its upload, because the system cannot define whether it is the last part. The part size is verified when the parts are assembled. • The minimum part size supported by an OBS 3.0 bucket is 100 KB, and that supported by an OBS 2.0 bucket is 5 MB. Value range: The value ranges from 100 KB to 5 GB, in bytes. Default value: 102400

Parameter	Type	Mandatory (Yes/No)	Description
attachMd5	boolean	No	Explanation: Whether to automatically calculate the MD5 value of the data to be uploaded. To ensure data integrity, set UploadPartRequest.setAttachMd5 to true to make the SDK automatically calculate the MD5 value (valid only when the data source is a local file) of each part and add the MD5 value to the Content-MD5 request header. The OBS server will calculate the MD5 value of the uploaded data and compare it with the MD5 value calculated by the SDK. Restrictions: If attachMd5 and contentMd5 are used at the same time, attachMd5 is ignored. Value range: true: The MD5 value of the data to be uploaded is automatically calculated. false: The MD5 value of the data to be uploaded is not automatically calculated. Default value: false

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the part to be uploaded, which uniquely identifies the content of the uploaded part and can be used to identify whether the object content is changed. Restrictions: If attachMd5 and contentMd5 are used at the same time, attachMd5 is ignored. Value range: The value must contain 32 characters. Default value: None
needCalculateCRC64	boolean	No	Explanation: Whether to automatically calculate the CRC64 value of the data to be uploaded. Restrictions: This parameter is not supported for POSIX or SFS objects. This parameter is supported only for file uploads. Value range: true: The SDK calculates CRC64 and submits it to the server for verification. false: CRC64 is not calculated. Default value: false
progressListener	ProgressListener	No	Explanation: Upload progress. For details, see Table 8-90.

Parameter	Type	Mandatory (Yes/No)	Description
progressInterval	long	No	Explanation: Interval for reporting the upload progress, in bytes. For example, 1024 * 1024L indicates that the upload progress is reported each time 1 MB data is uploaded. Default value: 100 * 1024L
autoClose	boolean	No	Explanation: Whether to automatically close data streams after the upload is complete. Value range: true: The data stream is automatically closed. false: The data stream is not automatically closed. Default value: true

Constant	Default Value
AES256	AES256

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-89 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-90 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-91 . Default value: None

Table 8-91 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-92 . Default value: None

Table 8-92 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.

Method	Return Value Type	Description
getInstantaneous-Speed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes() ()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Responses

Table 8-93 UploadPartResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
partNumber	int	Explanation: Part number. Value range: [1,10000] Default value: None

Parameter	Type	Description
etag	String	Explanation: Base64-encoded, 128-bit MD5 value of a part. ETag is the unique identifier of the part contents and is used to determine whether the contents of a part are changed. For example, if the ETag value is A when a part is uploaded and is B when the part is downloaded, this indicates the contents of the part are changed. The ETag reflects changes only to the contents of a part, not its metadata. Parts created by the upload and copy operations have unique ETags after being encrypted using MD5. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
clientCalculatedCRC64	String	Explanation: The 64-bit CRC value calculated for the part by the client based on the ECMA-182 standard. The CRC64 value is the unique identifier of the object content. It can be used to determine whether the object content is changed. If an object has different CRC64 values when being uploaded and downloaded, its content has been changed. CRC64 reflects changes to the contents of the object, not its metadata. Restrictions: - This parameter is returned when the CRC64 value was verified for the uploaded part, or when the CRC64 feature has been enabled for the bucket. - This parameter is not supported for POSIX or SFS objects. Value range: A 64-bit CRC value calculated based on the ECMA-182 standard Default value: None

Code Examples

This example calls **obsClient.uploadPart** to upload parts to object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PartEtag;
import com.obs.services.model.UploadPartRequest;
import com.obs.services.model.UploadPartResult;
import java.io.File;
import java.util.ArrayList;
import java.util.List;
public class UploadPart001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            String uploadId = "upload id from initiateMultipartUpload";
            List<PartEtag> partEtags = new ArrayList<PartEtag>();
            // Upload the first part.
            UploadPartRequest request = new UploadPartRequest("examplebucket", "objectname");
            // Set the upload ID.
            request.setUploadId(uploadId);
            // Set the part number, which ranges from 1 to 10000.
            request.setPartNumber(1);
            // Specify the large file to be uploaded.
            request.setFile(new File("localfile"));
            // Set the part size.
            request.setPartSize(5 * 1024 * 1024L);
            UploadPartResult result = obsClient.uploadPart(request);
            partEtags.add(new PartEtag(result.getEtag(), result.getPartNumber()));
            // Upload the second part.
            request = new UploadPartRequest("examplebucket", "objectname");
            // Set the upload ID.
            request.setUploadId(uploadId);
            // Set the part number.
            request.setPartNumber(2);
            // Specify the large file to be uploaded.
            request.setFile(new File("localfile"));
            // Set the offset of the second part. This parameter takes effect only when a file is uploaded. This
            // parameter is not supported for streams.
            request.setOffset(5 * 1024 * 1024L);
            // Set the part size.
            request.setPartSize(5 * 1024 * 1024L);
            result = obsClient.uploadPart(request);
            partEtags.add(new PartEtag(result.getEtag(), result.getPartNumber()));
```



```
        System.out.println("uploadPart successfully");
    } catch (ObsException e) {
        System.out.println("uploadPart failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("uploadPart failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

8.7.4 Assembling Parts (SDK for Java)

Function

This API assembles the uploaded parts to complete the multipart upload. Before performing this operation, you cannot download the uploaded data. When assembling parts, you need to copy the additional message header information recorded during the multipart upload initiation to the object metadata. Such information is processed the same way the information in a common object upload is processed. In the case of assembling parts concurrently, last write wins is applied, but the time of last write is defined as the time when a multipart upload was initiated.

As long as the multipart upload is not aborted, all uploaded parts occupy the space. However, after you assembled the specified parts, those uploaded but not assembled will be deleted to free up space.

When assembling parts, OBS creates an object by putting part numbers in ascending order. If any object metadata is provided in the initiation of the multipart upload, OBS will associate the metadata with the object. After the multipart upload is complete, the parts will no longer exist. A part assembling request must contain the upload ID, part numbers, and a list of corresponding ETag values. In response to the request, the ETag that uniquely identifies the assembled parts is contained. This ETag is not the MD5 hash value of the entire object.

Restrictions

- To assemble parts, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- After a multipart upload is complete, the uploaded parts that are not assembled will be automatically deleted and cannot be recovered. Before assembling parts, use the API for listing uploaded parts to check all parts to ensure that no part is missed.

Method

obsClient.completeMultipartUpload([CompleteMultipartUploadRequest request](#))

Request Parameters

Table 8-94 List of request parameters

Parameter	Type	Man dato ry (Yes/ No)	Description
request	CompleteMulti partUploadReques t	Yes	Explanation: Request parameters for assembling parts. For details, see Table 8-95 .

Table 8-95 CompleteMultipartUploadRequest

Parameter	Type	Mandato ry (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
partEtag	List< PartEtag >	Yes	Explanation: List of parts to be assembled. For details, see Table 8-96 .
uploadId	String	Yes	Explanation: Multipart upload ID, for example, 000001648453845DBB78F2340DD460D8 . Value range: The value must contain 32 characters. Default value: None
encodingType	String	No	Explanation: Encoding type for objectKey in the response. If objectKey in the response contains control characters that are not supported by the XML 1.0 standard, you can specify this parameter to encode objectKey . Value range: url Default value: None. If you leave this parameter blank, encoding is not applied.
userHeaders	HashMap<String, String>	No	Explanation: User header list. In HashMap , the String key and value indicate the name and value of the user header field respectively. The SDK does not process the userHeaders and instead transparently transmits it to the server for later use. Default value: None

Table 8-96 PartEtag

Parameter	Type	Mandatory (Yes/No)	Description
etag	String	Yes	Explanation: Part ETag. Base64-encoded, 128-bit MD5 value of the part. Value range: The value must contain 32 characters. Default value: None
partNumber	Integer	Yes	Explanation: Part number. Part numbers can be inconsecutive. Value range: An integer ranging from 1 to 10000. Default value: None

Responses

Table 8-97 CompleteMultipartUploadResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
etag	String	Explanation: Base64-encoded, 128-bit MD5 value of an object. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket in which parts are assembled. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.) – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: The name of the object the parts are assembled into. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
location	String	Explanation: URL of the object the parts are assembled into. Default value: None
versionId	String	Explanation: Version ID of the object the parts are assembled into. If versioning is enabled for the bucket, the object version number will be returned. Value range: The value must contain 32 characters. Default value: None
objectUrl	String	Explanation: Full path to the object the parts are assembled into. Default value: None
encodingType	String	Explanation: Encoding type for objectKey in the response. If objectKey in the response contains control characters that are not supported by the XML 1.0 standard, you can specify this parameter to encode objectKey . Value range: url Default value: None. If you leave this parameter blank, encoding is not applied.

Code Examples

This example calls **ObsClient.completeMultipartUpload** to assemble parts into object **objectname** in bucket **examplebucket** based on **uploadId** and **partEtags**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CompleteMultipartUploadRequest;
import com.obs.services.model.PartEtag;
import java.util.ArrayList;
import java.util.List;
public class CompleteMultipartUpload001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
```



```
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    String uploadId = "upload id from initiateMultipartUpload";
    List<PartEtag> partEtags = new ArrayList<PartEtag>();
    // First part
    PartEtag part1 = new PartEtag();
    part1.setPartNumber(1);
    part1.setEtag("etag1");
    partEtags.add(part1);
    // Second part
    PartEtag part2 = new PartEtag();
    part2.setPartNumber(2);
    part2.setEtag("etag2");
    partEtags.add(part2);
    CompleteMultipartUploadRequest request =
        new CompleteMultipartUploadRequest("examplebucket", "objectname", uploadId, partEtags);
    obsClient.completeMultipartUpload(request);
    System.out.println("completeMultipartUpload successfully");
} catch (ObsException e) {
    System.out.println("CompleteMultipartUpload failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("completeMultipartUpload failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.7.5 Code Example of a Multipart Upload (SDK for Java)

Multipart upload is mainly used for large file upload or when the network connection is poor.

You can use **UploadPartRequest.setOffset** and **UploadPartRequest.setPartSize** to set the start and end positions for uploading a part.

This example uploads a large file by concurrently uploading parts.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.CompleteMultipartUploadRequest;
import com.obs.services.model.InitiateMultipartUploadRequest;
import com.obs.services.model.InitiateMultipartUploadResult;
import com.obs.services.model.PartEtag;
import com.obs.services.model.UploadPartRequest;
import com.obs.services.model.UploadPartResult;
import java.io.File;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.TimeUnit;
public class ConcurrentUploadPart001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            final String bucketName = "examplebucket";
            final String objectKey = "objectname";
            // Initialize the thread pool.
            ExecutorService executorService = Executors.newFixedThreadPool(20);
            final File largeFile = new File("localfile");
            // Initiate a multipart upload.
            InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest(bucketName,
objectKey);
            InitiateMultipartUploadResult result = obsClient.initiateMultipartUpload(request);
            final String uploadId = result.getUploadId();
            System.out.println("\t" + uploadId + "\n");
            // Set the part size to 100 MB.
            long partSize = 100 * 1024 * 1024L;
            long fileSize = largeFile.length();
            // Calculate the number of parts to be uploaded.
            long partCount = fileSize % partSize == 0 ? fileSize / partSize : fileSize / partSize + 1;
            final List<PartEtag> partEtags = Collections.synchronizedList(new ArrayList<PartEtag>());
            // Start uploading parts concurrently.
            for (int i = 0; i < partCount; i++) {
                // Set the start position of a part in the file.
                final long offset = i * partSize;
                // Set the part size.
                final long currPartSize = (i + 1 == partCount) ? fileSize - offset : partSize;
                // Set the part number.
                final int partNumber = i + 1;
                executorService.execute(
                    new Runnable() {
                        @Override
                        public void run() {
                            UploadPartRequest uploadPartRequest = new UploadPartRequest();
                            uploadPartRequest.setBucketName(bucketName);
                            uploadPartRequest.setObjectKey(objectKey);
```

```
        uploadPartRequest.setUploadId(uploadId);
        uploadPartRequest.setFile(largeFile);
        uploadPartRequest.setPartSize(currPartSize);
        uploadPartRequest.setOffset(offset);
        uploadPartRequest.setPartNumber(partNumber);
        UploadPartResult uploadPartResult;
        try {
            uploadPartResult = obsClient.uploadPart(uploadPartRequest);
            System.out.println("Part#" + partNumber + " done\n");
            partEtags.add(
                new PartEtag(uploadPartResult.getEtag(),
uploadPartResult.getPartNumber()));
        } catch (ObsException e) {
            e.printStackTrace();
        }
    }
    });
}
// Wait until the upload is complete.
executorService.shutdown();
while (!executorService.isTerminated()) {
    try {
        executorService.awaitTermination(5, TimeUnit.SECONDS);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}
// Assemble parts.
CompleteMultipartUploadRequest completeMultipartUploadRequest =
    new CompleteMultipartUploadRequest(bucketName, objectKey, uploadId, partEtags);
obsClient.completeMultipartUpload(completeMultipartUploadRequest);
System.out.println("completeMultipartUpload successfully");
} catch (ObsException e) {
    System.out.println("CompleteMultipartUpload failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("completeMultipartUpload failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.7.6 Aborting a Multipart Upload (SDK for Java)

Function

This API aborts a multipart upload using the multipart upload ID.

After a multipart upload is aborted, the upload ID cannot be used to upload any part. The storage occupied by any uploaded parts will be released. If any part uploads are in progress, aborting the multipart upload might or might not make the uploads successful. To release the storage occupied by all uploaded parts, you can only abort the multipart upload after all parts have been uploaded.

Restrictions

- To abort a multipart upload, you must be the bucket owner or have the required permission (**obs:object:AbortMultipartUpload** in IAM or **AbortMultipartUpload** in a bucket policy).

Method

obsClient.abortMultipartUpload([AbortMultipartUploadRequest request](#))

Request Parameters

Table 8-98 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	AbortMultipartUploadRequest	Yes	Explanation: Request parameters for aborting a multipart upload. For details, see Table 8-99 .

Table 8-99 AbortMultipartUploadRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Bucket name. Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
uploadId	String	Yes	Explanation: Multipart upload ID, for example, 000001648453845DBB78F2340DD460D8 . Value range: The value must contain 32 characters. Default value: None

Responses

Table 8-100 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example calls **ObsClient.abortMultipartUpload** to abort the multipart upload of object **objectname** in bucket **examplebucket** based on the upload ID obtained during the multipart upload initiation.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.AbortMultipartUploadRequest;  
public class AbortMultipartUpload001 {
```

```
public static void main(String[] args) {
    // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
    // hard coding may result in leakage.
    // Obtain an AK/SK pair on the management console.
    String ak = System.getenv("ACCESS_KEY_ID");
    String sk = System.getenv("SECRET_ACCESS_KEY_ID");
    // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
    // advised not to use hard coding, which may result in information leakage.
    // Obtain an AK/SK pair and a security token using environment variables or import them in other
    // ways.
    String securityToken = System.getenv("SECURITY_TOKEN");
    // Enter the endpoint corresponding to the region where the bucket is to be created.
    String endPoint = "https://your-endpoint";
    // Obtain an endpoint using environment variables or import it in other ways.
    //String endPoint = System.getenv("ENDPOINT");

    // Create an ObsClient instance.
    // Use the permanent AK/SK pair to initialize the client.
    ObsClient obsClient = new ObsClient(ak, sk, endPoint);
    // Use the temporary AK/SK pair and security token to initialize the client.
    // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

    try {
        String uploadId = "upload id from initiateMultipartUpload";
        // Abort the multipart upload request.
        AbortMultipartUploadRequest request = new AbortMultipartUploadRequest("examplebucket",
"objectname", uploadId);
        // Abort the multipart upload API.
        obsClient.abortMultipartUpload(request);
        System.out.println("AbortMultipartUpload successfully");
    } catch (ObsException e) {
        System.out.println("AbortMultipartUpload failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("AbortMultipartUpload failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

8.7.7 Listing Uploaded Parts (SDK for Java)

Function

This API lists the uploaded parts in a specified bucket. This request must contain the multipart upload ID.

You can list the uploaded parts of a specified multipart upload or of all ongoing multipart uploads. A maximum of 1,000 uploaded parts can be returned in a response. If your multipart upload has more than 1,000 parts, you need to send multiple requests to list all uploaded parts. Assembled parts will not be listed.

Restrictions

- To list uploaded parts, you must be the bucket owner or have the required permission (**obs:object:ListMultipartUploadParts** in IAM or **ListMultipartUploadParts** in a bucket policy).

Method

obsClient.listParts([ListPartsRequest request](#))

Request Parameters

Table 8-101 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	ListPartsRequest	Yes	Explanation: Request parameters for listing uploaded parts. For details, see Table 8-102 .

Table 8-102 ListPartsRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
uploadId	String	Yes	Explanation: Multipart upload ID, for example, 000001648453845DBB78F2340DD460D8. Value range: The value must contain 1 to 32 characters. Default value: None
maxParts	Integer	No	Explanation: Maximum number of parts that can be listed per page. Restrictions: If the specified value is greater than 1000, only 1,000 parts are returned. Value range: The value ranges from 1 to 1000. Default value: 1000

Parameter	Type	Mandatory (Yes/No)	Description
partNumberMarker	Integer	No	Explanation: Part number the listing starts from. Restrictions: OBS lists only parts with greater numbers than that specified by this parameter. Default value: None
encodingType	String	No	Explanation: Encoding type for key in the response. If key in the response contains control characters that are not supported by the XML 1.0 standard, you can specify this parameter to encode key . Value range: url Default value: None. If you leave this parameter blank, encoding is not applied.

Responses

Table 8-103 ListPartsResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
bucket	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Description
key	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
uploadId	String	Explanation: Multipart upload ID, for example, 000001648453845DBB78F2340DD460D8. Value range: The value must contain 32 characters. Default value: None
initiator	Owner	Explanation: Initiator of the multipart upload. For details, see Table 8-106. Default value: None
owner	Owner	Explanation: Owner of the multipart upload, which is consistent with initiator. For details, see Table 8-106. Default value: None
storageClass	StorageClassEnum	Explanation: Storage class of the object to be uploaded. Value range: See Table 8-105. Default value: None

Parameter	Type	Description
multipartList	List< Multipart >	Explanation: List of uploaded parts. For details, see Table 8-104. Default value: None
maxParts	Integer	Explanation: Maximum number of parts that can be listed per page, which is consistent with that set in the request. Restrictions: If the specified value is greater than 1,000, only 1,000 parts are returned. Value range: The value ranges from 1 to 1,000. Default value: 1000
isTruncated	boolean	Explanation: Whether all parts are returned in the response. Value range: • true: Not all parts are returned. • false: All parts are returned. Default value: None
partNumberMarker	String	Explanation: Part number after which part listing begins, which is consistent with that set in the request. Default value: None
nextPartNumberMarker	String	Explanation: Part number to start with for the next part listing request. If only part of the uploaded parts are returned for the current request, this parameter is included in the response for setting partNumberMarker in the subsequent request. Default value: None

Table 8-104 Multipart

Parameter	Type	Description
partNumber	Integer	Explanation: Part number. Value range: An integer ranging from 1 to 10000. Default value: None
lastModified	Date	Explanation: Last time the part was uploaded. Default value: None
etag	String	Explanation: Part ETag. Base64-encoded, 128-bit MD5 value of the part. Value range: The value must contain 32 characters. Default value: None
size	Long	Explanation: Part size. Default value: None

Table 8-105 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-106 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Code Example: Listing Up to 1,000 Uploaded Parts

This example lists up to 1,000 parts uploaded to object **objectname** in bucket **examplebucket** based on the upload ID obtained through **initiateMultipartUpload**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListPartsRequest;
import com.obs.services.model.ListPartsResult;
import com.obs.services.model.Multipart;
public class ListParts001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
```



```
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    String uploadId = "upload id from initiateMultipartUpload";
    // List uploaded parts. uploadId is obtained from the initiateMultipartUpload API.
    ListPartsRequest request = new ListPartsRequest("examplebucket", "objectname");
    request.setUploadId(uploadId);
    ListPartsResult result = obsClient.listParts(request);
    for (Multipart part : result.getMultipartList()) {
        // Part number, specified during the upload
        System.out.println("PartNumber:" + part.getPartNumber());
        // Part size
        System.out.println("Size:" + part.getSize());
        // Part ETag
        System.out.println("Etag:" + part.getEtag());
        // Time when the part was last uploaded
        System.out.println("LastModified:" + part.getLastModified());
    }
    System.out.println("listParts successfully");
} catch (ObsException e) {
    System.out.println("listParts failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listParts failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Listing All Uploaded Parts

This example lists over 1,000 parts using pagination.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListPartsRequest;
import com.obs.services.model.ListPartsResult;
import com.obs.services.model.Multipart;
public class ListParts002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
```

```
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    String uploadId = "upload id from initiateMultipartUpload";
    // List uploaded parts. uploadId is obtained from the initiateMultipartUpload API.
    ListPartsRequest request = new ListPartsRequest("examplebucket", "objectname");
    request.setUploadId(uploadId);
    ListPartsResult result;
    do {
        result = obsClient.listParts(request);
        for (Multipart part : result.getMultipartList()) {
            // Part number, specified during the upload
            System.out.println("PartNumber:" + part.getPartNumber());
            // Part size
            System.out.println("Size:" + part.getSize());
            // Part ETag
            System.out.println("Etag:" + part.getEtag());
            // Time when the part was last uploaded
            System.out.println("LastModified:" + part.getLastModified());
        }
        request.setPartNumberMarker(Integer.parseInt(result.getNextPartNumberMarker()));
    } while (result.isTruncated());
    System.out.println("listParts successfully");
} catch (ObsException e) {
    System.out.println("listParts failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listParts failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.7.8 Listing Multipart Uploads (SDK for Java)

Function

This API lists ongoing multipart uploads.

Ongoing multipart uploads are the multipart uploads that have been initiated but have not been completed or aborted. A maximum of 1,000 multipart uploads can be returned in a response. If there are over 1,000 ongoing tasks satisfying the list criteria, you need to send more requests to query additional multipart uploads.

Restrictions

- To list multipart uploads, you must be the bucket owner or have the required permission (**obs:bucket:ListBucketMultipartUploads** in IAM or **ListBucketMultipartUploads** in a bucket policy).
- By default, only the bucket owner and multipart upload initiator have this permission. The bucket owner can grant others the **ListBucketMultipartUploads** permission.

Method

obsClient.listMultipartUploads([ListMultipartUploadsRequest request](#))

Request Parameters

Table 8-107 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	ListMultipartUploadsRequest	Yes	Explanation: Request parameters for listing multipart uploads. For details, see Table 8-108 .

Table 8-108 ListMultipartUploadsRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
prefix	String	No	Explanation: Prefix that the object names in the multipart uploads to be listed must contain. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you specify logs/ as the prefix, the multipart uploads of logs/day1, logs/day2, and logs/day3 will be returned. If you leave this parameter blank, all multipart uploads in the bucket will be returned. Restrictions: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
delimiter	String	No	Explanation: Object names are grouped by this parameter, which is often used with prefix. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one commonPrefixes. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one commonPrefixes. Assume that a bucket has objects abcd, abcde, and bbcde in it. If delimiter is set to d and prefix is set to a, objects abcd and abcde are grouped into a commonPrefixes with abcd as the prefix. If only delimiter is set to d, objects abcd and abcde are grouped into a commonPrefixes with abcd as the prefix, and bbcde is grouped separately into another commonPrefixes with bbcd as the prefix. Value range: The value must contain 1 to 1,024 characters. Default value: None
maxUploads	Integer	No	Explanation: Maximum number of multipart uploads to list. Restrictions: If the specified value is greater than 1000, only 1,000 multipart uploads are returned. Value range: An integer from 1 to 1000 Default value: 1000

Parameter	Type	Mandatory (Yes/No)	Description
keyMarker	String	No	Explanation: Object name after which the multipart upload listing begins. Value range: The value of nextKeyMarker in the response body of the last request. Default value: None
uploadIdMarker	String	No	Explanation: Upload ID after which the multipart upload listing begins. Restrictions: This parameter must be used together with keyMarker, indicating multipart uploads with IDs greater than the specified uploadIdMarker for the specified keyMarker are listed. Value range: The value of nextUploadIdMarker in the response body of the last request. Default value: None
encodingType	String	No	Explanation: Encoding type for some elements in the response. If delimiter, keyMarker, prefix, nextKeyMarker, and objectKey contain control characters that are not supported by the XML 1.0 standard, you can set encodingType to encode delimiter, keyMarker, prefix (including the prefix in commonPrefixes), nextKeyMarker, and objectKey in the response. Value range: url Default value: None. If you leave this parameter blank, encoding is not applied.

Responses

Table 8-109 MultipartUploadListing

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Parameter	Type	Description
bucket	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
keyMarker	String	Explanation: Object name after which the multipart upload listing begins, which is consistent with that set in the request. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
uploadIdMarker	String	Explanation: Upload ID after which the multipart upload listing begins, which is consistent with that set in the request. Value range: The value must contain 1 to 32 characters. Default value: None
nextKeyMarker	String	Explanation: Object name to start with for the next multipart upload listing request. If only part of multipart uploads are returned for the current request, this parameter is included in the response for setting keyMarker in the subsequent request. Value range: The value must contain 1 to 1,024 characters. Default value: None
nextUploadIdMarker	String	Explanation: Upload ID to start with for the next multipart upload listing request. It is used with the nextKeyMarker parameter. If only part of multipart uploads are returned for the current request, this parameter is included in the response for setting uploadIdMarker in the subsequent request. Value range: The value must contain 1 to 32 characters. Default value: None

Parameter	Type	Description
prefix	String	Explanation: Object name prefix in multipart uploads, which is consistent with that set in the request. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you specify logs/ as the prefix, the multipart uploads of logs/day1, logs/day2, and logs/day3 will be returned. If you leave this parameter blank, all multipart uploads in the bucket will be returned. Restrictions: The value must contain 1 to 1,024 characters. Default value: None
maxUploads	int	Explanation: Maximum number of listed multipart uploads, which is consistent with that set in the request. Restrictions: If the specified value is greater than 1000, only 1,000 multipart uploads are returned. Value range: An integer from 1 to 1000 Default value: 1000
truncated	boolean	Explanation: Whether all multipart uploads are returned in the response. Value range: • true: Not all multipart uploads are returned. • false: All multipart uploads are returned. Default value: None
multipartTaskList	List< Multipart Upload >	Explanation: List of multipart uploads. For details, see Table 8-110.

Parameter	Type	Description
delimiter	String	Explanation: A character used to group object names in multipart uploads, which is consistent with that set in the request. This parameter is often used with prefix. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one commonPrefixes. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one commonPrefixes. Assume that a bucket has objects abcd, abcde, and bbcde in it. If delimiter is set to d and prefix is set to a, objects abcd and abcde are grouped into a commonPrefixes with abcd as the prefix. If only delimiter is set to d, objects abcd and abcde are grouped into a commonPrefixes with abcd as the prefix, and bbcde is grouped separately into another commonPrefixes with bbcde as the prefix. Value range: The value must contain 1 to 1,024 characters. Default value: None
commonPrefixes	String[]	Explanation: List of object name prefixes grouped according to the delimiter parameter (if specified). Value range: The value must contain 1 to 1,024 characters. Default value: None

Table 8-110 MultipartUpload

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
uploadId	String	Explanation: Multipart upload ID, for example, 000001648453845DBB78F2340DD460D8 . Value range: The value must contain 1 to 32 characters. Default value: None
initiatedDate	java.util.Date	Explanation: Time when the multipart upload is initiated. Restrictions: The time must be in the ISO8601 format. Default value: None
storageClass	StorageClass Enum	Explanation: Storage class of the object to be uploaded. Value range: See Table 8-111 . Default value: None
initiator	Owner	Explanation: Initiator of the multipart upload. For details, see Table 8-112 .
owner	Owner	Explanation: Owner of the multipart upload, which is consistent with initiator . For details, see Table 8-112 .

Table 8-111 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-112 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Code Example: Listing Multipart Uploads

This example lists up to 1,000 multipart uploads of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListMultipartUploadsRequest;
import com.obs.services.model.MultipartUpload;
import com.obs.services.model.MultipartUploadListing;
public class ListMultipartUploads001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
```

```
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    ListMultipartUploadsRequest request = new ListMultipartUploadsRequest("examplebucket");
    MultipartUploadListing result = obsClient.listMultipartUploads(request);
    for (MultipartUpload upload : result.getMultipartTaskList()) {
        System.out.println("UploadId:" + upload.getUploadId());
        System.out.println("ObjectKey:" + upload.getObjectKey());
        System.out.println("InitiatedDate:" + upload.getInitiatedDate());
    }
    System.out.println("ListMultipartUploads successfully");
} catch (ObsException e) {
    System.out.println("ListMultipartUploads failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("ListMultipartUploads failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Listing Multipart Uploads Using Pagination

This example lists over 1,000 multipart uploads using pagination.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListMultipartUploadsRequest;
import com.obs.services.model.MultipartUpload;
import com.obs.services.model.MultipartUploadListing;
public class ListMultipartUploads002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
```



```
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    ListMultipartUploadsRequest request = new ListMultipartUploadsRequest("examplebucket");
    MultipartUploadListing result;
    do {
        result = obsClient.listMultipartUploads(request);
        for (MultipartUpload upload : result.getMultipartTaskList()) {
            System.out.println("UploadId:" + upload.getUploadId());
            System.out.println("ObjectKey:" + upload.getObjectKey());
            System.out.println("InitiatedDate:" + upload.getInitiatedDate());
        }
        request.setKeyMarker(result.getNextKeyMarker());
        request.setUploadIdMarker(result.getNextUploadIdMarker());
    } while (result.isTruncated());
    System.out.println("ListMultipartUploads successfully");
} catch (ObsException e) {
    System.out.println("ListMultipartUploads failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("ListMultipartUploads failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.8 Configuring Lifecycle Rules (SDK for Java)

Function

You can configure lifecycle rules to periodically delete objects or transition objects between storage classes. The object expiration time set using this API takes precedence over that set in a bucket lifecycle rule.

You can also set the object expiration time when uploading an object or initiating a multipart upload.

NOTE

- Objects in the Warm storage class have a minimum storage period of 30 days and those in the Cold storage class have a minimum storage period of 90 days. After an object is transitioned to the Cold storage class, if it stays in this storage class for less than 90 days, you still need to pay for a full 90 days.

Restrictions

- There is no limit on the number of lifecycle rules in a bucket, but the total size of XML descriptions about all lifecycle rules in a bucket cannot exceed 20 KB.
- A maximum of 20 lifecycle rules can be configured for a parallel file system.
- To configure a lifecycle rule for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutLifecycleConfiguration** in IAM or **PutLifecycleConfiguration** in a bucket policy).
- This API only specifies the time duration in days after which an object will expire. The OBS server automatically clears expired objects.

Method

obsClient.putObject([PutObjectRequest request](#))

Request Parameters

Table 8-113 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	PutObjectRequest	Yes	Explanation: Request parameters for uploading an object. For details, see Table 8-114 .

Table 8-114 PutObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-121. Default value: None
acl	AccessControlList	No	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 8-115 for the available options. - To use a user-defined ACL, see Table 8-122 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-119. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: - This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call. - Expired objects will be permanently deleted and cannot be recovered. Value range: An integer greater than 0, in days. Default value: None
progressListener	ProgressListener	No	Explanation: Upload progress. For details, see Table 8-116 .

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-115 Pre-defined ACL

Constant	Description
AccessControllist.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-116 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-117. Default value: None

Table 8-117 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-118 . Default value: None

Table 8-118 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-119 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
AES256	AES256

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-120 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-121 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the file.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the file. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-120. Restrictions: Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The metadata size cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-122 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-123.

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-124 .

Table 8-123 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-124 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-126 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-125 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-125 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Table 8-126 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 8-127 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 8-128 . Default value: None

Table 8-127 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-128 GroupGrantee

Constant	Description
ALL_USERS	All users.

Constant	Description
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Responses

Table 8-129 PutObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
storageClass	StorageClass Enum	Explanation: Object storage class. If the storage class is Standard, this parameter is left blank. Value range: See Table 8-120 . Default value: None

Parameter	Type	Description
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version ID will be returned. Value range: The value must contain 32 characters. Default value: None
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An uploaded object or copied object has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Code Examples

This example uploads **localfile** to object **objectname** in bucket **examplebucket** and sets the object expiration time to 30 days.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject012 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
```

```
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    PutObjectRequest request = new PutObjectRequest();
    request.setBucketName("examplebucket");
    request.setObjectKey("objectname");
    request.setFile(new File("localfile"));
    // When uploading an object, set the object to expire after 30 days.
    request.setExpires(30);
    obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.9 Uploading an Object - Append (SDK for Java)

Function

1. This API uploads a file or folder to an existing OBS bucket. These files can be texts, images, videos, or any other type of files.
2. The **appendObject** operation adds data to the end of an object in a specified bucket. If there is no object with the same names in the bucket, a new object is created. The latest modification time of the object is updated each time an upload is appended.
3. After an appendable upload is successful, you can call **AppendObjectResult.getNextPosition** or use the **ObsClient.getObjectMetadata** API to get the start position for next appending.

Restrictions

- To upload an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- The size of each append upload cannot exceed 5 GB.

- A maximum of 10,000 appendable uploads can be performed on a single object. If you attempt to append more than 10,000 uploads to an object, OBS returns **409 Conflict** with an error code **ObjectNotAppendable**.
- If the object storage class is **COLD** (Cold), this API cannot be called.
- Objects uploaded using **ObsClient.putObject**, referred to as common objects, can overwrite objects uploaded using **ObsClient.appendObject**, referred to as appendable objects. Data cannot be appended to an appendable object anymore once the object has been overwritten by a common object.
- You can append data to an existing object only if it is appendable. Otherwise, an exception will be reported (HTTP status code **409**). To check whether an object is appendable, see [Obtaining Object Metadata \(SDK for Java\)](#).

Method

obsClient.appendObject([AppendObjectRequest request](#))

Request Parameters

Table 8-130 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	AppendObjectRequest	Yes	Explanation: Request parameters for an append upload. For details, see Table 8-131 .

Table 8-131 AppendObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.). - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-138. Default value: None
acl	AccessControlList	No	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 8-132 for the available options. - To use a user-defined ACL, see Table 8-139 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
extensionPermissionMap	Map< ExtensionObjectPermissionEnum >, Set<String>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-136. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured when uploading an object or using the x-obs-expires header in a metadata API call. Value range: An integer greater than 0, in days. Default value: None
needCalculateCRC64	boolean	No	Explanation: Whether to automatically calculate the CRC64 value of the data to be uploaded and submit the CRC64 value to the server for verification. Restrictions: This parameter is not supported for POSIX or SFS objects. This parameter is supported only for file uploads. This parameter is supported only for the first append upload. Value range: true: The SDK calculates CRC64 and submits it to the server for verification. false: CRC64 is not calculated. Default value: false
progressListener	ProgressListener	No	Explanation: Upload progress. For details, see Table 8-133.

Parameter	Type	Mandatory (Yes/No)	Description
position	long	Yes	Explanation: Position where the object data is appended. Restrictions: For an object to be appended, the value of position must be set to 0 when the object is uploaded for the first time. The value of position will be carried in the x-obs-next-append-position header of the response returned by the server when the object is successfully uploaded next time. Default value: None
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 8-132 Pre-defined ACL

Constant	Description
AccessControlList.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControlList.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.

Constant	Description
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControlList.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-133 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-134 . Default value: None

Table 8-134 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-135 . Default value: None

Table 8-135 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-136 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
AES256	AES256

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-137 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-138 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.

Parameter	Type	Mandatory (Yes/No)	Description
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the file. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClass	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-137. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-139 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-140.

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-141 .

Table 8-140 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-141 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-143 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-142 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-142 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Table 8-143 GranteeInterface

Parameter	Description
CanonicalGrantee	Explanation: Grantee (user) information. For details, see Table 8-144 .
GroupGrantee	Explanation: Grantee (user group) information. Value range: See Table 8-145 . Default value: None

Table 8-144 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-145 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.

Constant	Description
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Responses

Table 8-146 AppendObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Description
etag	String	Explanation: Base64-encoded, 128-bit MD5 value of the appended content. The ETag returned for the append upload is the ETag for the appended content, not that for the entire object. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
storageClass	StorageClass Enum	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-137 . Default value: None
nextPosition	long	Explanation: Position from which the next append upload starts Default value: None

Code Examples

This example appends data to object **objectname** in bucket **examplebucket** by specifying an append position.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.AppendObjectRequest;
import com.obs.services.model.AppendObjectResult;
import com.obs.services.model.ObjectMetadata;
import java.io.ByteArrayInputStream;
public class AppendObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try (ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint)) {
            // Append data for the first time.
            AppendObjectRequest request = new AppendObjectRequest();
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname");
            request.setPosition(0);
            request.setInput(new ByteArrayInputStream("Hello OBS".getBytes()));
            AppendObjectResult result = obsClient.appendObject(request);
            // Append data for the second time.
            request.setPosition(result.getNextPosition());
            request.setInput(new ByteArrayInputStream("Hello OBS Again".getBytes()));
            result = obsClient.appendObject(request);
            System.out.println("appendObject successfully");
            System.out.println("NextPosition:" + result.getNextPosition());
            System.out.println("Etag:" + result.getEtag());
            // Use the API for obtaining object metadata to get the start position for the next append.
            ObjectMetadata metadata = obsClient.getObjectMetadata("examplebucket", "objectname");
            System.out.println("NextPosition from metadata:" + metadata.getNextPosition());
        } catch (ObsException e) {
            System.out.println("appendObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("appendObject failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

8.10 Uploading an Object - Resumable (SDK for Java)

Function

The resumable upload is an encapsulated and enhanced version of the multipart upload used for dealing with possible upload failures of large files when the network connection is unstable or a program crashes. This API splits the file into multiple parts and uploads them individually. The upload result of each part is recorded in a checkpoint file in real time. A success message is returned only when all parts are uploaded. If any parts fail, an error message is returned telling you to call the API again to upload the failed parts. Since the checkpoint file contains the progress of each part, it saves you uploading all parts again in the event of an error.

Restrictions

- To upload an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- The file uploaded by the resumable upload API must exceed 100 KB.
- To obtain the progress of the last upload, you must enable resumable upload when you use this API.

Method

obsClient.uploadFile([UploadFileRequest request](#))

Request Parameters

Table 8-147 uploadFile

Parameter	Type	Mandatory (Yes/No)	Description
request	UploadFileRequest	Yes	Explanation: Request parameters for uploading an object. For details, see Table 8-148 .

Table 8-148 UploadFileRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-)

Parameter	Type	Mandatory (Yes/No)	Description
			adjacent to each other, for example, my-.bucket or my.-bucket. If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
objectMetadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 8-155. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
acl	AccessControlList	No	Explanation: An ACL that can be specified at bucket creation. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 8-149 for the available options. - To use a user-defined ACL, see Table 8-156 to configure the required parameters. Default value: AccessControlList. REST_CANNED_PRIVATE
enableCheckpoint	boolean	No	Explanation: Whether to enable the resumable mode. Value range: true: The resumable mode is enabled. false: The resumable mode is disabled. Default value: false

Parameter	Type	Mandatory (Yes/No)	Description
checkpointFile	String	No	Explanation: Path of a file generated for recording the progress of a resumable upload. The file contains the information about parts and the upload progress. Restrictions: This parameter is valid only for resumable uploads. Default value: If this parameter is left blank, the progress file will be in the same directory as the local file to be uploaded.
uploadFile	String	Yes	Explanation: Full path of the file or folder to be uploaded, for example, aa/bb.txt or aa/. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map< ExtensionObjectPermissionEnum , Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 8-153. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
progressListener	ProgressListener	No	Explanation: Data transmission listener, used for obtaining the upload progress. For details, see Table 8-150.

Parameter	Type	Mandatory (Yes/No)	Description
partSize	long	No	Explanation: Part size. Value range: The value ranges from 100 KB to 5 GB, in bytes. Default value: 9 MB
taskNum	int	No	Explanation: Maximum number of files that can be uploaded concurrently in a multipart upload. Value range: An integer from 1 to 10000 Default value: 1, indicating concurrent uploads are not used.
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Parameter	Type	Mandatory (Yes/No)	Description
needCalculateCRC64	boolean	No	Explanation: Whether to automatically calculate the CRC64 value of the data to be uploaded and submit the CRC64 value to the server for verification. Restrictions: This parameter is not supported for POSIX or SFS objects. Value range: true: The SDK calculates CRC64 and submits it to the server for verification. false: CRC64 is not calculated. Default value: false

Table 8-149 Pre-defined ACL

Constant	Description
AccessControlList.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControlList.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.

Constant	Description
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControlList.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 8-150 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the upload progress. For details, see Table 8-151 . Default value: None

Table 8-151 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 8-152 . Default value: None

Table 8-152 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 8-153 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
AES256	AES256

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Table 8-154 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 8-155 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.

Parameter	Type	Mandatory (Yes/No)	Description
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 8-154. Default value: None
websiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map , the String key indicates the name of the user-defined metadata that starts with x-obs-meta- , and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata . For details, see Obtaining Object Metadata (SDK for Java) . Restrictions: - An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. - When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 8-156 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 8-157 .

Parameter	Type	Mandatory (Yes/No)	Type
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 8-158 .

Table 8-157 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-158 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 8-160 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 8-159 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 8-159 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Table 8-160 GranteeInterface

Parameter	Description
CanonicalGrantee	Explanation: Grantee (user) information. For details, see Table 8-161 .
GroupGrantee	Explanation: Grantee (user group) information. Value range: See Table 8-162 . Default value: None

Table 8-161 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 8-162 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.

Constant	Description
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Responses

Table 8-163 CompleteMultipartUploadResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.) – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version number will be returned. Value range: The value must contain 32 characters. Default value: None

Code Examples

This example uses the resumable method to upload **localfile** to bucket **examplebucket** as object **objectKey**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CompleteMultipartUploadResult;
import com.obs.services.model.UploadFileRequest;
public class UploadFile001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
```

```
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk,endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    UploadFileRequest request = new UploadFileRequest("examplebucket", "objectKey");
    // Configure the local file to be uploaded. localfile is the path of the file to be uploaded. You must
specify a file name with the extension.
    request.setUploadFile("localfile");
    // Set the maximum number of parts that can be concurrently uploaded.
    request.setTaskNum(5);
    // Set the part size to 10 MB.
    request.setPartSize(10 * 1024 * 1024);
    // Enable the resumable upload.
    request.setEnableCheckpoint(true);
    // Perform a resumable upload.
    CompleteMultipartUploadResult result = obsClient.uploadFile(request);
    System.out.println("UploadFile successfully");
} catch (ObsException e) {
    // If there is an exception, you can call the API again to resume the upload.
    System.out.println("UploadFile failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("UploadFile failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.11 Uploading an Object - Browser-Based (SDK for Java)

Function

This API uploads an object up to 5 GB to a specified bucket in HTML form.

You can call **ObsClient.createPostSignature** to generate request parameters for a browser-based upload. You can use code to simulate a browser-based upload. You can also perform a browser-based upload with the following steps:

- Step 1** Call **ObsClient.createPostSignature** to generate request parameters for authentication.

Step 2 Prepare an HTML form.

Step 3 Enter the request parameters in the page.

Step 4 Select a local file and upload it in browser-based mode.

----End

NOTE

There are two request parameters generated for authentication:

- **Policy**, which corresponds to the **policy** parameter in the form
- **Signature**, which corresponds to the **signature** parameter in the form

Restrictions

- To upload an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- Values of **policy** and **signature** in the HTML form are obtained from the returned results of **ObsClient.createPostSignature**.
- During the object upload, the value of **contentType** needs to be manually changed to the corresponding **Content-Type** value.
- You can submit a service ticket to obtain the HTML form example and download it.

Method

obsClient.createPostSignature([PostSignatureRequest request](#))

Table 8-164 createPostSignature

Parameter	Type	Mandatory (Yes/No)	Description
request	PostSignatureRequest	Yes	Explanation: Request parameters for a browser-based upload. For details, see Table 8-165 .

Table 8-165 PostSignatureRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	No	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	No	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
requestDate	Date	No	Explanation: Time the request is initiated. Default value: None
expiryDate	Date	No	Explanation: Date of expiry. Default value: None
expires	long	No	Explanation: Validity period of authentication for a browser-based upload Value range: An integer greater than 0, in seconds. Default value: 300
conditions	List<String>	No	Explanation: The conditions specified for the form. The SDK uses the specified value to calculate the policy and ignores the formParams. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
formParams	Map<String, Object>	No	Explanation: Form parameters in the request. <i>String</i> indicates the name of the form parameter, and <i>Object</i> indicates the value of the form parameter. Default value: None

Responses

Table 8-166 PostSignatureResponse

Parameter	Type	Description
OriginPolicy	String	Explanation: Value of Policy that is not encoded by Base64. This parameter can only be used for verification. For example: <pre>{"expiration":"2023-09-12T12:52:59Z","conditions":[{"content-type":"text/plain"}, {"bucket":"examplebucket"}, {"key":"example/objectname"},]}</pre> Default value: None

Parameter	Type	Description
Policy	String	Explanation: Base64-encoded value of the policy. For example: eyJleHBpcmF0aW9uljoiMjAyMy0wOS0xMlQxMjo1Mjo1OVoiLCJjb25kaXRpb25zIjpbeyJjb250ZW50LXR5cGUiOiJ0ZXh0L3BsYWluIn0seyJidWNrZXQiOiJleGFtcGxlYnVja2V0In0seyJrZXkiOiJleGFtcGxlL29iamVjdG5hbWUifSxdfQ== Default value: None
Signature	String	Explanation: signature in the form. For example: g0jQr4v9VWd1Q2FOFDG6LGfV9Cw= Default value: None

Code Examples

This example generates authorization parameters for a browser-based upload.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PostSignatureRequest;
import com.obs.services.model.PostSignatureResponse;
import java.util.HashMap;
import java.util.Map;
public class PostObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
```

```
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Generate a request for a browser-based upload.
    PostSignatureRequest request = new PostSignatureRequest();
    // Set form parameters.
    Map<String, Object> formParams = new HashMap<String, Object>();
    // Set the object ACL to public read.
    formParams.put("x-obs-acl", "public-read");
    // Set the MIME type for an object.
    formParams.put("content-type", "text/plain");
    request.setFormParams(formParams);
    // Set the validity period for the browser-based upload request, in seconds.
    request.setExpires(3600);
    PostSignatureResponse response = obsClient.createPostSignature(request);
    System.out.println("createPostSignature successfully");
    // Obtain the request parameters.
    System.out.println("Policy:" + response.getPolicy());
    System.out.println("Signature:" + response.getSignature());
} catch (ObsException e) {
    System.out.println("createPostSignature failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("createPostSignature failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

This example is an HTML form.

```
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>
<form action="http://bucketname.your-endpoint/" method="post" enctype="multipart/form-data">
  Object key
  <!-- Object name -->
  <input type="text" name="key" value="objectname" />
  <p>
    ACL
    <!-- Object ACL -->
    <input type="text" name="x-obs-acl" value="public-read" />
  <p>
    Content-Type
    <!-- Object MIME type -->
    <input type="text" name="content-type" value="text/plain" />
  <p>
    <!-- Use the value returned by PostSignatureResponse.getPolicy(). -->
    <input type="hidden" name="policy" value="*** Provide your policy ***" />
    <!-- AK -->
    <input type="hidden" name="AccessKeyId" value="*** Provide your access key ***"/>
    <!-- Signature string -->
    <input type="hidden" name="signature" value="*** Provide your signature ***"/>
    <!-- If x-obs-security-token exists, remove the comment for the following line and specify the actual
    value of x-obs-security-token. -->
    <!-- <input type="hidden" name="x-obs-security-token" value="*** Provide your x-obs-security-token
    ***"/>-->
  </form>
```



```
<input name="file" type="file" />
<input name="submit" value="Upload" type="submit" />
</form>
</body>
</html>
```

8.12 Using CRC64 to Verify File Consistency (SDK for Java)

8.12.1 Using CRC64 to Verify File Consistency During Uploads

You can use CRC64 to verify file consistency during uploads through the following APIs:

- [Uploading an Object - File-Based \(SDK for Java\)](#)
- [Uploading an Object - Append \(SDK for Java\)](#)
- [Multipart Upload \(SDK for Java\)](#)
- [Uploading an Object - Resumable \(SDK for Java\)](#)

Sample Code 1: Uploading an Object - File-Based

The following code shows how to use CRC64 to verify data consistency during an object upload:

```
import static com.obs.services.internal.Constants.CommonHeaders.HASH_CRC64ECMA;
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PutObjectRequest;
import com.obs.services.model.PutObjectResult;
import java.io.File;
import java.util.Map;

public class PutObjectWithCRC64{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Upload a local file.
            String exampleBucket = "examplebucket";
            String localFilePath = "localfile";
            String exampleObjectKey = "objectKey";
```

```
PutObjectRequest putObjectRequest = new PutObjectRequest(exampleBucket, exampleObjectKey,
new File(localFilePath));
putObjectRequest.setNeedCalculateCRC64(true);
PutObjectResult putObjectResult = obsClient.putObject(putObjectRequest);
System.out.println(
    "Server returned crc64 string:" +
putObjectResult.getResponseHeaders().get(HASH_CRC64ECMA));
System.out.println("PutObject successfully");
} catch (ObsException e) {
    System.out.println("PutObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    // In an abnormal scenario, print all headers, which may contain error information.
    Map<String, String> headers = e.getResponseHeaders();
    if(headers != null){
        for (Map.Entry<String, String> header : headers.entrySet()) {
            System.out.println(header.getKey()+"."+header.getValue());
        }
    }
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("PutObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Sample Code 2: Uploading an Object - Append

CAUTION

CRC64 can be used to verify file consistency only during the first append upload.

The following code shows how to use CRC64 to verify data consistency during an append upload:

```
import static com.obs.services.internal.Constants.CommonHeaders.HASH_CRC64ECMA;
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AppendObjectRequest;
import com.obs.services.model.AppendObjectResult;
import java.io.File;
import java.util.Map;
public class AppendObjectWithCRC64{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
```

```
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Append data. CRC64 verification is supported only for the first append upload.
    String exampleBucket = "examplebucket";
    String localFilePath = "localfile";
    String exampleObjectKey = "objectKeyAppend";
    AppendObjectRequest appendObjectRequest = new AppendObjectRequest(exampleBucket);
    appendObjectRequest.setObjectKey(exampleObjectKey);
    appendObjectRequest.setFile(new File(localFilePath));
    appendObjectRequest.setNeedCalculateCRC64(true);
    AppendObjectResult appendObjectResult = obsClient.appendObject(appendObjectRequest);
    System.out.println(
        "Server returned crc64 string:" +
        appendObjectResult.getResponseHeaders().get(HASH_CRC64ECMA));
    System.out.println("AppendObject successfully");
} catch (ObsException e) {
    System.out.println("AppendObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    // In an abnormal scenario, print all headers, which may contain error information.
    Map<String, String> headers = e.getResponseHeaders();
    if (headers != null) {
        for (Map.Entry<String, String> header : headers.entrySet()) {
            System.out.println(header.getKey() + ":" + header.getValue());
        }
    }
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("AppendObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Sample Code 3: Multipart Upload

The following code shows how to use CRC64 to verify data consistency during multipart uploads:

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.internal.utils.CRC64;
import com.obs.services.model.CompleteMultipartUploadRequest;
import com.obs.services.model.CompleteMultipartUploadResult;
import com.obs.services.model.InitiateMultipartUploadRequest;
import com.obs.services.model.InitiateMultipartUploadResult;
import com.obs.services.model.PartEtag;
import com.obs.services.model.UploadPartRequest;
import com.obs.services.model.UploadPartResult;
import java.io.File;
import java.util.ArrayList;
import java.util.List;
import java.util.Map;
```

```
import static com.obs.services.internal.Constants.CommonHeaders.HASH_CRC64ECMA;

public class CompleteMultiPartUploadWithCRC64{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Perform a multipart upload.
            String exampleBucket = "examplebucket";
            String localFilePath = "localfile";
            String exampleObjectKey = "objectKey";
            File exampleFile = new File(localFilePath);
            long exampleFileSize = exampleFile.length();
            long partSize1 = exampleFileSize / 2;
            long partSize2 = exampleFileSize - partSize1;

            InitiateMultipartUploadRequest initiateMultipartUploadRequest =
                new InitiateMultipartUploadRequest(exampleBucket, exampleObjectKey);
            InitiateMultipartUploadResult initiateMultipartUploadResult =
                obsClient.initiateMultipartUpload(initiateMultipartUploadRequest);
            String uploadId = initiateMultipartUploadResult.getUploadId();

            UploadPartRequest uploadPartRequest = new UploadPartRequest(exampleBucket,
                exampleObjectKey,
                exampleFile);
            uploadPartRequest.setUploadId(uploadId);
            uploadPartRequest.setNeedCalculateCRC64(true);
            uploadPartRequest.setPartNumber(1);
            uploadPartRequest.setPartSize(partSize1);
            UploadPartResult uploadPartResult1 = obsClient.uploadPart(uploadPartRequest);
            CRC64 crc641 = uploadPartResult1.getClientCalculatedCRC64();
            System.out.println(
                "Client calculated crc64_1 string:" + crc641.toString());
            uploadPartRequest.setPartNumber(2);
            uploadPartRequest.setFile(exampleFile);
            uploadPartRequest.setOffset(partSize1);
            uploadPartRequest.setPartSize(partSize2);
            UploadPartResult uploadPartResult2 = obsClient.uploadPart(uploadPartRequest);
            CRC64 crc642 = uploadPartResult2.getClientCalculatedCRC64();
            System.out.println(
                "Client calculated crc64_2 string:" + crc642.toString());
            List<PartEtag> eTags = new ArrayList<>();
            eTags.add(new PartEtag(uploadPartResult1.getEtag(), 1));
            eTags.add(new PartEtag(uploadPartResult2.getEtag(), 2));
            CompleteMultipartUploadRequest completeMultipartUploadRequest =
                new CompleteMultipartUploadRequest(exampleBucket, exampleObjectKey ,uploadId ,eTags);
            CRC64 crc64Combined = new CRC64(CRC64.combine(crc641.getValue(), crc642.getValue(),
                partSize2));
            completeMultipartUploadRequest.addUserHeaders("x-obs-" + HASH_CRC64ECMA,
                crc64Combined.toString());
```

```
CompleteMultipartUploadResult completeMultipartUploadResult =
    obsClient.completeMultipartUpload(completeMultipartUploadRequest);
System.out.println("Server returned completeMultipartUpload crc64 string:"
    + completeMultipartUploadResult.getResponseHeaders().get(HASH_CRC64ECMA));
System.out.println("CompleteMultipartUpload successfully");
} catch (ObsException e) {
    System.out.println("AppendObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    // In an abnormal scenario, print all headers, which may contain error information.
    Map<String, String> headers = e.getResponseHeaders();
    if (headers != null) {
        for (Map.Entry<String, String> header : headers.entrySet()) {
            System.out.println(header.getKey() + ":" + header.getValue());
        }
    }
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("AppendObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Sample Code 4: Uploading an Object - Resumable

The following code shows how to use CRC64 to verify data consistency during a resumable upload:

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CompleteMultipartUploadResult;
import com.obs.services.model.UploadFileRequest;
public class UploadFileWithCRC64{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            UploadFileRequest request = new UploadFileRequest("examplebucket", "objectKey");
            // Configure the local file to be uploaded. localfile is the path of the file to be uploaded. You must
            // specify a file name with the extension.
            request.setUploadFile("localfile");
```

```
// Set the maximum number of parts that can be concurrently uploaded.
request.setTaskNum(5);
// Set the part size to 10 MB.
request.setPartSize(10 * 1024 * 1024);
// Enable the resumable upload.
request.setEnableCheckpoint(true);
// Enable CRC64 verification for the file.
request.setNeedCalculateCRC64(true);
// Perform a resumable upload.
CompleteMultipartUploadResult result = obsClient.uploadFile(request);
System.out.println("Server returned crc64 string:" +
result.getResponseHeaders().get(HASH_CRC64ECMA));
System.out.println("UploadFile successfully");
} catch (ObsException e) {
    // If there is an exception, you can call the API again to resume the upload.
    System.out.println("UploadFile failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("UploadFile failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.12.2 Using CRC64 to Verify File Consistency During Downloads

You can call the following API to obtain the CRC64 value of an object and compare it with the locally calculated CRC64 value to verify data consistency:

- [Downloading an Object - Resumable \(SDK for Java\)](#)

Sample Code: Downloading an Object - Resumable

The following code shows how to use CRC64 to verify data consistency during a resumable download:

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.DownloadFileRequest;
import com.obs.services.model.DownloadFileResult;
public class DownloadFile001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
```

```
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    DownloadFileRequest request = new DownloadFileRequest("examplebucket", "objectname");
    // Set the local path the object will be downloaded to.
    request.setDownloadFile("localfile");
    // Set the maximum number of parts that can be concurrently downloaded.
    request.setTaskNum(5);
    // Set the part size to 10 MB.
    request.setPartSize(10 * 1024 * 1024);
    // Enable the resumable download.
    request.setEnableCheckpoint(true);
    // Enable CRC64 verification for the file.
    request.setNeedCalculateCRC64(true);
    // Perform a resumable download.
    DownloadFileResult result = obsClient.downloadFile(request);
    System.out.println("Server returned crc64 string:" + result.getObjectMetadata().getCrc64());
    System.out.println("downloadFile successfully");
} catch (ObsException e) {
    System.out.println("downloadFile failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("downloadFile failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

8.12.3 Obtaining CRC64 of an Object

You can call the following APIs to obtain the CRC64 value of an object and compare it with the locally calculated CRC64 value to verify data consistency:

- [Downloading an Object - Streaming \(SDK for Java\)](#)
- [Downloading an Object - Range-Based \(SDK for Java\)](#)
- [Obtaining Object Metadata \(SDK for Java\)](#)
- [Copying an Object \(SDK for Java\)](#)

CAUTION

If the CRC64 value of an object exists, the CRC64 value is returned. Otherwise, no value is returned.

Sample Code 1: Downloading an Object - Streaming

The following code shows how to obtain the CRC64 value when using streaming to download an object:

```
import static com.obs.services.internal.Constants.CommonHeaders.HASH_CRC64ECMA;
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObsObject;
import java.io.IOException;
import java.io.InputStream;
import java.util.Map;

public class GetObjectWithCRC64{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Download an object.
            String exampleBucket = "examplebucket";
            String exampleObjectKey = "objectKey";
            GetObjectRequest getObjectRequest = new GetObjectRequest(exampleBucket, exampleObjectKey);
            ObsObject obsObject = obsClient.getObject(getObjectRequest);
            InputStream inputStream = obsObject.getObjectContent();
            System.out.println("Server returned crc64 string:" + obsObject.getMetadata().getCrc64());
            System.out.println("GetObject successfully");
        } catch (ObsException e) {
            System.out.println("GetObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            // In an abnormal scenario, print all headers, which may contain error information.
            Map<String, String> headers = e.getResponseHeaders();
            if (headers != null) {
                for (Map.Entry<String, String> header : headers.entrySet()) {
                    System.out.println(header.getKey() + ":" + header.getValue());
                }
            }
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("GetObject failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```



```
    }  
  }  
}
```

Sample Code 2: Downloading an Object - Range-Based

The following code shows how to obtain the CRC64 value when downloading an object by specifying a range.

```
import static com.obs.services.internal.Constants.CommonHeaders.HASH_CRC64ECMA;  
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.GetObjectRequest;  
import com.obs.services.model.ObsObject;  
import java.io.IOException;  
import java.io.InputStream;  
import java.util.Map;  
  
public class GetObjectWithCRC64{  
    public static void main(String[] args) {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using  
        // hard coding may result in leakage.  
        // Obtain an AK/SK pair on the management console.  
        String ak = System.getenv("ACCESS_KEY_ID");  
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");  
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are  
        // advised not to use hard coding, which may result in information leakage.  
        // Obtain an AK/SK pair and a security token using environment variables or import them in other  
        // ways.  
        // String securityToken = System.getenv("SECURITY_TOKEN");  
        // Enter the endpoint corresponding to the region where the bucket is located.  
        String endPoint = "https://your-endpoint";  
        // Obtain an endpoint using environment variables or import it in other ways.  
        //String endPoint = System.getenv("ENDPOINT");  
  
        // Create an ObsClient instance.  
        // Use the permanent AK/SK pair to initialize the client.  
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
        // Use the temporary AK/SK pair and security token to initialize the client.  
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);  
  
        try {  
            // Download an object.  
            String exampleBucket = "examplebucket";  
            String exampleObjectKey = "objectKey";  
            GetObjectRequest getObjectRequest = new GetObjectRequest(exampleBucket, exampleObjectKey);  
            // Set the start point and end point.  
            getObjectRequest.setRangeStart(0);  
            getObjectRequest.setRangeEnd(1000);  
            ObsObject obsObject = obsClient.getObject(getObjectRequest);  
            InputStream inputStream = obsObject.getObjectContent();  
            System.out.println("Server returned crc64 string:" + obsObject.getMetadata().getCrc64());  
            System.out.println("GetObject successfully");  
        } catch (ObsException e) {  
            System.out.println("GetObject failed");  
            // Request failed. Print the HTTP status code.  
            System.out.println("HTTP Code:" + e.getResponseCode());  
            // Request failed. Print the server-side error code.  
            System.out.println("Error Code:" + e.getErrorCode());  
            // Request failed. Print the error details.  
            System.out.println("Error Message:" + e.getErrorMessage());  
            // Request failed. Print the request ID.  
            System.out.println("Request ID:" + e.getErrorRequestId());  
            System.out.println("Host ID:" + e.getErrorHostId());  
            // In an abnormal scenario, print all headers, which may contain error information.  
            Map<String, String> headers = e.getResponseHeaders();  
            if (headers != null) {  
                for (Map.Entry<String, String> header : headers.entrySet()) {  
                    System.out.println(header.getKey() + ":" + header.getValue());  
                }  
            }  
        }  
    }  
}
```

```
    }  
    }  
    e.printStackTrace();  
} catch (Exception e) {  
    System.out.println("GetObject failed");  
    // Print other error information.  
    e.printStackTrace();  
}  
}  
}
```

Sample Code 3: Obtaining Object Metadata

The following code shows how to obtain the CRC64 value of an object:

```
import static com.obs.services.internal.Constants.CommonHeaders.HASH_CRC64ECMA;  
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.ObjectMetadata;  
  
import java.util.Map;  
  
public class GetObjectMetaWithCRC64 {  
    public static void main(String[] args) {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using  
        // hard coding may result in leakage.  
        // Obtain an AK/SK pair on the management console.  
        String ak = System.getenv("ACCESS_KEY_ID");  
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");  
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are  
        // advised not to use hard coding, which may result in information leakage.  
        // Obtain an AK/SK pair and a security token using environment variables or import them in other  
        // ways.  
        // String securityToken = System.getenv("SECURITY_TOKEN");  
        // Enter the endpoint corresponding to the region where the bucket is located.  
        String endPoint = "https://your-endpoint";  
        // Obtain an endpoint using environment variables or import it in other ways.  
        //String endPoint = System.getenv("ENDPOINT");  
  
        // Create an ObsClient instance.  
        // Use the permanent AK/SK pair to initialize the client.  
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
        // Use the temporary AK/SK pair and security token to initialize the client.  
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);  
  
        try {  
            // Obtain the object metadata.  
            String exampleBucket = "examplebucket";  
            String exampleObjectKey = "objectKey";  
            ObjectMetadata objectMetadata = obsClient.getObjectMetadata(exampleBucket,  
exampleObjectKey);  
            System.out.println(  
                "Server returned crc64 string:" + objectMetadata.getCrc64());  
            System.out.println("GetObject successfully");  
        } catch (ObsException e) {  
            System.out.println("GetObject failed");  
            // Request failed. Print the HTTP status code.  
            System.out.println("HTTP Code:" + e.getResponseCode());  
            // Request failed. Print the server-side error code.  
            System.out.println("Error Code:" + e.getErrorCode());  
            // Request failed. Print the error details.  
            System.out.println("Error Message:" + e.getErrorMessage());  
            // Request failed. Print the request ID.  
            System.out.println("Request ID:" + e.getErrorRequestId());  
            System.out.println("Host ID:" + e.getErrorHostId());  
            // In an abnormal scenario, print all headers, which may contain error information.  
            Map<String, String> headers = e.getResponseHeaders();  
            if (headers != null) {  
                for (Map.Entry<String, String> header : headers.entrySet()) {
```

```
        System.out.println(header.getKey() + ":" + header.getValue());
    }
}
e.printStackTrace();
} catch (Exception e) {
    System.out.println("GetObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Sample Code 4: Copying an Object

The following code shows how to obtain the CRC64 value when copying an object:

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CopyObjectRequest;
import com.obs.services.model.CopyObjectResult;

import java.util.Map;
public class CopyObjectWithCRC64{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Copy a file.
            String exampleSourceBucket = "examplebucket";
            String exampleDestinationBucket = "destinationexamplebucket";
            String exampleSourceObjectKey = "objectKey";
            String exampleDestinationObjectKey = "copyObjectKey";
            CopyObjectRequest copyObjectRequest =
                new CopyObjectRequest(
                    exampleSourceBucket,
                    exampleSourceObjectKey,
                    exampleDestinationBucket,
                    exampleDestinationObjectKey);
            CopyObjectResult copyObjectResult = obsClient.copyObject(copyObjectRequest);
            String crc64 = copyObjectResult.getCRC64();
            System.out.println("Server returned crc64 string:" + crc64);
            System.out.println("CopyObject successfully");
        } catch (ObsException e) {
            System.out.println("AppendObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
        }
    }
}
```

```
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
// In an abnormal scenario, print all headers, which may contain error information.
Map<String, String> headers = e.getResponseHeaders();
if (headers != null) {
    for (Map.Entry<String, String> header : headers.entrySet()) {
        System.out.println(header.getKey() + ":" + header.getValue());
    }
}
e.printStackTrace();
} catch (Exception e) {
    System.out.println("AppendObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

9 Object Download (SDK for Java)

9.1 Overview (SDK for Java)

OBS Java SDK provides abundant APIs for object download in the following methods:

- [Streaming Download](#)
- [Partial Download](#)
- [Resumable download](#)

Request Parameters

Parameter	Type	Mandatory (Yes/No)	Description
isEncodeHeaders	boolean	No	Whether to automatically encode the request header. The default value is true .
userHeaders	HashMap<String, String>	No	User's header field.
ifMatchTag	String	No	Returns the object content if the object's ETag is the same as this parameter value. Otherwise, an error is reported.
ifNoneMatchTag	String	No	Returns the object content if the object's ETag is different from this parameter value. Otherwise, an error is reported.
ifModifiedSince	Date	No	Returns the object if it has been modified since the specified time; otherwise, an error is returned.

Parameter	Type	Mandatory (Yes/No)	Description
ifUnmodifiedSince	Date	No	Returns the object if it has not been modified since the specified time; otherwise, an error is returned.
rangeStart	Long	No	Start byte for a partial download.
rangeEnd	Long	No	End byte for a partial download.
versionId	String	No	Version number.

9.2 Downloading an Object - Streaming (SDK for Java)

Function

When you call **ObsClient.getObject**, an instance of **ObsObject** will be returned. This instance contains the contents of the object such as its name, attributes, input stream, and which bucket it is stored in. You can perform operations on the object input stream to read the contents to a local file or memory.

Restrictions

- To download an object, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).
- Objects in the storage class can be downloaded only when they are restored.
- Object input streams obtained by **ObsObject.getObjectContent** must be closed explicitly. Otherwise, resource leakage occurs.

Method

obsClient.getObject([GetObjectRequest request](#))

Request Parameters

Table 9-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 9-2 .

Table 9-2 GetObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (.), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
rangeStart	Long	No	Explanation: Start position for object download. Value range: A non-negative integer. Default value: None
rangeEnd	Long	No	Explanation: End position for object download. Restrictions: The upper limit of the value is the object length minus 1, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
IfModifiedSince	Date	No	Explanation: The object is returned if it has been modified since the specified time; otherwise, an error is returned. Default value: None
IfUnmodifiedSince	Date	No	Explanation: The object is returned if it has not been modified since the specified time; otherwise, an error is returned. Default value: None
replaceMetadata	ObjectRepleace Metadata	No	Explanation: Additional information about the object. For details, see Table 9-3.
progressListener	ProgressListener	No	Explanation: Data transmission listener for obtaining the download progress. For details, see Table 9-4.

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 9-3 ObjectReplaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentLanguage	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Table 9-4 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-5 . Default value: None

Table 9-5 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-6 . Default value: None

Table 9-6 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 9-7 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 9-8 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 9-9 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: • The MD5 value of the object must be Base64 encoded. • If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
crc64	String	No	Explanation: The 64-bit CRC value calculated for the object based on the ECMA-182 standard. The CRC64 value is the unique identifier of the object content. It can be used to determine whether the object content is changed. If an object has different CRC64 values when being uploaded and downloaded, its content has been changed. CRC64 reflects changes to the contents of the object, not its metadata. Restrictions: - This parameter is returned when the CRC64 value was verified for the uploaded object, or when the CRC64 feature has been enabled for the bucket. - This parameter is not supported for POSIX or SFS objects. Value range: A 64-bit CRC value calculated based on the ECMA-182 standard Default value: None
storageClass	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 9-8. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 9-10 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.) - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 9-7.

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 9-9 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Code Examples

This example downloads **objectname** from **examplebucket** using streaming.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsObject;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
public class GetObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Download the object using streaming.
            ObsObject obsObject = obsClient.getObject("examplebucket", "objectname");
            // Read the object content.
            System.out.println("Object content:");
            InputStream input = obsObject.getObjectContent();
            byte[] b = new byte[1024];
            ByteArrayOutputStream bos = new ByteArrayOutputStream();
            int len;
            while ((len = input.read(b)) != -1) {
                bos.write(b, 0, len);
            }
            System.out.println("getObjectContent successfully");
            System.out.println(new String(bos.toByteArray()));
            bos.close();
            input.close();
        } catch (ObsException e) {
            System.out.println("getObjectContent failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
        }
    }
}
```

```
// Request failed. Print the server-side error code.
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObjectContent failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

9.3 Downloading an Object - Range-Based (SDK for Java)

Function

This API downloads partial data of an object by specifying a range. If the specified range is from 0 to 1,000, data from byte 0 to byte 1,000, 1,001 bytes in total, are returned. If the specified range is invalid, the whole object is returned.

Restrictions

- To download an object, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).
- Objects in the storage class can be downloaded only when they are restored.
- If the specified range is invalid (for example, the start or end position is set to a negative integer, or the start position is greater than the end position), the API throws an exception. If the specified range is valid and the end position is greater than the object size, the entire object is returned.

Method

obsClient.getObject([GetObjectRequest request](#))

Request Parameters

Table 9-11 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 9-12 .

Table 9-12 GetObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
rangeStart	Long	No	Explanation: Start position for object download. Value range: A non-negative integer. Default value: None
rangeEnd	Long	No	Explanation: End position for object download. Restrictions: The upper limit of the value is the object length minus 1, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
IfModifiedSince	Date	No	Explanation: The object is returned if it has been modified since the specified time; otherwise, an error is returned. Default value: None
IfUnmodifiedSince	Date	No	Explanation: The object is returned if it has not been modified since the specified time; otherwise, an error is returned. Default value: None
replaceMetadata	ObjectRepleace Metadata	No	Explanation: Additional information about the object. For details, see Table 9-13.
progressListener	ProgressListener	No	Explanation: Data transmission listener for obtaining the download progress. For details, see Table 9-14.

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 9-13 ObjectRepleaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentLanguage	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Table 9-14 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-15 . Default value: None

Table 9-15 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-16 . Default value: None

Table 9-16 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 9-17 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 9-18 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 9-19 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: • The MD5 value of the object must be Base64 encoded. • If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
crc64	String	No	Explanation: The 64-bit CRC value calculated for the object based on the ECMA-182 standard. The CRC64 value is the unique identifier of the object content. It can be used to determine whether the object content is changed. If an object has different CRC64 values when being uploaded and downloaded, its content has been changed. CRC64 reflects changes to the contents of the object, not its metadata. Restrictions: • This parameter is returned when the CRC64 value was verified for the uploaded object, or when the CRC64 feature has been enabled for the bucket. • The CRC64 value of the entire object is returned for a partial download. • This parameter is not supported for POSIX or SFS objects. Value range: A 64-bit CRC value calculated based on the ECMA-182 standard Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If this header is not set, the default storage class of the bucket is used as the storage class of the object. Value range: See Table 9-18. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 9-20 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.) - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 9-17.

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 9-19 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Code Examples

This example downloads part of object **objectname** in bucket **examplebucket** by specifying a range from **0l** to **1000l**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObsObject;
import java.io.InputStream;
public class GetObject002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Perform a partial download.
            GetObjectRequest request = new GetObjectRequest("examplebucket", "objectname");
            // Set the start point and end point.
            request.setRangeStart(0l);
            request.setRangeEnd(1000l);
            ObsObject obsObject = obsClient.getObject(request);
            // Read data.
            byte[] buf = new byte[1024];
            InputStream in = obsObject.getObjectContent();
            for (int n = 0; n != -1; ) {
                n = in.read(buf, 0, buf.length);
            }
            System.out.println("GetObject successfully");
            in.close();
        } catch (ObsException e) {
            System.out.println("GetObject failed");
            // Request failed. Print the HTTP status code.
        }
    }
}
```

```
System.out.println("HTTP Code:" + e.getResponseCode());  
// Request failed. Print the server-side error code.  
System.out.println("Error Code:" + e.getErrorCode());  
// Request failed. Print the error details.  
System.out.println("Error Message:" + e.getErrorMessage());  
// Request failed. Print the request ID.  
System.out.println("Request ID:" + e.getErrorRequestId());  
System.out.println("Host ID:" + e.getErrorHostId());  
e.printStackTrace();  
} catch (Exception e) {  
    System.out.println("GetObject failed");  
    // Print other error information.  
    e.printStackTrace();  
}  
}
```

9.4 Obtaining the Download Progress (SDK for Java)

Function

This API returns the download progress of a specified object.

You can call **GetObjectRequest.setProgressInterval** to obtain the download progress.

Restrictions

- To obtain the download progress, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).
- You can obtain the download progress when downloading an object in streaming, partial, or resumable mode.

Method

GetObjectRequest.setProgressListener([ProgressListener progressListener](#))

Request parameters

Table 9-21 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
progressListener	Progress Listener	Yes	Explanation: Data transmission listener used for obtaining the progress. For details, see Table 9-22 .

Table 9-22 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-23 .

Table 9-23 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-24 .

Table 9-24 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Code Examples

This example returns the progress of downloading object **objectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObsObject;
import com.obs.services.model.ProgressListener;
import com.obs.services.model.ProgressStatus;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
public class GetObject003 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Obtain the download progresses.
            GetObjectRequest request = new GetObjectRequest("examplebucket", "objectname");
            request.setProgressListener(
                new ProgressListener() {
                    @Override
                    public void progressChanged(ProgressStatus status) {
                        // Obtain the average download rate.
                        System.out.println("AverageSpeed:" + status.getAverageSpeed());
                        // Obtain the download progress in percentage.
                        System.out.println("TransferPercentage:" + status.getTransferPercentage());
                    }
                }
            );
            // Refresh the download progress each time 1 MB data is downloaded.
            request.setProgressInterval(1024 * 1024L);
            ObsObject obsObject = obsClient.getObject(request);
            // Read the object content.
            System.out.println("Object content:");
            InputStream input = obsObject.getObjectContent();
            byte[] b = new byte[1024];
            ByteArrayOutputStream bos = new ByteArrayOutputStream();
            int len;
            while ((len = input.read(b)) != -1) {
                bos.write(b, 0, len);
            }
            System.out.println("getObjectContent successfully");
            System.out.println(new String(bos.toByteArray()));
            bos.close();
            input.close();
        } catch (ObsException e) {
            System.out.println("getObjectContent failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
        }
    }
}
```



```
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("getObjectContent failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

9.5 Downloading an Object - Conditional (SDK for Java)

Function

This API returns the objects that meet one or more conditions. If there are no objects that meet the specified conditions, an error is returned.

Restrictions

- To download an object, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).
- Objects in the storage class can be downloaded only when they are restored.

Method

obsClient.getObject([GetObjectRequest request](#))

Request Parameters

Table 9-25 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 9-26 .

Table 9-26 GetObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
rangeStart	Long	No	Explanation: Start position for object download. Value range: A non-negative integer. Default value: None
rangeEnd	Long	No	Explanation: End position for object download. Restrictions: The upper limit of the value is the object length minus 1, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: 1. The ETag of an object is the MD5 value of the object. Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. 2. If a request includes If-Unmodified-Since or If-Match and the specified condition is not met, 412 Precondition Failed will be returned. Value range: The value must contain 32 characters. Default value: None
ifNoneMatchTag	String	No	Explanation: 1. Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. 2. If a request includes If-Modified-Since or If-None-Match, and the specified condition is not met, 304 Not Modified will be returned. Value range: The value must contain 32 characters. Default value: None
IfModifiedSince	Date	No	Explanation: 1. The object is returned if it has been modified since the specified time; otherwise, an error is returned. 2. If a request includes If-Modified-Since or If-None-Match, and the specified condition is not met, 304 Not Modified will be returned. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
IfUnmodifiedSince	Date	No	Explanation: 1. The object is returned if it has not been modified since the specified time; otherwise, an error is returned. 2. If a request includes If-Unmodified-Since or If-Match and the specified condition is not met, 412 Precondition Failed will be returned. Default value: None
replaceMetadata	ObjectReplaceMetadata	No	Explanation: Additional information about the object. For details, see Table 9-27.
progressListener	ProgressListener	No	Explanation: Data transmission listener for obtaining the progress. For details, see Table 9-28.
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 9-27 ObjectRepleaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Table 9-28 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-29 . Default value: None

Table 9-29 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-30 . Default value: None

Table 9-30 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 9-31 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 9-32 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 9-33 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If this header is not set, the default storage class of the bucket is used as the storage class of the object. Value range: See Table 9-32. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 9-34 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.) – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 9-31 .

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 9-33 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Code Examples

This example downloads **objectname** from **examplebucket** based on a specific condition. In this example, **objectname** can be downloaded only when it is modified after 00:00:00 on January 1, 2016.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObsObject;
import java.text.SimpleDateFormat;
public class GetObject004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Perform a conditional download.
            GetObjectRequest request = new GetObjectRequest("examplebucket", "objectname");
            request.setRangeStart(0l);
            request.setRangeEnd(1000l);
            request.setIfModifiedSince(new SimpleDateFormat("yyyy-MM-dd").parse("2016-01-01"));
            ObsObject obsObject = obsClient.getObject(request);
            System.out.println("getObject successfully");
            obsObject.getObjectContent().close();
        } catch (ObsException e) {
            System.out.println("getObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
```



```
        System.out.println("Error Message:" + e.getMessage());  
        // Request failed. Print the request ID.  
        System.out.println("Request ID:" + e.getErrorRequestId());  
        System.out.println("Host ID:" + e.getErrorHostId());  
        e.printStackTrace();  
    } catch (Exception e) {  
        System.out.println("getObject failed");  
        // Print other error information.  
        e.printStackTrace();  
    }  
}
```

9.6 Rewriting Response Headers (SDK for Java)

Function

This API rewrites the following HTTP/HTTPS response headers when downloading an object: **Content-Type**, **Content-Language**, **Expires**, **Cache-Control**, **Content-Disposition**, and **Content-Encoding**.

Restrictions

- To download an object, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).

Method

obsClient.getObject([GetObjectRequest request](#))

Request Parameters

Table 9-35 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 9-36 .

Table 9-36 GetObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
rangeStart	Long	No	Explanation: Start position for object download. Value range: A non-negative integer. Default value: None
rangeEnd	Long	No	Explanation: End position for object download. Restrictions: The upper limit of the value is the object length minus 1, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
IfModifiedSince	Date	No	Explanation: The object is returned if it has been modified since the specified time; otherwise, an error is returned. Default value: None
IfUnmodifiedSince	Date	No	Explanation: The object is returned if it has not been modified since the specified time; otherwise, an error is returned. Default value: None
replaceMetadata	ObjectRepleace Metadata	No	Explanation: Additional information about the object. For details, see Table 9-37 .
progressListener	ProgressListener	No	Explanation: Data transmission listener for obtaining the progress. For details, see Table 9-38 .

Table 9-37 ObjectRepleaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentLanguage	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Table 9-38 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-39 . Default value: None

Table 9-39 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-40 . Default value: None

Table 9-40 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 9-41 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 9-42 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 9-43 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Object size. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If this header is not set, the default storage class of the bucket is used as the storage class of the object. Value range: See Table 9-42. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 9-44 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.) – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 9-41 .

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 9-43 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Code Examples

This example rewrites response headers and downloads object **objectname** from bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObjectReplaceMetadata;
import com.obs.services.model.ObsObject;
public class GetObject005 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Rewrite the response headers.
            GetObjectRequest request = new GetObjectRequest("examplebucket", "objectname");
            ObjectReplaceMetadata replaceMetadata = new ObjectReplaceMetadata();
            replaceMetadata.setContentType("image/jpeg");
            request.setReplaceMetadata(replaceMetadata);
            ObsObject obsObject = obsClient.getObject(request);
            System.out.println(obsObject.getMetadata().getContentType());
            System.out.println("getObject successfully");
            obsObject.getObjectContent().close();
        } catch (ObsException e) {
            System.out.println("getObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
        }
    }
}
```

```
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

9.7 Obtaining User-defined Metadata (SDK for Java)

Function

Object metadata is a set of key-value pairs that describe the object and is used for object management.

This API returns the user-defined object metadata after the object is successfully downloaded.

Restrictions

- To obtain object metadata, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).

Method

obsClient.getObject([GetObjectRequest request](#))

Request Parameters

Table 9-45 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 9-46 .

Table 9-46 GetObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
rangeStart	Long	No	Explanation: Start position for object download. Value range: A non-negative integer. Default value: None
rangeEnd	Long	No	Explanation: End position for object download. Restrictions: The upper limit of the value is the object length minus 1, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
IfModifiedSince	Date	No	Explanation: The object is returned if it has been modified since the specified time; otherwise, an error is returned. Default value: None
IfUnmodifiedSince	Date	No	Explanation: The object is returned if it has not been modified since the specified time; otherwise, an error is returned. Default value: None
replaceMetadata	ObjectRepleace Metadata	No	Explanation: Additional information about the object. For details, see Table 9-47.
progressListener	ProgressListener	No	Explanation: Data transmission listener for obtaining the progress. For details, see Table 9-48.

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 9-47 ObjectRepleaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentLanguage	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Table 9-48 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-49 . Default value: None

Table 9-49 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-50 . Default value: None

Table 9-50 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 9-51 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 9-52 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 9-53 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If this header is not set, the default storage class of the bucket is used as the storage class of the object. Value range: See Table 9-52. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 9-54 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.) - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 9-51.

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 9-53 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Code Examples

This example downloads object **objectname** from bucket **examplebucket** and returns the user-defined object metadata.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.ObsObject;
import com.obs.services.model.PutObjectRequest;
public class GetObject006 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Obtain the user-defined metadata.
            // Upload the object and customize the metadata.
            PutObjectRequest request = new PutObjectRequest("examplebucket", "objectname");
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.addUserMetadata("property", "property-value");
            request.setMetadata(metadata);
            obsClient.putObject(request);
            // Download the object to obtain the object's user-defined metadata.
            GetObjectRequest request1 = new GetObjectRequest("examplebucket", "objectname");
            ObsObject obsObject = obsClient.getObject(request1);
            System.out.println("getObject successfully");
            // Get the object metadata.
            System.out.println(obsObject.getMetadata().getContentType());
            // Obtain the user-defined metadata named property.
            System.out.println(obsObject.getMetadata().getUserMetadata("property"));
            // Obtain all user-defined metadata.
```

```
        System.out.println("allUserMetadata:" + metadata.getAllMetadata());
        obsObject.getObjectContent().close();
    } catch (ObsException e) {
        System.out.println("getObject failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessages());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("getObject failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

9.8 Restoring a Cold Object (SDK for Java)

Function

Cold objects must first be restored before you can download them. Cold objects can be restored in either of the following ways.

Option	Description	Value in OBS Java SDK
Expedited	Data can be restored within 1 to 5 minutes.	RestoreTierEnum.EXPEDITED
Standard	Data can be restored within 3 to 5 hours. This is the default option.	RestoreTierEnum.STANDARD

Restrictions

- To restore object, you must be the bucket owner or have the required permission (**obs:object:RestoreObject** in IAM or **RestoreObject** in a bucket policy.)
- To prolong the validity period of the data restored, you can repeatedly restore the data, but you will be billed for each restore. After a second restore, the validity period of Standard object copies will be prolonged, and you need to pay for storing these copies during the prolonged period.
- The object specified in **ObsClient.restoreObject** must be in the Cold storage class. Otherwise, an exception will be thrown when you call this API.

Method

obsClient.restoreObject([RestoreObjectRequest request](#))

Request Parameters

Table 9-55 List of request parameters

Parameter	Type	Mandator y (Yes/No)	Description
request	RestoreObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 9-56 .

Table 9-56 RestoreObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (.), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Default value: None. If this parameter is left blank, the latest version of the object is specified.
days	int	Yes	Explanation: After an object is restored, a Standard copy of it is generated. This parameter specifies how long the Standard copy can be retained, that is, the validity period of the restored object. Restrictions: The value must be a positive integer. Value range: The value ranges from 1 to 30, in days. Default value: None
tier	RestoreTierEnum	No	Explanation: The restore option, which indicates the time spent on restoring the object. Value range: For details, see Table 9-57. Default value: Standard

Table 9-57 RestoreTierEnum

Constant	Default Value	Description
EXPEDITED	Expedited	Objects can be restored at an expedited speed within 1 to 5 minutes.
STANDARD	Standard	Objects can be restored at a standard speed within 3 to 5 hours.

Responses

Table 9-58 RestoreObjectStatus

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example restores object **objectname** from the Cold storage class using **ObsClient.restoreObject** and downloads it using **ObsClient.getObject**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsObject;
import com.obs.services.model.RestoreObjectRequest;
import com.obs.services.model.RestoreTierEnum;
public class GetObject007 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
```

```
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Restore the Cold object.
    RestoreObjectRequest request = new RestoreObjectRequest();
    request.setBucketName("examplebucket");
    request.setObjectKey("objectname");
    request.setDays(1);
    request.setRestoreTier(RestoreTierEnum.EXPEDITED);
    obsClient.restoreObject(request);
    // Wait for the object to be restored.
    Thread.sleep(60 * 6 * 1000);
    // Download the object.
    ObsObject obsObject = obsClient.getObject("examplebucket", "objectname");
    System.out.println("getObject successfully");
    obsObject.getObjectContent().close();
} catch (ObsException e) {
    System.out.println("getObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

9.9 Downloading an Object - Resumable (SDK for Java)

Function

1. Downloading large files often fails due to an unstable network or program breakdown. It is a waste of resources to download files again. Moreover, the restarted download may still fail due to an unstable network. To resolve such issues, the resumable download API splits the file to be downloaded into multiple parts and downloads them separately. The download result of each part is recorded in a checkpoint file in real time. Only when all parts are downloaded is a message indicating the download is successful returned. If any parts fail to be downloaded, a message is returned telling you to call the API again to download the failed parts. Since the checkpoint file contains the

progress of all parts, it helps you avoid downloading all parts in re-downloads, so that you can enjoy a cost-effective, efficient download.

2. The resumable download API is an encapsulated and enhanced version of the API for partial download. For details about the partial download, see [Downloading an Object - Range-Based \(SDK for Java\)](#).
3. This API saves resources and improves efficiency upon the re-download, and speeds up the download process by concurrently downloading parts. You do not need to worry about internal service details, such as the creation and deletion of checkpoint files, division of objects, or concurrent downloads of parts.

Restrictions

- To download an object, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).

Method

obsClient.downloadFile([DownloadFileRequest request](#))

Request Parameters

Table 9-59 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	DownloadFileRequest	Yes	Explanation: Request parameters for downloading a large object. For details, see Table 9-60 .

Table 9-60 DownloadFileRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
downloadFile	String	No	Explanation: Full path of the local directory to which the object is downloaded. Default value: If the value of this parameter is left blank, the downloaded object is saved in the directory where the program is executed.
partSize	long	No	Explanation: Part size. Value range: The value ranges from 100 KB to 5 GB, in bytes. Default value: 9 MB
taskNum	int	No	Explanation: Maximum number of parts that can be downloaded concurrently in a multipart download Value range: The value must be greater than 0, but not exceed the file size divided by the part size, rounded up. Default value: 1, indicating concurrent downloads are not used.
enableCheckpoint	boolean	No	Explanation: Whether to enable the resumable mode. Value range: true: The resumable mode is enabled. false: The resumable mode is disabled. Default value: false

Parameter	Type	Mandatory (Yes/No)	Description
checkpointFile	String	No	Explanation: Path of a file generated for recording the progress of a resumable download. The file contains the information about parts and progress. Restrictions: This parameter is valid only in the resumable mode. Default value: If this parameter is left blank, the checkpoint file will be saved in the current directory.
enableCheckSum	boolean	No	Explanation: Whether to verify the file to download. If this function is enabled, before each task restarts, the system verifies whether the file to download is the one used during task initialization. Restrictions: This parameter is valid only in the resumable mode. Value range: true: The file is verified. false: The file is not verified. Default value: false
versionId	String	No	Explanation: Object version ID, for example, G001117FCE89978B0000401205D5DC9. Value range: The value must contain 32 characters. Default value: None. If this parameter is left blank, the latest version of the object is obtained.

Parameter	Type	Man dato ry (Yes /No)	Description
encodeHeaders	boolean	No	Explanation: Whether to automatically decode the response headers. Default value: None
needCalculateCRC64	boolean	No	Explanation: Whether to automatically calculate and verify the CRC64 value of the downloaded data. Restrictions: This parameter is not supported for POSIX or SFS objects. Value range: true: The SDK calculates and verifies CRC64. false: CRC64 is not calculated. Default value: false
ifModifiedSince	Date	No	Explanation: Returns the object if it has been modified since the specified time; otherwise, an error is returned. Default value: None
ifUnmodifiedSince	Date	No	Explanation: Returns the object if it has not been modified since the specified time; otherwise, an error is returned. Default value: None
ifMatchTag	String	No	Explanation: Returns the object content if the object's ETag is the same as the value of this parameter. Otherwise, an error is reported. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifNoneMatchTag	String	No	Explanation: Returns the object content if the object's ETag is different from the value of this parameter. Otherwise, an error is reported. Default value: None
progressListener	Progress Listener	No	Explanation: Data transmission listener for obtaining the download progress. For details, see Table 9-61 .
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 9-61 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 9-62 . Default value: None

Table 9-62 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 9-63 . Default value: None

Table 9-63 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 9-64 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 9-65 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If this header is not set, the default storage class of the bucket is used as the storage class of the object. Value range: See Table 9-64. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map , the String key indicates the name of the user-defined metadata that starts with x-obs-meta- , and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata . For details, see Obtaining Object Metadata (SDK for Java) . Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 9-66 DownloadFileResult

Parameter	Type	Description
objectMetadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 9-65 .

Code Examples

This example downloads object **objectname** from bucket **examplebucket** to **localfile** using the resumable download.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.DownloadFileRequest;  
import com.obs.services.model.DownloadFileResult;  
public class DownloadFile001 {  
    public static void main(String[] args) {
```



```
// Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    DownloadFileRequest request = new DownloadFileRequest("examplebucket", "objectname");
    // Set the local path the object will be downloaded to.
    request.setDownloadFile("localfile");
    // Set the maximum number of parts that can be concurrently downloaded.
    request.setTaskNum(5);
    // Set the part size to 10 MB.
    request.setPartSize(10 * 1024 * 1024);
    // Enable the resumable upload.
    request.setEnableCheckpoint(true);
    // Perform a resumable download.
    DownloadFileResult result = obsClient.downloadFile(request);
    System.out.println("downloadFile successfully");
    System.out.println("Etag:" + result.getObjectMetadata().getEtag());
} catch (ObsException e) {
    System.out.println("downloadFile failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("downloadFile failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

10 Object Management (SDK for Java)

10.1 Configuring Object Metadata (SDK for Java)

Function

Object metadata is a set of name-value pairs that describe the object and is used for object management.

Currently, only the system-defined metadata is supported.

There are two kinds of metadata: system-controlled metadata and user-controlled metadata. The kind of metadata like **Last-Modified** is controlled by the system and cannot be modified. However, the kind of metadata configured for objects such as **ContentLanguage** can be modified by calling APIs.

This API sends a HEAD request to configure the metadata of an object in a specified bucket.

Restrictions

- To configure object metadata, you must be the bucket owner or have the required permission (**obs:object:ModifyObjectMetaData** in IAM or **ModifyObjectMetaData** in a bucket policy).
- An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total.
- Currently, metadata names only support ASCII characters. Non-ASCII characters must be Base64-encoded.
- If versioning is enabled for a bucket, you can set metadata for objects of the latest version, but cannot set metadata for noncurrent objects.
- You cannot set metadata for objects.
- If a folder or directory is an actual object in an object bucket, their metadata can be modified. If a directory is a simulated directory, for example, **test/** in the **test/test1.txt** object, the metadata of this simulated directory cannot be modified.

Method

obsClient.setObjectMetadata([SetObjectMetadataRequest request](#))

Request Parameters

Table 10-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetObjectMetadataRequest	Yes	Explanation: Request parameters for obtaining object metadata. For details, see Table 10-2 .

Table 10-2 SetObjectMetadataRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
storageClass	StorageClassEnum	No	Explanation: Storage class of the object. Value range: See Table 10-3 . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
removeUnset	boolean	No	Explanation: Whether to delete the previously configured metadata if the request is successful. Value range: true: If the API is successfully invoked, the previous metadata will be deleted. Standard metadata is not affected. For user-defined metadata, only the new metadata is retained. false: The previous metadata will not be deleted even if the API is successfully invoked. Standard metadata is not affected. For user-defined metadata, only the new metadata is retained. Default value: false

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, String>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key and value indicate the name and value of the user-defined metadata respectively. After the metadata is defined, when you later download the object or query its metadata, the user-defined metadata is returned in the response. Restrictions: • The user-defined metadata cannot exceed 8 KB in total. To measure the size, calculate the sum of bytes of all UTF-8 encoded keys and values. • The user-defined metadata keys are case insensitive, because OBS converts the uppercase keys to lowercase for storage. However, the metadata values are case sensitive. • Both user-defined metadata keys and their values must conform to US-ASCII standards. If non-ASCII or unrecognizable characters are necessary, they must be encoded and decoded in URL or Base64 on the client, because the server side does not perform any decoding. Default value: None
userHeaders	HashMap<String, String>	No	Explanation: User header list. In HashMap, the String key and value indicate the name and value of the user header field respectively. The SDK does not process the userHeaders and instead transparently transmits it to the server for later use. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true
replaceMetadata	ObjectReplaceMetadata	No	Explanation: It specifies the metadata returned when the object is downloaded. For details, see Table 10-4.
websiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ OBS obtains the specified value from the header and stores it in the object metadata WebsiteRedirectLocation. Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None

Table 10-3 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class.
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Table 10-4 ObjectRepleaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Responses

Table 10-5 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentEncoding	String	No	Explanation: It specifies the content encoding format when an object is downloaded. Default value: None
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
cacheControl	String	No	Explanation: It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None
contentMD5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClass	StorageClassEnum	No	Explanation: Object storage class. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 10-3 . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map , the String key indicates the name of the user-defined metadata that starts with x-obs-meta- , and the Object value indicates the value of the user-defined metadata. Default value: None
statusCode	int	This parameter is automatically set in the response.	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	This parameter is automatically set in the response.	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Example: Configuring Metadata for the Object

This example configures metadata for object **objectname** by specifying the user-defined metadata and using **userHeaders** that are transparently transmitted.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.SetObjectMetadataRequest;
import java.util.HashMap;
public class SetObjectMetadata001 {
    public static void main(String[] args) {
```

```
// Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Configure metadata for the object.
    SetObjectMetadataRequest request = new SetObjectMetadataRequest("examplebucket",
"objectname");
    // Standard HTTP headers of an object
    request.setContentType("ContentType");
    request.setExpires("Expires");
    // During the user-defined metadata setting, the SDK automatically adds x-obs-meta-.
    request.addUserMetadata("property1", "property-value1");
    HashMap<String, String> userHeaders = new HashMap<>();
    // userHeaders set by the SDK are transparently transmitted. They are not processed so they can be
used more flexibly later. If a userHeader needs to be displayed as user-defined metadata on the server, add
x-obs-meta- before the userHeader for the server to identify.
    userHeaders.put("property2", "property-value2");
    userHeaders.put("x-obs-meta-property3", "property-value3");
    request.setUserHeaders(userHeaders);
    ObjectMetadata metadata = obsClient.setObjectMetadata(request);
    System.out.println("setObjectMetadata successfully");
    System.out.println("UserMetadata:" + metadata.getUserMetadata("property1"));
} catch (ObsException e) {
    System.out.println("setObjectMetadata failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setObjectMetadata failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

After you run the sample code above and configure the user-defined metadata **property1** and user headers **property2** and **x-obs-meta-property3**, the HTTP header in the packet capture is as follows:

```
PUT /?metadata HTTP/1.1
x-obs-meta-property1: property-value1
x-obs-metadata-directive: REPLACE_NEW
property2: property-value2
x-obs-meta-property3: property-value3
```

Code Example: Deleting Metadata for the Object

If you want to delete some metadata by calling the API for setting object metadata, configure the metadata to be retained and set **SetObjectMetadataRequest.setRemoveUnset** to **true**. Then, once the API is called, the data whose metadata is not configured will be deleted. Therefore, exercise caution when using **SetObjectMetadataRequest.setRemoveUnset(true)**.

This example deletes the user-defined metadata of object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.SetObjectMetadataRequest;
import java.util.HashMap;

public class SetObjectMetadata002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            SetObjectMetadataRequest request = new SetObjectMetadataRequest();
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname");
            request.setRemoveUnset(true);
            HashMap<String, String> userHeaders = new HashMap<>();
            // After request.setRemoveUnset is set to true, the server saves metadata property1 and deletes
            // the user-defined metadata saved before.
            userHeaders.put("x-obs-meta-property1", "property-value1");
            request.setUserHeaders(userHeaders);
            obsClient.setObjectMetadata(request);
            System.out.println("setObjectMetadata successfully");
        } catch (ObsException e) {
            System.out.println("setObjectMetadata failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
        }
    }
}
```



```
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("setObjectMetadata failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

10.2 Obtaining Object Metadata (SDK for Java)

Function

Object metadata is a set of name-value pairs that describe the object and is used for object management.

Users with the READ permission on objects can send HEAD requests to retrieve metadata from objects. The metadata of the objects is included in the response.

This operation supports server-side encryption.

Restrictions

- To obtain object metadata, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).
- You can also call **metadata.getOriginalHeaders** to obtain information about all original response headers.

Method

obsClient.getObjectMetadata([GetObjectMetadataRequest request](#))

Request Parameters

Table 10-6 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectMetadataRequest	Yes	Explanation: Request parameters for obtaining object metadata. For details, see Table 10-7 .

Table 10-7 GetObjectMetadataRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
userHeaders	HashMap<String, String>	No	Explanation: User header list. In HashMap, the String key and value indicate the name and value of the user header field respectively. The SDK does not process the userHeaders and instead transparently transmits it to the server for later use. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Constant	Default Value
AES256	AES256

Constant	Default Value
KMS	kms
AES256	AES256

Responses

Table 10-8 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: • The MD5 value of the object must be Base64 encoded. • If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
crc64	String	No	Explanation: The 64-bit CRC value calculated for the object based on the ECMA-182 standard. The CRC64 value is the unique identifier of the object content. It can be used to determine whether the object content is changed. If an object has different CRC64 values when being uploaded and downloaded, its content has been changed. CRC64 reflects changes to the contents of the object, not its metadata. Restrictions: - This parameter is returned when the CRC64 value was verified for the uploaded object, or when the CRC64 feature has been enabled for the bucket. - This parameter is not supported for POSIX or SFS objects. Value range: A 64-bit CRC value calculated based on the ECMA-182 standard Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: - If the storage class is Standard, leave this parameter blank. - See Table 10-9. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 10-9 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Code Examples

This example returns the metadata of object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
public class GetObjectMetadata001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Obtain the object metadata.
            ObjectMetadata metadata = obsClient.getObjectMetadata("examplebucket", "objectname");
            System.out.println("getObjectMetadata successfully");
            System.out.println("ContentType:" + metadata.getContentType());
            System.out.println("ContentLength:" + metadata.getContentLength());
            // Obtain the user-defined metadata.
            System.out.println("UserMetadata:" + metadata.getUserMetadata("property"));
            // Obtain all original response headers.
            System.out.println("OriginalHeaders:" + metadata.getOriginalHeaders());
        } catch (ObsException e) {
            System.out.println("getObjectMetadata failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("getObjectMetadata failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

10.3 Configuring an Object ACL (SDK for Java)

Function

Access control lists (ACLs) allow resource owners to grant other accounts the permissions to access resources. By default, only the resource owner has full control over resources when a bucket or object is created. That is, the bucket creator has full control over the bucket, and the object uploader has full control over the object. Other accounts do not have the permissions to access resources. If resource owners want to grant other accounts the read and write permissions on resources, they can use ACLs. ACLs grant permissions to accounts. After an account is granted permissions, both the account and its IAM users can access the resources. If an object is encrypted with SSE-KMS, the ACL configured for it is not in effect in the cross-tenant case.

An object ACL can be configured in any of the following ways:

1. [Code Example: Setting a Pre-defined ACL When Uploading the Object](#)
2. [Code Example: Setting a Pre-defined ACL for the Object](#)
3. [Code Example: Setting an ACL for the Object Directly](#)

Restrictions

- To configure an object ACL, you must be the bucket owner or have the required permission (**obs:object:PutObjectAcl** in IAM or **PutObjectAcl** in a bucket policy).
- An object can have a maximum of 100 rules in its ACL.

Method

obsClient.setObjectAcl([SetObjectAclRequest request](#))

Request Parameters

Table 10-10 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetObjectAclRequest	Yes	Explanation: Request parameters for setting an object ACL. For details, see Table 10-11 .

Table 10-11 SetObjectAclRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
acl	AccessControlList	Yes	Explanation: An ACL specified for the object. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 10-12 for the available options. - To use a user-defined ACL, see Table 10-13 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE

Table 10-12 Pre-defined ACL

Constant	Description
AccessControllist.REST_CAN_NED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CAN_NED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN_NED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 10-13 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Description
owner	Owner	Yes	Explanation: Bucket owner information. For details, see Table 10-14 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 10-15 .

Table 10-14 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-15 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 10-16 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 10-19 . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 10-16 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 10-17 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 10-18 . Default value: None

Table 10-17 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-18 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 10-19 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Responses

Table 10-20 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Example: Setting a Pre-defined ACL When Uploading the Object

This example uploads **localfile** to bucket **examplebucket** as object **objectname**, and specifies a pre-defined ACL during the upload.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject012 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
```

```
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Specify a pre-defined ACL during object upload.
    PutObjectRequest request = new PutObjectRequest();
    request.setBucketName("examplebucket");
    request.setObjectKey("objectname");
    request.setFile(new File("localfile"));
    // Set the object ACL to private read and write.
    request.setAcl(AccessControlList.REST_CANNED_PRIVATE);
    obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Setting a Pre-defined ACL for the Object

This example specifies a pre-defined ACL of private read and write for object **objectname** in **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
public class SetObjectAcl001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set a predefined ACL for the object.
            // Set the object ACL to private read and write.
            obsClient.setObjectAcl("examplebucket", "objectname", AccessControlList.REST_CANNED_PRIVATE);
            System.out.println("setObjectAcl successfully");
        } catch (ObsException e) {
            //
        }
    }
}
```

```
        System.out.println("setObjectAcl failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("setObjectAcl failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

Code Example: Setting an ACL for the Object Directly

This example configures an ACL for object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.CanonicalGrantee;
import com.obs.services.model.GroupGrantee;
import com.obs.services.model.Owner;
import com.obs.services.model.Permission;
public class SetObjectAcl002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            AccessControlList acl = new AccessControlList();
            Owner owner = new Owner();
            owner.setId("ownerid");
            // (Mandatory) Owner ID
            acl.setOwner(owner);
            //Retain the owner's full control permissions. (Note that if this permission is not set, the owner
            // does not have the access permission.)
            acl.grantPermission(new CanonicalGrantee("ownerid"), Permission.PERMISSION_FULL_CONTROL);
            // Grant the full control permission to a specified user.
            acl.grantPermission(new CanonicalGrantee("userid"), Permission.PERMISSION_FULL_CONTROL);
            // Grant the read permission to all users.
            acl.grantPermission(GroupGrantee.ALL_USERS, Permission.PERMISSION_READ);
            obsClient.setObjectAcl("examplebucket", "objectname", acl);
            System.out.println("setObjectAcl successfully");
            System.out.println(acl);
        } catch (ObsException e) {
            System.out.println("setObjectAcl failed");
            e.printStackTrace();
        }
    }
}
```

```
    } catch (ObsException e) {
        System.out.println("setObjectAcl failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("setObjectAcl failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

10.4 Obtaining an Object ACL (SDK for Java)

Function

Access control lists (ACLs) allow resource owners to grant other accounts the permissions to access resources. By default, only the resource owner has full control over resources when a bucket or object is created. That is, the bucket creator has full control over the bucket, and the object uploader has full control over the object. Other accounts do not have the permissions to access resources. If resource owners want to grant other accounts the read and write permissions on resources, they can use ACLs. ACLs grant permissions to accounts. After an account is granted permissions, both the account and its IAM users can access the resources. If an object is encrypted with SSE-KMS, the ACL configured for it is not in effect in the cross-tenant case.

This API obtains the ACL of an object in a specified bucket.

Restrictions

- To obtain an object ACL, you must be the bucket owner or have the required permission (**obs:object:GetObjectAcl** in IAM or **GetObjectAcl** in a bucket policy).

Method

obsClient.getObjectAcl([GetObjectAclRequest request](#))

Request Parameters

Table 10-21 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectAclRequest	Yes	Explanation: Request parameters for obtaining an object ACL. For details, see Table 10-22 .

Table 10-22 GetObjectAclRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None

Responses

Table 10-23 AccessControlList

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
owner	Owner	Explanation: Object owner. For details, see Table 10-28 .
grants	Set< Grant And Permission >	Explanation: Grantee information. For details, see Table 10-24 .
delivered	boolean	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 10-24 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 10-25 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 10-29 . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 10-25 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 10-26 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 10-27 . Default value: None

Table 10-26 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-27 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 10-28 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-29 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.

Constant	Default Value	Description
PERMISSION_READ_ACL	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACL	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Code Examples

This example returns the ACL information of object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
public class GetObjectAcl001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
```



```
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Obtain the object ACL.
    AccessControlList acl = obsClient.getObjectAcl("examplebucket", "objectname");
    System.out.println("getObjectAcl successfully");
    System.out.println(acl);
} catch (ObsException e) {
    System.out.println("getObjectAcl failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObjectAcl failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

10.5 Listing Objects (SDK for Java)

Function

This API lists some or all of the objects in a bucket. You can use parameters such as the prefix, number of returned objects, and start position to list objects that meet specified criteria. Returned objects are listed in alphabetical order by object name.

Restrictions

- To list objects in a bucket, you must be the bucket owner or have the required permission (**obs:bucket:ListBucket** in IAM or **ListBucket** in a bucket policy).
- A maximum of 1,000 object versions can be returned in a single request. If a bucket contains more than 1,000 object versions, you can use [pagination](#) to list all versions. If **ListVersionsResult.isTruncated** is **true**, only part of object versions are returned. In this case, you can use **ListVersionsResult.getNextKeyMarker** and **ListVersionsResult.getNextVersionIdMarker** to obtain the start position for the next listing.

Method

obsClient.listObjects([ListObjectsRequest request](#))

Request Parameters

Table 10-30 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	ListObjectsRequest	Yes	Explanation: Request parameters for listing the objects in a bucket. For details, see Table 10-31 .

Table 10-31 ListObjectsRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
prefix	String	No	Explanation: Name prefix that the objects to be listed must contain. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you specify logs/ as the prefix, logs/day1, logs/day2, and logs/day3 will be returned. If this parameter is left blank and there are no other filtering criteria, all objects in the bucket will be returned. Value range: The value must contain 1 to 1,024 characters. Default value: None
marker	String	No	Explanation: Name of the object to start with when listing objects in a bucket. All objects following this object are listed in alphabetical order by object name. Assume that you have the following objects: test/a, test/b, test/c, and test/d. If you specify test/b as the marker, test/c and test/d are returned. Restrictions: This parameter is only available for listing objects with a single version. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
maxKeys	int	No	Explanation: The maximum number of objects returned in the response in alphabetical order. Value range: The value ranges from 1 to 1,000. If the specified value is beyond this range, only 1,000 objects are returned. Default value: 1000

Parameter	Type	Mandatory (Yes/No)	Description
delimiter	String	No	Explanation: Object names are grouped by this parameter, which is often used with Prefix. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one CommonPrefixes. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one CommonPrefixes. Assume that a bucket has objects abcd, abcde, and bbcde in it. If delimiter is set to d and prefix is set to a, files abcd and abcde are grouped into a CommonPrefixes with abcd as the prefix. If only delimiter is set to d, files abcd and abcde are grouped into a CommonPrefixes with abcd as the prefix, and bbcde is grouped separately into another CommonPrefixes with bbcd as the prefix. For a PFS, if this parameter is not specified, all the contents in the current directory are recursively listed by default, and subdirectories are also listed. In big data scenarios, PFSs usually have multiple directory levels and each directory level has a large number of files. In such case, you are advised to configure [delimiter=/] to list the contents in the current directory but exclude the contents in subdirectories, thereby accelerating the listing. Value range: The value must contain 1 to 1,024 characters. Default value: None

Responses

Table 10-32 ObjectListing

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Description
prefix	String	Explanation: Object name prefix. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you specify logs/ as the prefix, logs/day1, logs/day2, and logs/day3 will be returned. If this parameter is left blank and there are no other filtering criteria, all objects in the bucket will be returned. Value range: The value must contain 1 to 1,024 characters. Default value: None
objectSummaries	List<ObsObject>	Explanation: List of objects in the bucket. For details, see Table 10-33.
commonPrefixes	List<String>	Explanation: List of object name prefixes grouped according to the delimiter parameter (if specified) Value range: The value must contain 1 to 1,024 characters. Default value: None
truncated	boolean	Explanation: Whether all objects are returned in the response. A maximum of 1,000 objects can be listed at a time. If the number of objects is greater than 1,000, the objects beyond 1,000 cannot be returned. Value range: true: Not all objects are returned. false: All objects are returned. Default value: None

Parameter	Type	Description
marker	String	Explanation: Where in the bucket the listing starts from. All objects following the object specified by this parameter are listed in alphabetical order by object name. Assume you have the following objects: test/a, test/b, test/c, and test/d. If you specify test/b as the marker, test/c and test/d are returned. Restrictions: This parameter is only available for listing objects with a single version. Value range: The value must contain 1 to 1,024 characters. Default value: None
maxKeys	int	Explanation: The maximum number of objects returned in the response in alphabetical order. Value range: The value ranges from 1 to 1,000. If the specified value is beyond this range, only 1,000 objects are returned. Default value: 1000

Parameter	Type	Description
delimiter	String	Explanation: Object names are grouped by this parameter, which is often used with Prefix. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one CommonPrefixes. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one CommonPrefixes. Assume that a bucket has objects abcd, abcde, and bbcde in it. If delimiter is set to d and prefix is set to a, files abcd and abcde are grouped into a CommonPrefixes with abcd as the prefix. If only delimiter is set to d, files abcd and abcde are grouped into a CommonPrefixes with abcd as the prefix, and bbcde is grouped separately into another CommonPrefixes with bbcde as the prefix. For a PFS, if this parameter is not specified, all the contents in the current directory are recursively listed by default, and subdirectories are also listed. In big data scenarios, PFSs usually have multiple directory levels and each directory level has a large number of files. In such case, you are advised to configure [delimiter=/] to list the contents in the current directory but exclude the contents in subdirectories, thereby accelerating the listing. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
nextMarker	String	Explanation: Where in the bucket the next listing begins. If not all objects are returned, the response contains this parameter to mark the last object returned for the current request. You can use the value of this parameter as the value of Marker in the subsequent request to list the remaining objects. Value range: The value must contain 1 to 1,024 characters. Default value: None
location	String	Explanation: Region where a bucket is located.
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Table 10-33 ObsObject

Parameter	Type	Description
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectContent	InputStream	Explanation: The data stream of the object. Default value: None

Parameter	Type	Description
owner	Owner	Explanation: Object owner. This parameter contains the domain ID and name of the object owner. For details, see Table 10-34 . Default value: None
metadata	ObjectMetadata	Explanation: Object metadata. Value range: Object metadata. For details, see Table 10-36 . Default value: None
objectType	ObjectTypeEnum	Explanation: Type of the object. Value range: Object type. For details, see Table 10-37 . Default value: None

Table 10-34 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-35 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 10-36 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the object data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the object must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 10-35. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Table 10-37 ObjectTypeEnum

Constant	Default Value	Description
NORMAL	NORMAL	Normal objects
APPENDABLE	APPENDABLE	Appendable objects

Code Example: Listing Objects in a Bucket

This example lists objects in bucket **examplebucket**. A maximum of 1,000 objects can be returned.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectListing;
import com.obs.services.model.ObsObject;
public class ListObjects001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Listing objects.
            ObjectListing result = obsClient.listObjects("examplebucket");
            for (ObsObject obsObject : result.getObjects()) {
                System.out.println("listObjects successfully");
                System.out.println("ObjectKey:" + obsObject.getObjectKey());
                System.out.println("Owner:" + obsObject.getOwner());
            }
        } catch (ObsException e) {
            System.out.println("listObjects failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("listObjects failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

```
}  
}
```

Code Example: Specifying the Object Count for Listing

This example lists a specified number of objects in bucket **examplebucket**. A maximum of 100 objects can be listed.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.ListObjectsRequest;  
import com.obs.services.model.ObjectListing;  
import com.obs.services.model.ObsObject;  
public class ListObjects002 {  
    public static void main(String[] args) {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using  
        // hard coding may result in leakage.  
        // Obtain an AK/SK pair on the management console.  
        String ak = System.getenv("ACCESS_KEY_ID");  
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");  
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are  
        // advised not to use hard coding, which may result in information leakage.  
        // Obtain an AK/SK pair and a security token using environment variables or import them in other  
        // ways.  
        // String securityToken = System.getenv("SECURITY_TOKEN");  
        // Enter the endpoint corresponding to the region where the bucket is to be created.  
        String endPoint = "https://your-endpoint";  
        // Obtain an endpoint using environment variables or import it in other ways.  
        //String endPoint = System.getenv("ENDPOINT");  
  
        // Create an ObsClient instance.  
        // Use the permanent AK/SK pair to initialize the client.  
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
        // Use the temporary AK/SK pair and security token to initialize the client.  
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);  
  
        try {  
            // List a specified number of objects.  
            ListObjectsRequest request = new ListObjectsRequest("examplebucket");  
            // Set the number of objects to be listed to 100.  
            request.setMaxKeys(100);  
            ObjectListing result = obsClient.listObjects(request);  
            for (ObsObject obsObject : result.getObjects()) {  
                System.out.println("listObjects successfully");  
                System.out.println("ObjectKey:" + obsObject.getObjectKey());  
                System.out.println("Owner:" + obsObject.getOwner());  
            }  
        } catch (ObsException e) {  
            System.out.println("listObjects failed");  
            // Request failed. Print the HTTP status code.  
            System.out.println("HTTP Code:" + e.getResponseCode());  
            // Request failed. Print the server-side error code.  
            System.out.println("Error Code:" + e.getErrorCode());  
            // Request failed. Print the error details.  
            System.out.println("Error Message:" + e.getErrorMessage());  
            // Request failed. Print the request ID.  
            System.out.println("Request ID:" + e.getErrorRequestId());  
            System.out.println("Host ID:" + e.getErrorHostId());  
            e.printStackTrace();  
        } catch (Exception e) {  
            System.out.println("listObjects failed");  
            // Print other error information.  
            e.printStackTrace();  
        }  
    }  
}
```

Code Example: Specifying the Prefix for Listing

This example specifies a prefix to list objects in bucket **examplebucket**. A maximum of 100 objects can be listed.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListObjectsRequest;
import com.obs.services.model.ObjectListing;
import com.obs.services.model.ObsObject;
public class ListObjects003 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // List objects by specifying a prefix.
            ListObjectsRequest request = new ListObjectsRequest("examplebucket");
            // List 100 objects with the specified prefix.
            request.setMaxKeys(100);
            request.setPrefix("prefix");
            ObjectListing result = obsClient.listObjects(request);
            for (ObsObject obsObject : result.getObjects()) {
                System.out.println("listObjects successfully");
                System.out.println("ObjectKey:" + obsObject.getObjectKey());
                System.out.println("Owner:" + obsObject.getOwner());
            }
        } catch (ObsException e) {
            System.out.println("listObjects failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("listObjects failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

Code Example: Specifying the Start Position for Listing

This example specifies a start position to list objects in bucket **examplebucket**. A maximum of 100 objects can be listed.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListObjectsRequest;
import com.obs.services.model.ObjectListing;
import com.obs.services.model.ObsObject;
public class ListObjects004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // List objects by specifying the start position.
            ListObjectsRequest request = new ListObjectsRequest("examplebucket");
            // Specify that 100 objects whose names follow test in lexicographical order will be listed.
            request.setMaxKeys(100);
            request.setMarker("test");
            ObjectListing result = obsClient.listObjects(request);
            for (ObsObject obsObject : result.getObjects()) {
                System.out.println("listObjects successfully");
                System.out.println("ObjectKey:" + obsObject.getObjectKey());
                System.out.println("Owner:" + obsObject.getOwner());
            }
        } catch (ObsException e) {
            System.out.println("listObjects failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("listObjects failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

Code Example: Using Pagination to List All Objects

This example lists all objects in bucket **examplebucket** using pagination. A maximum of 100 objects can be listed on each page.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListObjectsRequest;
import com.obs.services.model.ObjectListing;
import com.obs.services.model.ObsObject;
public class ListObjects005 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // List all objects using pagination.
            ListObjectsRequest request = new ListObjectsRequest("examplebucket");
            // Set the number of objects displayed per page to 100.
            request.setMaxKeys(100);
            ObjectListing result;
            do {
                result = obsClient.listObjects(request);
                for (ObsObject obsObject : result.getObjects()) {
                    System.out.println("listObjects successfully");
                    System.out.println("ObjectKey:" + obsObject.getObjectKey());
                    System.out.println("Owner:" + obsObject.getOwner());
                }
                request.setMarker(result.getNextMarker());
            } while (result.isTruncated());
        } catch (ObsException e) {
            System.out.println("listObjects failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("listObjects failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

Code Example: Listing Objects in a Folder

There is no concept of folders in OBS. All elements stored in OBS buckets are objects. Folders are actually objects whose sizes are 0 and whose names end with a slash (/). You can set a folder name as a prefix to list objects in this folder. This example lists objects in a folder.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListObjectsRequest;
import com.obs.services.model.ObjectListing;
import com.obs.services.model.ObsObject;
public class ListObjects006 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            //List all objects in a folder.
            ListObjectsRequest request = new ListObjectsRequest("examplebucket");
            // Set folder name dir/ as the prefix.
            request.setPrefix("dir/");
            request.setMaxKeys(100);
            ObjectListing result;
            do {
                result = obsClient.listObjects(request);
                for (ObsObject obsObject : result.getObjects()) {
                    System.out.println("listObjects successfully");
                    System.out.println("ObjectKey:" + obsObject.getObjectKey());
                    System.out.println("Owner:" + obsObject.getOwner());
                }
                request.setMarker(result.getNextMarker());
            } while (!result.isTruncated());
        } catch (ObsException e) {
            System.out.println("listObjects failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("listObjects failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```



```
}  
}
```

Code Example: Listing All Objects in a Bucket by Folder Name

This example lists all objects in bucket **examplebucket**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.ListObjectsRequest;  
import com.obs.services.model.ObjectListing;  
import com.obs.services.model.ObsObject;  
public class ListObjects007 {  
    public static void main(String[] args) {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using  
hard coding may result in leakage.  
        // Obtain an AK/SK pair on the management console.  
        String ak = System.getenv("ACCESS_KEY_ID");  
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");  
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are  
advised not to use hard coding, which may result in information leakage.  
        // Obtain an AK/SK pair and a security token using environment variables or import them in other  
ways.  
        // String securityToken = System.getenv("SECURITY_TOKEN");  
        // Enter the endpoint corresponding to the region where the bucket is to be created.  
        String endPoint = "https://your-endpoint";  
        // Obtain an endpoint using environment variables or import it in other ways.  
        //String endPoint = System.getenv("ENDPOINT");  
  
        // Create an ObsClient instance.  
        // Use the permanent AK/SK pair to initialize the client.  
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
        // Use the temporary AK/SK pair and security token to initialize the client.  
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);  
  
        try {  
            // List all objects in the root directory.  
            ListObjectsRequest request = new ListObjectsRequest("examplebucket");  
            request.setMaxKeys(1000);  
            // Set folder isolators to slashes (/).  
            request.setDelimiter("/");  
            ObjectListing result = obsClient.listObjects(request);  
            System.out.println("Objects in the root directory:");  
            for (ObsObject obsObject : result.getObjects()) {  
                System.out.println("listObjects successfully");  
                System.out.println("ObjectKey:" + obsObject.getObjectKey());  
                System.out.println("Owner:" + obsObject.getOwner());  
            }  
            listObjectsByPrefix(obsClient, request, result);  
        } catch (ObsException e) {  
            System.out.println("listObjects failed");  
            // Request failed. Print the HTTP status code.  
            System.out.println("HTTP Code:" + e.getResponseCode());  
            // Request failed. Print the server-side error code.  
            System.out.println("Error Code:" + e.getErrorCode());  
            // Request failed. Print the error details.  
            System.out.println("Error Message:" + e.getErrorMessage());  
            // Request failed. Print the request ID.  
            System.out.println("Request ID:" + e.getErrorRequestId());  
            System.out.println("Host ID:" + e.getErrorHostId());  
            e.printStackTrace();  
        } catch (Exception e) {  
            System.out.println("listObjects failed");  
            // Print other error information.  
            e.printStackTrace();  
        }  
    }  
}
```

Code Example: Recursively Listing Objects in a Sub-folder

This example recursively lists objects in a sub-folder.

```
// Recursively list objects in a sub-folder.
static void listObjectsByPrefix(ObsClient obsClient, ListObjectsRequest request, ObjectListing result)
    throws ObsException {
    for (String prefix : result.getCommonPrefixes()) {
        System.out.println("Objects in folder [" + prefix + "]:");
        request.setPrefix(prefix);
        result = obsClient.listObjects(request);
        for (ObsObject obsObject : result.getObjects()) {
            System.out.println("ObjectKey:" + obsObject.getObjectKey());
            System.out.println("Owner:" + obsObject.getOwner());
        }
        listObjectsByPrefix(obsClient, request, result);
    }
}
```

NOTE

- The sample code does not apply to scenarios where the number of objects in a folder exceeds 1,000.
- Since all folder names end with a slash (/) and the objects and sub-folders to list are under the folder, **delimiter** is always a slash (/).
- In the returned result of each recursion, **ObjectListing.getObjects** includes the objects under the folder, and **ObjectListing.getCommonPrefixes** includes the sub-folders under the folder.

10.6 Deleting an Object (SDK for Java)

Function

This API deletes objects from a specific bucket.

Restrictions

- To delete an object, you must be the bucket owner or have the required permission (**obs:object:DeleteObject** in IAM or **DeleteObject** in a bucket policy).
- If versioning is not enabled for a bucket, deleted objects cannot be recovered.

Method

obsClient.deleteObject([DeleteObjectRequest request](#))

Request Parameters

Table 10-38 List of request parameters

Param eter	Type	Mandat ory (Yes/No)	Description
request	DeleteObjectRequest	Yes	Explanation: Request parameters for deleting an object. For details, see Table 10-39 .

Table 10-39 DeleteObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None

Responses

Table 10-40 DeleteObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
versionId	String	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
deleteMarker	boolean	Explanation: Whether the deleted object is a delete marker. Value range: • true: The deleted object is a delete marker. • false: The deleted object is not a delete marker. Default value: false
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Code Examples

This example deletes object **objectname** from bucket **examplebucket**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
public class DeleteObject001 {  
    public static void main(String[] args) {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
```

```
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Delete the object.
    obsClient.deleteObject("examplebucket", "objectname");
    System.out.println("deleteObject successfully");
} catch (ObsException e) {
    System.out.println("deleteObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

10.7 Batch Deleting Objects (SDK for Java)

Function

This API deletes objects in batches from a specific bucket. Deleted objects cannot be recovered.

In a batch delete operation, OBS concurrently deletes the specified objects and returns the deletion result of each object.

Restrictions

- To delete objects in a batch, you must be the bucket owner or have the required permission (**obs:object:DeleteObject** in IAM or **DeleteObject** in a bucket policy).
- If versioning is not enabled for a bucket, deleted objects cannot be recovered.
- A maximum of 1,000 objects can be deleted at a time. If you send a request for deleting more than 1,000 objects, OBS returns an error message.

- After concurrent tasks are assigned, if an internal error occurs during cyclic deletion of multiple objects, an object may be deleted in the index data but still exist in the metadata.

Method

obsClient.deleteObjects([DeleteObjectsRequest deleteRequest](#))

Request Parameters

Table 10-41 List of request parameters

Parameter	Type	Mand atory (Yes/ No)	Description
deleteRequ est	DeleteObj ectsReque st	Yes	Explanation: Request parameters for deleting objects in batches. For details, see Table 10-42 .

Table 10-42 DeleteObjectsRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
keyAndVersions	List< Key AndVersion >	Yes	Explanation: List of objects to be deleted. For details, see Table 10-43 .

Parameter	Type	Mandatory (Yes/No)	Description
quiet	boolean	No	Explanation: Response mode to the request for deleting objects in a batch. Value range: • false: The detailed mode. Results of both successful and failed deletions are returned. • true: The quiet mode. Only results of failed deletions are returned. Default value: false

Table 10-43 KeyAndVersion

Parameter	Type	Mandatory (Yes/No)	Description
key	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None. If this parameter is left blank, the latest version of the object is deleted.

Responses

Table 10-44 DeleteObjectsResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
deletedObjectResults	List< DeleteObjectResult >	Explanation: Response results of the request for deleting objects in a batch. For details, see Table 10-45 .
errorResults	List< ErrorResult >	Explanation: List of objects that fail to be deleted. For details, see Table 10-46 .

Table 10-45 DeleteObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
versionId	String	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
deleteMarker	boolean	Explanation: Whether the deleted object is a delete marker. Value range: • true: The deleted object is a delete marker. • false: The deleted object is not a delete marker. Default value: false

Parameter	Type	Description
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Table 10-46 ErrorResult

Parameter	Type	Description
versionId	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
errorCode	String	Explanation: Error code for the failed deletion.

Parameter	Type	Description
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
message	String	Explanation: Error message for the failed deletion.

Code Examples

This example deletes all objects whose names start with **exampleObjectPrefix** in bucket **example-bucket** in a batch. Note that if **exampleObjectPrefix** is left blank, **all objects in the bucket will be deleted**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.DeleteObjectsRequest;
import com.obs.services.model.DeleteObjectsResult;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
import java.util.Map;

public class DeleteObjects002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create a single instance of ObsClient.
        // Use a permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use a temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
```

```
try {
    String exampleBucket = "example-bucket";
    String exampleObjectPrefix = "exampleObjectPrefix";
    // Batch delete the objects.
    ListVersionsRequest request = new ListVersionsRequest(exampleBucket);
    // Delete 100 objects at a time.
    request.setMaxKeys(100);
    request.setPrefix(exampleObjectPrefix);
    ListVersionsResult result;
    do {
        result = obsClient.listVersions(request);
        DeleteObjectsRequest deleteRequest = new DeleteObjectsRequest(exampleBucket);
        // deleteRequest.setQuiet(true);
        // This example uses the verbose mode. To use the quiet mode, add this line of code.
        for (VersionOrDeleteMarker v : result.getVersions()) {
            // Obtain object names and version IDs.
            deleteRequest.addKeyAndVersion(v.getKey(), v.getVersionId());
        }
        if(deleteRequest.getKeyAndVersions().length > 0){
            DeleteObjectsResult deleteResult = obsClient.deleteObjects(deleteRequest);
            // Print the deleted objects.
            System.out.println("DeletedObjectResults:" + deleteResult.getDeletedObjectResults());
            // Print the objects that were not deleted.
            System.out.println("ErrorResults:" + deleteResult.getErrorResults());
        }else{
            System.out.println("No object to delete");
        }
        request.setKeyMarker(result.getNextKeyMarker());
        // If versioning is not enabled, do not configure VersionIdMarker.
        request.setVersionIdMarker(result.getNextVersionIdMarker());
    } while (result.isTruncated());
    System.out.println("deleteObjects successfully");
} catch (ObsException e) {
    System.out.println("deleteObjects failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    Map<String, String> headers = e.getResponseHeaders();// Check Map entries and print all headers
    associated with the error.
    if(headers != null){
        for (Map.Entry<String, String> header : headers.entrySet()) {
            if(header.getKey().contains("error")){
                System.out.println(header.getKey()+"."+header.getValue());
            }
        }
    }
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteObjects failed");
    // Print other error details.
    e.printStackTrace();
}
}
```

This example batch deletes all objects in bucket **examplebucket**. Exercise caution when performing this operation.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.DeleteObjectsRequest;
import com.obs.services.model.DeleteObjectsResult;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
```

```
import com.obs.services.model.VersionOrDeleteMarker;
public class DeleteObjects001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Batch delete the objects.
            ListVersionsRequest request = new ListVersionsRequest("examplebucket");
            // Delete 100 objects at a time.
            request.setMaxKeys(100);
            ListVersionsResult result;
            do {
                result = obsClient.listVersions(request);
                DeleteObjectsRequest deleteRequest = new DeleteObjectsRequest("examplebucket");
                // deleteRequest.setQuiet(true); // This example uses the verbose mode. To use the quiet mode,
                // add this line of code.
                for (VersionOrDeleteMarker v : result.getVersions()) {
                    // Obtain objects' names and version IDs.
                    deleteRequest.addKeyAndVersion(v.getKey(), v.getVersionId());
                }
                if(deleteRequest.getKeyAndVersions().length > 0){
                    DeleteObjectsResult deleteResult = obsClient.deleteObjects(deleteRequest);
                    // Print the deleted objects.
                    System.out.println("DeletedObjectResults:" + deleteResult.getDeletedObjectResults());
                    // Print the objects that were not deleted.
                    System.out.println("ErrorResults:" + deleteResult.getErrorResults());
                }else{
                    System.out.println("No object to delete");
                }
                request.setKeyMarker(result.getNextKeyMarker());
                // If versioning is not enabled, do not configure VersionIdMarker.
                request.setVersionIdMarker(result.getNextVersionIdMarker());
            } while (result.isTruncated());
            System.out.println("deleteObjects successfully");
        } catch (ObsException e) {
            System.out.println("deleteObjects failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("deleteObjects failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```



```
    }  
  }  
}
```

This example deletes all objects in the **test** folder in the **examplebucket** bucket with versioning disabled.

```
import com.obs.services.ObsClient;  
import com.obs.services.ObsConfiguration;  
import com.obs.services.exception.ObsException;  
import com.obs.services.model.*;  
import java.io.IOException;  
import java.util.Map;  
  
public class Delete_Objects_test {  
    public static void main(String[] args) throws IOException {  
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using  
        // hard coding may result in leakage.  
        // Obtain an AK/SK pair on the management console.  
        String ak = System.getenv("ACCESS_KEY_ID");  
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");  
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are  
        // advised not to use hard coding, which may result in information leakage.  
        // Obtain an AK/SK pair and a security token using environment variables or import them in other  
        // ways.  
        // String securityToken = System.getenv("SECURITY_TOKEN");  
        // Enter the endpoint corresponding to the region where the bucket is located.  
        String endPoint = "https://your-endpoint";  
        // Obtain an endpoint using environment variables or import it in other ways.  
        // String endPoint = System.getenv("ENDPOINT");  
  
        // Create a single instance of ObsClient.  
        // Use a permanent AK/SK pair to initialize the client.  
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
        // Use a temporary AK/SK pair and security token to initialize the client.  
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);  
        try {  
            String exampleBucket = "examplebucket";  
            String exampleObjectPrefix = "test/";  
            ListObjectsRequest listObjectsRequest = new ListObjectsRequest(exampleBucket);  
            listObjectsRequest.setMaxKeys(100);  
            listObjectsRequest.setPrefix(exampleObjectPrefix);  
            ObjectListing objectListing;  
  
            do {  
                objectListing = obsClient.listObjects(listObjectsRequest);  
                DeleteObjectsRequest deleteRequest = new DeleteObjectsRequest(exampleBucket);  
                // This example uses the verbose mode. To use the quiet mode, add this line of code.  
                deleteRequest.setQuiet(true);  
                for (ObsObject object : objectListing.getObjects()) {  
                    deleteRequest.addKeyAndVersion(object.getObjectKey(), null);  
                }  
  
                if (deleteRequest.getKeyAndVersions().length > 0) {  
                    DeleteObjectsResult deleteResult = obsClient.deleteObjects(deleteRequest);  
                    // Print the deleted objects.  
                    System.out.println("DeletedObjectResults:" + deleteResult.getDeletedObjectResults());  
                    // Print the objects that were not deleted.  
                    System.out.println("ErrorResults:" + deleteResult.getErrorResults());  
                } else {  
                    System.out.println("No object to delete");  
                }  
                listObjectsRequest.setMarker(objectListing.getNextMarker());  
            } while (objectListing.isTruncated());  
            System.out.println("deleteObjects successfully");  
        } catch (ObsException e) {  
            System.out.println("deleteObjects failed");  
            // Request failed. Print the HTTP status code.  
            System.out.println("HTTP Code:" + e.getResponseCode());  
            // Request failed. Print the server-side error code.
```

```
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
Map<String, String> headers = e.getResponseHeaders();
// Check all Map entries and print all headers associated with the error.
if (headers != null) {
    for (Map.Entry<String, String> header : headers.entrySet()) {
        if (header.getKey().contains("error")) {
            System.out.println(header.getKey() + ":" + header.getValue());
        }
    }
}
e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteObjects failed");
    // Print other error details.
    e.printStackTrace();
}
}
```

10.8 Copying an Object (SDK for Java)

Function

This API copies an object stored in OBS to another path, with a copy created during the process. You can create a copy of an object up to 5 GB in a single operation.

Restrictions

- To copy an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- You must have the read permission on the source object.
- The object copy request carries the information about the source bucket and object to be copied in the header field. The message body cannot be carried.
- Cross-bucket replication in the same region is supported, but cross-region replication is not supported.
- An object copy can be up to 5 GB in size. If the source object exceeds 5 GB, you can only perform a [multipart copy](#).
- If the source object is in the Cold storage class, you must restore it first.

Method

obsClient.copyObject([CopyObjectRequest request](#))

Request Parameters

Table 10-47 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	CopyObjectRequest	Yes	Explanation: Request parameters for copying an object. For details, see Table 10-48 .

Table 10-48 CopyObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Target bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those

Parameter	Type	Mandatory (Yes/No)	Description
			set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Target object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
acl	AccessControlList	No	Explanation: An ACL specified for the object. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 10-49 for the available options. - To use a user-defined ACL, see Table 10-50 to configure the required parameters. Default value: AccessControlList.RES_T_CANNED_PRIVATE

Parameter	Type	Mandatory (Yes/No)	Description
successRedirect-Location	String	No	Explanation: Address (URL) to which a successfully answered request is redirected. • If the value is valid and the request is successful, OBS returns status code 303. Location contains SuccessActionRedirect as well as the bucket name, object name, and object ETag. • If the value is invalid, OBS ignores this parameter. In such case, the Location header is the object address, and OBS returns a status code based on whether the operation succeeds or fails. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
sourceBucketName	String	Yes	Explanation: Name of the source bucket. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those

Parameter	Type	Mandatory (Yes/No)	Description
			set in the first creation request. Default value: None
sourceObjectKey	String	Yes	Explanation: Source object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
newObjectMetadata	ObjectMetadata	No	Explanation: User-defined metadata of the target object. For details, see Table 10-57 . Restrictions: replaceMetadata must be used together with newObjectMetadata . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
replaceMetadata	boolean	No	Explanation: Whether to rewrite the metadata of the source object. Restrictions: replaceMetadata must be used together with newObjectMetadata . Value range: true: The metadata of the source object is rewritten. false: The metadata of the source object is not rewritten. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifModifiedSince	java.util.Date	No	Explanation: The source object is copied if it has been modified since the specified time; otherwise, an exception is thrown. Restrictions: • If the object copy request includes ifUnmodifiedSince, ifMatchTag, ifModifiedSince, or ifNoneMatchTag, and the specified condition is not met, the copy will fail and an exception will be thrown with HTTP status code 412 precondition failed returned. • ifModifiedSince and ifNoneMatchTag can be used together. So do ifUnmodifiedSince and ifMatchTag. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifUnmodifiedSince	java.util.Date	No	Explanation: If the source object has not been modified since the specified time, it is copied. Otherwise, an exception is thrown. Restrictions: - If the object copy request includes ifUnmodifiedSince, ifMatchTag, ifModifiedSince, or ifNoneMatchTag, and the specified condition is not met, the copy will fail and an exception will be thrown with HTTP status code 412 precondition failed returned. ifModifiedSince and ifNoneMatchTag can be used together. So do ifUnmodifiedSince and ifMatchTag. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the source object is the same as the preset ETag, the object is returned. Otherwise, an error is returned. The ETag of the source object is the MD5 value of it. Restrictions: - If the object copy request includes ifUnmodifiedSince, ifMatchTag, ifModifiedSince, or ifNoneMatchTag, and the specified condition is not met, the copy will fail and an exception will be thrown with HTTP status code 412 precondition failed returned. ifModifiedSince and ifNoneMatchTag can be used together. So do ifUnmodifiedSince and ifMatchTag. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the source object is different from the preset ETag, the object is returned. Otherwise, an error is returned. The ETag of the source object is the MD5 value of it. Restrictions: - If the object copy request includes ifUnmodifiedSince, ifMatchTag, ifModifiedSince, or ifNoneMatchTag, and the specified condition is not met, the copy will fail and an exception will be thrown with HTTP status code 412 precondition failed returned. ifModifiedSince and ifNoneMatchTag can be used together. So do ifUnmodifiedSince and ifMatchTag. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
versionId	String	No	Explanation: Source object version ID. Value range: The value must contain 32 characters. Default value: None

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 10-49 Pre-defined ACL

Constant	Description
AccessControllist.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.

Constant	Description
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControlList.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Table 10-50 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 10-51 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< Grant AndPermission >	No	Explanation: Grantee information. For details, see Table 10-52 .

Table 10-51 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-52 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 10-54 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 10-53 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 10-53 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 10-54 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 10-55 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 10-56 . Default value: None

Table 10-55 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 10-56 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 10-57 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the object data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 10-58. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 10-58 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Responses

Table 10-59 CopyObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
lastModified	java.util.Date	Explanation: Last time the target object was modified. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of the target object. Base64-encoded, 128-bit MD5 value of an object. ETag is the unique identifier of the object contents and is used to determine whether the contents of an object are changed. For example, if the ETag value is A when an object is uploaded and is B when the object is downloaded, this indicates the contents of the object are changed. The ETag reflects changes only to the contents of an object, not its metadata. Objects created by the upload and copy operations have unique ETags after being encrypted using MD5. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
crc64	str	Explanation: The 64-bit CRC value calculated for the object based on the ECMA-182 standard. The CRC64 value is the unique identifier of the object content. It can be used to determine whether the object content is changed. If an object has different CRC64 values when being uploaded and downloaded, its content has been changed. CRC64 reflects changes to the contents of the object, not its metadata. Restrictions: • This parameter is returned when the CRC64 value was verified for the source object, or when the CRC64 feature has been enabled for the bucket. The returned CRC64 value is for the target object. • This parameter is not supported for POSIX or SFS objects. Value range: A 64-bit CRC value calculated based on the ECMA-182 standard Default value: None

Parameter	Type	Description
versionId	String	Explanation: Version ID of the target object. Value range: The value must contain 32 characters. Default value: None
copySourceVersionId	String	Explanation: Version ID of the source object. Value range: The value must contain 32 characters. Default value: None
storageClass	StorageClassEnum	Explanation: Storage class of the target object. Value range: See Table 10-58 . Default value: None

Code Example: Copying an Object

This example copies object **sourceobjectname** from bucket **sourcebucketname** to bucket **destbucketname** as object **destobjectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CopyObjectResult;
public class CopyObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");
```

```
// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Copy an object.
    CopyObjectResult result =
        obsClient.copyObject("sourcebucketname", "sourceobjectname", "destbucketname",
"destobjectname");
    System.out.println("copyObject successfully");
    System.out.println("StatusCode:" + result.getStatusCode());
    System.out.println("Etag:" + result.getEtag());
} catch (ObsException e) {
    System.out.println("copyObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("copyObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Copying an Object and Rewriting Object Attributes

This example rewrites object attributes when object **sourceobjectname** is copied from bucket **sourcebucketname** to bucket **destbucketname** as object **destobjectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CopyObjectRequest;
import com.obs.services.model.CopyObjectResult;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.StorageClassEnum;
public class CopyObject002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
```

```
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Rewrite object attributes when copying an object.
    CopyObjectRequest request = new CopyObjectRequest("sourcebucketname", "sourceobjectname",
"destbucketname", "destobjectname");
    // Rewrite object attributes.
    request.setReplaceMetadata(true);
    HashMap<String, String> userHeaders = new HashMap<>();
    userHeaders.put("Content-Type","image/jpeg");
    request.setUserHeaders(userHeaders);
    CopyObjectResult result = obsClient.copyObject(request);
    System.out.println("copyObject successfully");
    System.out.println("Etag:" + result.getEtag());
} catch (ObsException e) {
    System.out.println("copyObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("copyObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Copying an Object Conditionally

This example specifies conditions when object **sourceobjectname** is copied from bucket **sourcebucketname** to bucket **destbucketname** as object **destobjectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CopyObjectRequest;
import com.obs.services.model.CopyObjectResult;
import java.text.SimpleDateFormat;
public class CopyObject003 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
    }
}
```

```
try {
    // Set conditions for the copy.
    CopyObjectRequest request =
        new CopyObjectRequest("sourcebucketname", "sourceobjectname", "destbucketname",
"destobjectname");
    request.setIfModifiedSince(new SimpleDateFormat("yyyy-MM-dd").parse("2016-01-01"));
    request.setIfNoneMatchTag("none-match-etag");
    CopyObjectResult result = obsClient.copyObject(request);
    System.out.println("copyObject successfully");
    System.out.println("Etag:" + result.getEtag());
} catch (ObsException e) {
    System.out.println("copyObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("copyObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Copying an Object and Rewriting its ACL

This example rewrites the object ACL when object **sourceobjectname** is copied from bucket **sourcebucketname** to bucket **destbucketname** as object **destobjectname**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.CopyObjectRequest;
import com.obs.services.model.CopyObjectResult;
public class CopyObject004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Rewrite the object ACL when copying the object.
            CopyObjectRequest request =
                new CopyObjectRequest("sourcebucketname", "sourceobjectname", "destbucketname",
```



```
"destObjectName");
    // Rewrite the object ACL to Private.
    request.setAcl(AccessControlList.REST_CANNED_PRIVATE);
    CopyObjectResult result = obsClient.copyObject(request);
    System.out.println("copyObject successfully");
    System.out.println("Etag:" + result.getEtag());
} catch (ObsException e) {
    System.out.println("copyObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("copyObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

10.9 Copying an Object - Multipart (SDK for Java)

Function

This API allows you to upload a part by copying an object or part of this object.
You can call **ObsClient.copyPart** to copy parts.

Restrictions

- To copy an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- You must have the read permission on the source object.
- The object copy request carries the information about the source bucket and object to be copied in the header field. The message body cannot be carried.
- Cross-bucket replication in the same region is supported, but cross-region replication is not supported.
- If the source object is in the Cold storage class, you must restore it first.

Method

obsClient.copyPart([CopyPartRequest request](#))

Request Parameters

Table 10-60 List of request parameters

Parameter	Type	Mandator y (Yes/No)	Description
request	CopyPartRe quest	Yes	Explanation: Request parameters for copying an object. For details, see Table 10-61 .

Table 10-61 CopyPartRequest

Parameter	Type	Mandatory (Yes/No)	Description
sourceBucketName	String	Yes	Explanation: Name of the source bucket. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
sourceObjectKey	String	Yes	Explanation: Source object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
destinationBucket-Name	String	Yes	Explanation: Target bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
destinationObject-Key	String	Yes	Explanation: Target object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
byteRangeStart	Long	No	Explanation: Start position for the multipart copy Value range: A non-negative integer, in bytes. Default value: 0
byteRangeEnd	Long	No	Explanation: End position for the multipart copy Restrictions: The value must be greater than that of RangeStart. The upper limit of the value is the object length minus 1, in bytes. Value range: A non-negative integer, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
versionId	String	No	Explanation: Version ID of the source object, for example, G001117FCE89978B0000401205D5DC9A . If the source object has multiple versions, you can specify this parameter. Value range: The value must contain 32 characters. Default value: None
partNumber	int	Yes	Explanation: Part number. Value range: An integer ranging from 1 to 10,000 Default value: None
uploadId	String	Yes	Explanation: Multipart upload ID, for example, 000001648453845DBB78F2340DD460D8 . Value range: The value must contain 32 characters. Default value: None

Constant	Default Value
AES256	AES256

Constant	Default Value
KMS	kms

Constant	Default Value
AES256	AES256

Responses

Table 10-62 CopyPartResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
partNumber	int	Explanation: Part number. Value range: An integer ranging from 1 to 10000. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
lastModified	java.util.Date	Explanation: Last time the target part was modified. Default value: None

Code Examples

This example copies object **sourceobjectname** in parts from bucket **sourcebucketname** to bucket **destbucketname** as object **destobjectname**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.CompleteMultipartUploadRequest;
import com.obs.services.model.CopyPartRequest;
import com.obs.services.model.CopyPartResult;
import com.obs.services.model.InitiateMultipartUploadRequest;
import com.obs.services.model.InitiateMultipartUploadResult;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.PartEtag;
import java.util.ArrayList;
import java.util.Collections;
import java.util.List;
import java.util.concurrent.ExecutorService;
import java.util.concurrent.Executors;
import java.util.concurrent.TimeUnit;
public class CopyPart001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Copy parts.
            final String destBucketName = "destbucketname";
            final String destObjectKey = "destobjectname";
            final String sourceBucketName = "sourcebucketname";
            final String sourceObjectKey = "sourceobjectname";
            // Initialize the thread pool.
            ExecutorService executorService = Executors.newFixedThreadPool(20);
            // Initiate a multipart upload.
            InitiateMultipartUploadRequest request = new InitiateMultipartUploadRequest(destBucketName,
            destObjectKey);
            InitiateMultipartUploadResult result = obsClient.initiateMultipartUpload(request);
            final String uploadId = result.getUploadId();
            System.out.println("uploadId:" + uploadId + "\n");
            // Obtain information about the large object.
            ObjectMetadata metadata = obsClient.getObjectMetadata(sourceBucketName, sourceObjectKey);
            // Set the part size to 100 MB.
            long partSize = 100 * 1024 * 1024L;
            long objectSize = metadata.getContentLength();
            // Calculate the number of parts to be copied.
            long partCount = objectSize % partSize == 0 ? objectSize / partSize : objectSize / partSize + 1;
            final List<PartEtag> partEtags = Collections.synchronizedList(new ArrayList<>());
            // Start copying parts concurrently.
            for (int i = 0; i < partCount; i++) {
                // Start position for copying parts
                final long rangeStart = i * partSize;
                // End position for copying parts
                final long rangeEnd = (i + 1 == partCount) ? objectSize - 1 : rangeStart + partSize - 1;
                // Set the part number.
                final int partNumber = i + 1;
                executorService.execute(
                    new Runnable() {
                        @Override
```

```
public void run() {
    CopyPartRequest request = new CopyPartRequest();
    request.setUploadId(uploadId);
    request.setSourceBucketName(sourceBucketName);
    request.setSourceObjectKey(sourceObjectKey);
    request.setDestinationBucketName(destBucketName);
    request.setDestinationObjectKey(destObjectKey);
    request.setByteRangeStart(rangeStart);
    request.setByteRangeEnd(rangeEnd);
    request.setPartNumber(partNumber);
    CopyPartResult result;
    try {
        result = obsClient.copyPart(request);
        System.out.println("Part#" + partNumber + " done\n");
        partEtags.add(new PartEtag(result.getEtag(), result.getPartNumber()));
    } catch (ObsException e) {
        e.printStackTrace();
    }
}

});

}

// Wait until the copy is complete.
executorService.shutdown();
while (!executorService.isTerminated()) {
    try {
        executorService.awaitTermination(5, TimeUnit.SECONDS);
    } catch (InterruptedException e) {
        e.printStackTrace();
    }
}

// Assemble parts.
CompleteMultipartUploadRequest completeMultipartUploadRequest =
    new CompleteMultipartUploadRequest(destBucketName, destObjectKey, uploadId, partEtags);
obsClient.completeMultipartUpload(completeMultipartUploadRequest);
System.out.println("copyObject successfully");
} catch (ObsException e) {
    System.out.println("copyObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("copyObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

10.10 Checking Whether an Object Exists (SDK for Java)

Function

This API checks whether an object exists. If the returned HTTP status code is **200**, the object exists. If the returned HTTP status code is **404**, the object or bucket does not exist.

Restrictions

- To check whether an object exists, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).

Method

doesObjectExist(final [GetObjectMetadataRequest request](#))

Request Parameters

Table 10-63 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectMetadataRequest	Yes	Explanation: Request parameters. For details, see Table 10-64 .

Table 10-64 GetObjectMetadataRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
userHeaders	HashMap<String, String>	No	Explanation: User header list. In HashMap , the String key and value indicate the name and value of the user header field respectively. The SDK does not process the userHeaders and instead transparently transmits it to the server for later use. Value range: The value must contain 32 characters. Default value: None
encodeHeaders	boolean	No	Explanation: Whether to encode and decode HTTP headers. Value range: true: HTTP headers are encoded and decoded. false: HTTP headers are not encoded or decoded. Default value: true

Constant	Default Value
AES256	AES256

Constant	Default Value
KMS	kms

Constant	Default Value
AES256	AES256

Responses

Table 10-65 Response headers

Method	Return Value Type	Description
<code>doesObjectExist(final GetObjectMetadataRequest request)</code>	boolean	Whether the object exists in the bucket.

Code Examples

This example checks whether object **objectname** exists in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class DoesObjectExist001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Check whether the specified object exists.
            System.out.println(obsClient.doesObjectExist("examplebucket", "objectname") ? "exists!" : "does
            not exist!");
            System.out.println("doesObjectExist successfully");
        } catch (ObsException e) {
            System.out.println("doesObjectExist failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
        }
```

```
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("doesObjectExist failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

10.11 Setting an Object Expiration Time (SDK for Java)

Function

ObsClient allows you to set the expiration time for an object when you upload it or after it has been uploaded.

Sample Code 1: Setting the Expiration Time for an Object When Uploading It

For the API reference, see [Uploading an Object - File-Based \(SDK for Java\)](#).

The following code shows how to set the expiration time for object **objectname** when uploading it to bucket **examplebucket**:

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class SetObjectExpiration001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            PutObjectRequest request = new PutObjectRequest ("bucketname", "objectkey");
            // localfile indicates the path of the local file to be uploaded, in which the file name must be
            // specified.
            request.setFile(new File("localfile"));
            // When uploading an object, set the object to expire after 30 days.
            request.setExpires(30);
            obsClient.putObject(request);
            System.out.println("putObject successfully");
        } catch (ObsException e) {
            System.out.println("putObject failed");
        }
    }
}
```



```
// Request failed. Print the HTTP status code.
System.out.println("HTTP Code:" + e.getResponseCode());
// Request failed. Print the server-side error code.
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Sample Code 2: Setting the Expiration Time for an Uploaded Object

For the API reference, see [Configuring Object Metadata \(SDK for Java\)](#).

The following code shows how to set the expiration time for object **objectname** in bucket **examplebucket**:

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.SetObjectMetadataRequest;
import java.util.HashMap;
public class SetObjectExpiration002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Configure metadata for the object.
            SetObjectMetadataRequest request = new SetObjectMetadataRequest("examplebucket",
"objectname");
            HashMap<String, String> userHeaders = new HashMap<>();
            // Set the expiration time of the object to five days.
            userHeaders.put("x-obs-expires", "5");
            // Set the metadata storage method.
            userHeaders.put("x-obs-metadata-directive", "REPLACE_NEW");
            request.setUserHeaders(userHeaders);
            ObjectMetadata metadata = obsClient.setObjectMetadata(request);
            System.out.println("setObjectMetadata successfully");
            System.out.println("UserMetadata:" + metadata.getUserMetadata("property1"));
        } catch (ObsException e) {
            System.out.println("setObjectMetadata failed");
        }
    }
}
```

```
        // Request failed. Print the HTTP status code.  
        System.out.println("HTTP Code:" + e.getResponseCode());  
        // Request failed. Print the server-side error code.  
        System.out.println("Error Code:" + e.getErrorCode());  
        // Request failed. Print the error details.  
        System.out.println("Error Message:" + e.getErrorMessage());  
        // Request failed. Print the request ID.  
        System.out.println("Request ID:" + e.getErrorRequestId());  
        System.out.println("Host ID:" + e.getErrorHostId());  
        e.printStackTrace();  
    } catch (Exception e) {  
        System.out.println("setObjectMetadata failed");  
        // Print other error information.  
        e.printStackTrace();  
    }  
}
```

11 Temporarily Authorized Access (SDK for Java)

11.1 Accessing OBS Using a Signed URL (SDK for Java)

Function

ObsClient allows you to create a URL with **Query** parameters that carry authentication information by specifying the AK and SK, HTTP method, and request parameters. This URL can be shared with other users for temporary access. When generating a URL, you need to specify the validity period of the URL to restrict the access duration of visitors.

If you want to grant other users the permission to perform operations on buckets or objects (for example, upload or download objects), you need to generate a URL for the corresponding request (for example, a URL with the PUT request for uploading an object) and provide the URL to other users.

If a CORS or signature mismatch error occurs, refer to the following steps to troubleshoot the issue:

1. If CORS is not configured, you need to configure CORS rules on OBS Console.
2. If the signatures do not match, check whether signature parameters are correct. For example, during an object upload, the backend uses **Content-Type** to calculate the signature and generate an authorized URL, but if **Content-Type** is not set or is set to an incorrect value when the frontend uses the authorized URL, a CORS error occurs. To avoid this issue, ensure that **Content-Type** fields are the same at the frontend and backend.
3. A CDN acceleration domain name cannot be used to create a signed URL.

Operations That Can Be Performed Using a Signed URL

The following table lists operations that can be performed through a signed URL.

Table 11-1 HttpMethodEnum & SpecialParamEnum

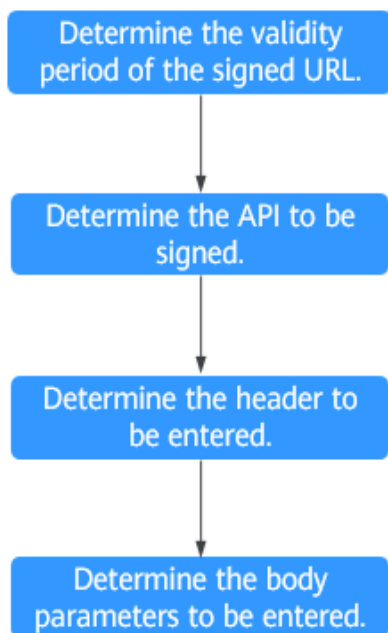
Operation	HTTP Request Method (Value in OBS SDK for Java)	Special Operator (Value in OBS SDK for Java)	Bucket Name Required (Yes/No)	Object Name Required (Yes/No)
Create Bucket	HttpMethodEnum.PUT	N/A	Yes	No
List Buckets	HttpMethodEnum.GET	N/A	No	No
Delete Bucket	HttpMethodEnum.DELETE	N/A	Yes	No
List Objects	HttpMethodEnum.GET	N/A	Yes	No
List Object Versions	HttpMethodEnum.GET	SpecialParamEnum.VERSIONS	Yes	No
List Multipart Uploads	HttpMethodEnum.GET	SpecialParamEnum.UPLOADS	Yes	No
Get Bucket Metadata	HttpMethodEnum.HEAD	N/A	Yes	No
Get Bucket Location	HttpMethodEnum.GET	SpecialParamEnum.LOCATION	Yes	No
Get Bucket Storageinfo	HttpMethodEnum.GET	SpecialParamEnum.STORAGEINFO	Yes	No
Set Bucket Quota	HttpMethodEnum.PUT	SpecialParamEnum.QUOTA	Yes	No
Get Bucket Quota	HttpMethodEnum.GET	SpecialParamEnum.QUOTA	Yes	No
Set Bucket Storage Policy	HttpMethodEnum.PUT	SpecialParamEnum.STORAGEPOLICY	Yes	No
Get Bucket Storage Policy	HttpMethodEnum.GET	SpecialParamEnum.STORAGEPOLICY	Yes	No
Configure Bucket ACL	HttpMethodEnum.PUT	SpecialParamEnum.ACL	Yes	No
Obtain Bucket ACL	HttpMethodEnum.GET	SpecialParamEnum.ACL	Yes	No

Operation	HTTP Request Method (Value in OBS SDK for Java)	Special Operator (Value in OBS SDK for Java)	Bucket Name Required (Yes/No)	Object Name Required (Yes/No)
Set Bucket Logging	HttpMethodEnum.PUT	SpecialParamEnum.LOGGING	Yes	No
Get Bucket Logging	HttpMethodEnum.GET	SpecialParamEnum.LOGGING	Yes	No
Set Bucket Policy	HttpMethodEnum.PUT	SpecialParamEnum.POLICY	Yes	No
Get Bucket Policy	HttpMethodEnum.GET	SpecialParamEnum.POLICY	Yes	No
Delete Bucket Policy	HttpMethodEnum.DELETE	SpecialParamEnum.POLICY	Yes	No
Set Lifecycle Rule	HttpMethodEnum.PUT	SpecialParamEnum.LIFECYCLE	Yes	No
Get Lifecycle Rule	HttpMethodEnum.GET	SpecialParamEnum.LIFECYCLE	Yes	No
Delete Lifecycle Rule	HttpMethodEnum.DELETE	SpecialParamEnum.LIFECYCLE	Yes	No
Set Website Hosting	HttpMethodEnum.PUT	SpecialParamEnum.WEBSITE	Yes	No
Get Website Hosting	HttpMethodEnum.GET	SpecialParamEnum.WEBSITE	Yes	No
Delete Website Hosting	HttpMethodEnum.DELETE	SpecialParamEnum.WEBSITE	Yes	No
Set Bucket Versioning	HttpMethodEnum.PUT	SpecialParamEnum.VERSIONING	Yes	No
Get Bucket Versioning	HttpMethodEnum.GET	SpecialParamEnum.VERSIONING	Yes	No
Set CORS Rule	HttpMethodEnum.PUT	SpecialParamEnum.CORS	Yes	No
Get CORS Rule	HttpMethodEnum.GET	SpecialParamEnum.CORS	Yes	No
Delete CORS Rule	HttpMethodEnum.DELETE	SpecialParamEnum.CORS	Yes	No

Operation	HTTP Request Method (Value in OBS SDK for Java)	Special Operator (Value in OBS SDK for Java)	Bucket Name Required (Yes/No)	Object Name Required (Yes/No)
Upload Object	HttpMethodEnum.PUT	N/A	Yes	Yes
Append Object	HttpMethodEnum.POST	SpecialParamEnum.APPEND	Yes	Yes
Get Object	HttpMethodEnum.GET	N/A	Yes	Yes
Copy Object	HttpMethodEnum.PUT	N/A	Yes	Yes
Delete Object	HttpMethodEnum.DELETE	N/A	Yes	Yes
Delete Objects	HttpMethodEnum.POST	SpecialParamEnum.DELETE	Yes	Yes
Get Object Metadata	HttpMethodEnum.HEAD	N/A	Yes	Yes
Configure Object ACL	HttpMethodEnum.PUT	SpecialParamEnum.ACL	Yes	Yes
Obtain Object ACL	HttpMethodEnum.GET	SpecialParamEnum.ACL	Yes	Yes
Initiate Multipart Upload	HttpMethodEnum.POST	SpecialParamEnum.UPLOADS	Yes	Yes
Upload Part	HttpMethodEnum.PUT	N/A	Yes	Yes
Copy Part	HttpMethodEnum.PUT	N/A	Yes	Yes
List Parts	HttpMethodEnum.GET	N/A	Yes	Yes
Assemble Parts	HttpMethodEnum.POST	N/A	Yes	Yes
Delete Multipart Upload	HttpMethodEnum.DELETE	N/A	Yes	Yes
Restore Cold Objects	HttpMethodEnum.POST	SpecialParamEnum.RESTORE	Yes	Yes

Process

To access OBS using a signed URL generated by the OBS SDK for Java, perform the following steps:



Procedure	Code Example
Check the validity period of the signed URL.	Set the expires parameter to determine the validity period of the URL.

Procedure	Code Example
Determine the API that you want to access using the signed URL. For details about APIs that support signed URLs, see APIs that support URLs. After determining the API, you can determine the HttpMethodEnum and SpecialParamEnum parameters. For example, if you want to set the object ACL using the signed URL, set HttpMethodEnum to HttpMethodEnum.PUT and SpecialParamEnum to SpecialParamEnum.ACL.	The following code sets the ACL of exampleobject in the bucket examplebucke: <pre>TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.PUT, expireSeconds); request.setBucketName("examplebucke"); request.setObjectKey("exampleobject"); request.setSpecialParam(SpecialParamEnum.ACL);</pre> For details about the complete code, see Code Example: Setting an Object ACL Using a Signed URL.
Confirm the body that needs to be included when you call the API.	The following code sets the ACL of exampleobject in the bucket examplebucke. The body contains the ACL configuration information. <pre>Request httpRequest = builder.url(response.getSignedUrl()) .put(RequestBody.create(null, ("<AccessControlPolicy>" + "<Owner><ID>" + OwnerId + "</ID></Owner>" + "<Delivered>true</Delivered>" + "<AccessControlList>" + "<Grant>" + "<Grantee><ID>" + OwnerId + "</ID></Grantee>" + "<Permission>FULL_CONTROL</Permission>" + "</Grant>" + "</AccessControlList>" + "</AccessControlPolicy>").getBytes())) .build();</pre> For details about the complete code, see Code Example: Setting an Object ACL Using a Signed URL.

Method

obsClient.createTemporarySignature([TemporarySignatureRequest request](#))

 NOTE

The **ObsClient.createTemporarySignature** function has encoded the generated URL. The browser will automatically decode the URL. You do not need to encode it again.

Request Parameters

Table 11-2 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	TemporarySignatureRequest	Yes	Explanation: Request parameters for creating a signed URL. For details, see Table 11-3 .

Table 11-3 TemporarySignatureRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	No	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	No	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
specialParam	SpecialParamEnum	No	Explanation: Special parameters that may be used in the request, indicating the sub-operations. Value range: See Table 11-1 . Default value: None
method	HttpMethodEnum	Yes	Explanation: HTTP method type. Value range: See Table 11-1 . Default value: None
headers	Map<String, String>	No	Explanation: Headers in the request. In Map , the String key and value indicate the name and value of the request header respectively. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
queryParams	Map<String, Object>	No	Explanation: Query parameters in the request. In Map , the String key indicates the name of the query parameter, and the Object value indicates the value of the query parameter. Default value: None
expires	long	Yes	Explanation: Expiration time of the signed URL. Value range: - If you use a temporary key to call the SDK API, the value of expires ranges from 0 to 86400 seconds. - If you use a permanent key to call the SDK API, the value of expires ranges from 0 to 630720000 seconds. Default value: 300
requestDate	Date	Yes	Explanation: Time when the request is initiated. Default value: None

Responses

Table 11-4 TemporarySignatureResponse

Parameter	Type	Description
signedUrl	String	Explanation: The signed URL that carries the authentication information. Default value: None
actualSignedRequestHeaders	Map<String, String>	Explanation: Actual headers in the request initiated using the signed URL. In Map , the String key and value indicate the name and value of the request header respectively. Default value: None

Code Example: Creating a Bucket

This example uses **HttpMethodEnum.PUT** to create bucket **examplebucket** based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import java.util.Map;
public class CreateBucket001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
```

```
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the validity period of the URL to 3600 seconds.
    long expireSeconds = 3600L;
    TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.PUT,
expireSeconds);
    request.setBucketName("examplebucket");
    TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
    System.out.println("Creating bucket using temporary signature url:");
    System.out.println("\t" + response.getSignedUrl());
    Request.Builder builder = new Request.Builder();
    for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
        builder.header(entry.getKey(), entry.getValue());
    }
    // Make a PUT request to create a bucket.
    String location = "your bucket location";
    Request httpRequest =
        builder.url(response.getSignedUrl())
            .put(
                RequestBody.create(
                    null,
                    ("<CreateBucketConfiguration><Location>"
                     + location
                     + "</Location></CreateBucketConfiguration>").getBytes())
            .build());
    OkHttpClient httpClient =
        new OkHttpClient.Builder()
            .followRedirects(false)
            .retryOnConnectionFailure(false)
            .cache(null)
            .build();
    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    System.out.println("\tStatus:" + res.code());
    if (res.body() != null) {
        System.out.println("\tContent:" + res.body().string() + "\n");
    }
    res.close();
    System.out.println("CreateBucket successfully");
} catch (ObsException e) {
    System.out.println("CreateBucket failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("CreateBucket failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Uploading an Object

This example uses **HttpMethodEnum.PUT** to upload object **objectname** to bucket **examplebucket** based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import java.util.HashMap;
import java.util.Map;
public class PutObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the validity period of the URL to 3600 seconds.
            long expireSeconds = 3600L;
            Map<String, String> headers = new HashMap<>();
            String contentType = "text/plain";
            headers.put("Content-Type", contentType);
            TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.PUT,
            expireSeconds);
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname");
            request.setHeaders(headers);
            TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
            System.out.println("Creating object using temporary signature url:");
            System.out.println("\t" + response.getSignedUrl());
            Request.Builder builder = new Request.Builder();
            for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
                builder.header(entry.getKey(), entry.getValue());
            }
            // Make a PUT request to upload an object.
            Request httpRequest =
                builder.url(response.getSignedUrl())
                    .put(RequestBody.create(MediaType.parse(contentType), "Hello
OBS".getBytes("UTF-8")))
                    .build();
            OkHttpClient httpClient =
                new OkHttpClient.Builder()
```

```
        .followRedirects(false)
        .retryOnConnectionFailure(false)
        .cache(null)
        .build();
    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    System.out.println("Status:" + res.code());
    if (res.body() != null) {
        System.out.println("Content:" + res.body().string() + "\n");
    }
    res.close();
    System.out.println("PutObject successfully");
} catch (ObsException e) {
    System.out.println("PutObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("PutObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Downloading an Object

This example uses **HttpMethodEnum.GET** to download object **objectname** from bucket **examplebucket** based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import java.io.InputStream;
import java.util.Map;
public class GetObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        // String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
```



```
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the validity period of the URL to 3600 seconds.
    long expireSeconds = 3600L;
    TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.GET,
    expireSeconds);
    request.setBucketName("examplebucket");
    request.setObjectKey("objectname");
    TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
    System.out.println("Getting object using temporary signature url:");
    System.out.println("SignedUrl:" + response.getSignedUrl());
    Request.Builder builder = new Request.Builder();
    for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
        builder.header(entry.getKey(), entry.getValue());
    }
    // Make a GET request to download an object.
    Request httpRequest = builder.url(response.getSignedUrl()).get().build();
    OkHttpClient httpClient =
        new OkHttpClient.Builder()
            .followRedirects(false)
            .retryOnConnectionFailure(false)
            .cache(null)
            .build();
    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    System.out.println("Status:" + res.code());
    InputStream objectContent = null;
    if (res.body() != null) {
        objectContent = res.body().byteStream();
        System.out.println("Content:" + res.body().string() + "\n");
    }
    if(objectContent != null) {
        // objectContent is the file stream to download.
        // You can read the objectContent stream to download the file. If the stream is not read for a
        long time, it will be disconnected from the server.
    }
    res.close();
    System.out.println("getObject successfully");
} catch (ObsException e) {
    System.out.println("getObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Listing Objects

This example uses **HttpMethodEnum.GET** to list object in bucket **examplebucket** based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
```

```
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import java.util.Map;
public class ListObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the validity period of the URL to 3600 seconds.
            long expireSeconds = 3600L;
            TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.GET,
expireSeconds);
            request.setBucketName("examplebucket");
            TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
            System.out.println("Getting object list using temporary signature url:");
            System.out.println("\t" + response.getSignedUrl());
            Request.Builder builder = new Request.Builder();
            for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
                builder.header(entry.getKey(), entry.getValue());
            }
            // Make a GET request to obtain the object list.
            Request httpRequest = builder.url(response.getSignedUrl()).get().build();
            OkHttpClient httpClient =
                new OkHttpClient.Builder()
                    .followRedirects(false)
                    .retryOnConnectionFailure(false)
                    .cache(null)
                    .build();
            Call c = httpClient.newCall(httpRequest);
            Response res = c.execute();
            System.out.println("Status:" + res.code());
            if (res.body() != null) {
                System.out.println("Content:" + res.body().string() + "\n");
            }
            res.close();
            System.out.println("ListObject successfully");
        } catch (ObsException e) {
            System.out.println("ListObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
        }
    }
}
```

```
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("ListObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Deleting an Object

This example uses **HttpMethodEnum.DELETE** to delete object **objectname** from bucket **examplebucket** based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import java.util.Map;
public class DeleteObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the validity period of the URL to 3600 seconds.
            long expireSeconds = 3600L;
            TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.DELETE,
            expireSeconds);
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname");
            TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
            System.out.println("Deleting object using temporary signature url:");
            System.out.println("\t" + response.getSignedUrl());
            Request.Builder builder = new Request.Builder();
            for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
                builder.header(entry.getKey(), entry.getValue());
            }
            // Make a DELETE request to delete an object.
            Request httpRequest = builder.url(response.getSignedUrl()).delete().build();
```

```
OkHttpClient httpClient =
    new OkHttpClient.Builder()
        .followRedirects(false)
        .retryOnConnectionFailure(false)
        .cache(null)
        .build();
Call c = httpClient.newCall(httpRequest);
Response res = c.execute();
System.out.println("\tStatus:" + res.code());
if (res.body() != null) {
    System.out.println("\tContent:" + res.body().string() + "\n");
}
res.close();
System.out.println("deleteObjects successfully");
} catch (ObsException e) {
    System.out.println("deleteObjects failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteObjects failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Initiating a Multipart Upload

This example uses **HttpMethodEnum.POST** to initiate a multipart upload based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.SpecialParamEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import java.util.Map;
public class InitiateMultiPartUpload001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");
    }
}
```

```
// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the validity period of the URL to 3600 seconds.
    long expireSeconds = 3600L;
    TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.POST,
    expireSeconds);
    request.setBucketName("examplebucket");
    request.setObjectKey("objectname");
    request.setSpecialParam(SpecialParamEnum.UPLOADS);
    TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
    System.out.println("initiate multipart upload using temporary signature url:");
    System.out.println("\t" + response.getSignedUrl());
    Request.Builder builder = new Request.Builder();
    for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
        builder.header(entry.getKey(), entry.getValue());
    }
    // Make a POST request to initiate the multipart upload.
    Request httpRequest = builder.url(response.getSignedUrl()).post(RequestBody.create(null,
    "")).build();
    OkHttpClient httpClient =
        new OkHttpClient.Builder()
            .followRedirects(false)
            .retryOnConnectionFailure(false)
            .cache(null)
            .build();
    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    System.out.println("Status:" + res.code());
    if (res.body() != null) {
        System.out.println("Content:" + res.body().string() + "\n");
    }
    res.close();
    System.out.println("InitiateMultiPartUpload successfully");
} catch (ObsException e) {
    System.out.println("InitiateMultiPartUpload failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("InitiateMultiPartUpload failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Uploading a Part

This example uses **HttpMethodEnum.PUT** to upload a part based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
```

```
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import java.util.HashMap;
import java.util.Map;
public class UploadPart001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the validity period of the URL to 3600 seconds.
            long expireSeconds = 3600L;
            TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.PUT,
            expireSeconds);
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname");
            Map<String, Object> queryParams = new HashMap<String, Object>();
            // Set the partNumber parameter, for example, queryParams.put("partNumber", "1").
            queryParams.put("partNumber", "partNumber");
            queryParams.put("uploadId", "your uploadId");
            request.setQueryParams(queryParams);
            TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
            System.out.println("upload part using temporary signature url:");
            System.out.println("SignedUrl:" + response.getSignedUrl());
            Request.Builder builder = new Request.Builder();
            for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
                builder.header(entry.getKey(), entry.getValue());
            }
            // Make a PUT request to upload a part.
            Request httpRequest =
                builder.url(response.getSignedUrl())
                    .put(RequestBody.create(null, new byte[6 * 1024 * 1024]))
                    .build();
            OkHttpClient httpClient =
                new OkHttpClient.Builder()
                    .followRedirects(false)
                    .retryOnConnectionFailure(false)
                    .cache(null)
                    .build();
            Call c = httpClient.newCall(httpRequest);
            Response res = c.execute();
            System.out.println("Status:" + res.code());
            if (res.body() != null) {
                System.out.println("Content:" + res.body().string() + "\n");
            }
            res.close();
            System.out.println("UploadPart successfully");
        } catch (ObsException e) {
```

```
        System.out.println("UploadPart failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("UploadPart failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

Code Example: Listing Uploaded Parts

This example uses **HttpMethodEnum.GET** to list uploaded parts based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.Response;
import java.util.HashMap;
import java.util.Map;
public class ListUploadedParts001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the validity period of the URL to 3600 seconds.
            long expireSeconds = 3600L;
            TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.GET,
            expireSeconds);
            request.setBucketName("examplebucket");
            request.setObjectKey("objectname");
            Map<String, Object> queryParams = new HashMap<String, Object>();
            queryParams.put("uploadId", "your uploadId");
            request.setQueryParams(queryParams);
```

```
TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
System.out.println("list parts using temporary signature url:");
System.out.println("\t" + response.getSignedUrl());
Request.Builder builder = new Request.Builder();
for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
    builder.header(entry.getKey(), entry.getValue());
}
// Make a GET request to list uploaded parts.
Request httpRequest = builder.url(response.getSignedUrl()).get().build();
OkHttpClient httpClient =
    new OkHttpClient.Builder()
        .followRedirects(false)
        .retryOnConnectionFailure(false)
        .cache(null)
        .build();
Call c = httpClient.newCall(httpRequest);
Response res = c.execute();
System.out.println("Status:" + res.code());
if (res.body() != null) {
    System.out.println("Content:" + res.body().string() + "\n");
}
res.close();
System.out.println("ListParts successfully");
} catch (ObsException e) {
    System.out.println("ListParts failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("ListParts failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Assembling Parts

This example uses **HttpMethodEnum.POST** to assemble parts based on a signed URL. The validity period of the URL is 3,600 seconds.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.MediaType;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import java.util.HashMap;
import java.util.Map;
public class CompleteMultiPartUpload001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
```


advised not to use hard coding, which may result in information leakage.

// Obtain an AK/SK pair and a security token using environment variables or import them in other ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the following line.

```
// String securityToken = System.getenv("SECURITY_TOKEN");  
// Enter the endpoint corresponding to the region where the bucket is located.  
String endPoint = "https://your-endpoint";  
// Obtain an endpoint using environment variables or import it in other ways.  
//String endPoint = System.getenv("ENDPOINT");
```

```
// Create an ObsClient instance.  
// Use the permanent AK/SK pair to initialize the client.  
ObsClient obsClient = new ObsClient(ak, sk, endPoint);  
// Use the temporary AK/SK pair and security token to initialize the client.  
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
```

```
try {  
    // Set the validity period of the URL to 3600 seconds.  
    long expireSeconds = 3600L;  
    TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.POST,  
expireSeconds);  
    request.setBucketName("examplebucket");  
    request.setObjectKey("objectname");  
    Map<String, String> headers = new HashMap<>();  
    String contentType = "application/xml";  
    headers.put("Content-Type", contentType);  
    request.setHeaders(headers);  
    Map<String, Object> queryParams = new HashMap<>();  
    queryParams.put("uploadId", "your uploadId");  
    request.setQueryParams(queryParams);  
    TemporarySignatureResponse response = obsClient.createTemporarySignature(request);  
    System.out.println("complete multipart upload using temporary signature url:");  
    System.out.println("\t" + response.getSignedUrl());  
    Request.Builder builder = new Request.Builder();  
    for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {  
        builder.header(entry.getKey(), entry.getValue());  
    }  
    // The following content is used as an example. You need to replace it with the responses of the  
request for listing uploaded parts.  
    String content = "<CompleteMultipartUpload>";  
    content += "<Part>";  
    content += "<PartNumber>1</PartNumber>";  
    content += "<ETag>da6a0d097e307ac52ed9b4ad551801fc</ETag>";  
    content += "</Part>";  
    content += "<Part>";  
    content += "<PartNumber>2</PartNumber>";  
    content += "<ETag>da6a0d097e307ac52ed9b4ad551801fc</ETag>";  
    content += "</Part>";  
    content += "</CompleteMultipartUpload>";  
    // Make a POST request to assemble the uploaded parts.  
    Request httpRequest =  
        builder.url(response.getSignedUrl())  
            .post(RequestBody.create(MediaType.parse(contentType), content.getBytes("UTF-8")))  
            .build();  
    OkHttpClient httpClient =  
        new OkHttpClient.Builder()  
            .followRedirects(false)  
            .retryOnConnectionFailure(false)  
            .cache(null)  
            .build();  
    Call c = httpClient.newCall(httpRequest);  
    Response res = c.execute();  
    System.out.println("\tStatus:" + res.code());  
    if (res.body() != null) {  
        System.out.println("\tContent:" + res.body().string() + "\n");  
    }  
    res.close();  
} catch (ObsException e) {  
    System.out.println("CompleteMultiPartUpload failed");
```

```
// Request failed. Print the HTTP status code.
System.out.println("HTTP Code:" + e.getResponseCode());
// Request failed. Print the server-side error code.
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("CompleteMultiPartUpload failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

 NOTE

- **HttpMethodEnum** is an enumeration function defined in OBS SDK for Java, whose value indicates the request method types.

Integrity Check When Uploading an Object

This example uses content-md5 for integrity check when using a temporary URL to upload an object to OBS.

```
import com.obs.services.ObsClient;
import com.obs.services.internal.utils.ServiceUtils;
import com.obs.services.model.*;
import okhttp3.*;
import java.io.ByteArrayOutputStream;
import java.io.File;
import java.io.FileInputStream;
import java.io.IOException;
import java.security.NoSuchAlgorithmException;
import java.util.HashMap;
import java.util.Map;
public class Create_TemporarySignature_Uploadobject {
    public static void main(String[] args) throws IOException, NoSuchAlgorithmException {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using hard
        // coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
        // following line.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");
        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
        // Set the validity period of the URL to 3600 seconds.
        long expireSeconds = 3600L;
        Map<String, String> headers = new HashMap<String, String>();
        String localFile_path = "localFile_path";
        //Convert the local file into a stream and calculate its MD5 value.
        try (FileInputStream fileInputStream = new FileInputStream(localFile_path)) {
            String md5 = ServiceUtils.toBase64(ServiceUtils.computeMD5Hash(fileInputStream));
```

```
        headers.put("content-md5",md5);
    }
    TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.PUT,
expireSeconds);
    request.setBucketName("your_bucketname");
    request.setObjectKey("objectkey");
    request.setHeaders(headers);
    TemporarySignatureResponse response = obsClient.createTemporarySignature(request);
    System.out.println("Creating object using temporary signature url:");
    System.out.println("\t" + response.getSignedUrl());
    Request.Builder builder = new Request.Builder();
    for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
        builder.header(entry.getKey(), entry.getValue());
    }
    // Make a PUT request to upload the file.
    Request httpRequest = builder.url(response.getSignedUrl())
        .put(RequestBody.create(getBytesByFile(localFile_path))).build();
    OkHttpClient httpClient = new
OkHttpClient.Builder().followRedirects(false).retryOnConnectionFailure(false)
        .cache(null).build();
    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    System.out.println("\tStatus:" + res.code());
    if (res.body() != null) {
        System.out.println("\tContent:" + res.body().string() + "\n");
    }
    res.close();
}
public static byte[] getBytesByFile(String pathStr) {
    File file = new File(pathStr);
    try (FileInputStream fis = new FileInputStream(file);
        ByteArrayOutputStream bos = new ByteArrayOutputStream(1000)) {
        byte[] b = new byte[1000];
        int n;
        while ((n = fis.read(b)) != -1) {
            bos.write(b, 0, n);
        }
        return bos.toByteArray();
    } catch (Exception e) {
        e.printStackTrace();
    }
    return null;
}
}
```

Code Example: Setting an Object ACL Using a Signed URL

The following code sets the ACL of **exampleobject** in the bucket **examplebucket** to **bucket-owner-full-control**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.HttpMethodEnum;
import com.obs.services.model.SpecialParamEnum;
import com.obs.services.model.TemporarySignatureRequest;
import com.obs.services.model.TemporarySignatureResponse;
import okhttp3.Call;
import okhttp3.OkHttpClient;
import okhttp3.Request;
import okhttp3.RequestBody;
import okhttp3.Response;
import java.util.HashMap;
import java.util.Map;
public class SetObjectAcl001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
```

```
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways. If you use a temporary AK/SK pair and a security token to make an API call, uncomment the
// following line.
//String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is located.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use a permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use a temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the validity period of the URL to 3600 seconds.
    long expireSeconds = 3600L;
    TemporarySignatureRequest request = new TemporarySignatureRequest(HttpMethodEnum.PUT,
    expireSeconds);
    request.setBucketName("examplebucket");
    request.setObjectKey("exampleobject");
    request.setSpecialParam(SpecialParamEnum.ACL);

    TemporarySignatureResponse response = obsClient.createTemporarySignature(request);

    System.out.println("Set object ACL using temporary signature url:");
    System.out.println("\t" + response.getSignedUrl());
    Request.Builder builder = new Request.Builder();
    for (Map.Entry<String, String> entry : response.getActualSignedRequestHeaders().entrySet()) {
        builder.header(entry.getKey(), entry.getValue());
    }

    // Make a PUT request to upload an object.
    String OwnerId = "5b690*****abc";
    Request httpRequest = builder.url(response.getSignedUrl())
        .put(RequestBody.create(null, ("<AccessControlPolicy>"
            + "<Owner><ID>"
            + OwnerId
            + "</ID></Owner>"
            + "<Delivered>true</Delivered>"
            + "<AccessControlList>"
            + "<Grant>"
            + "<Grantee><ID>"
            + OwnerId
            + "</ID></Grantee>"
            + "<Permission>FULL_CONTROL</Permission>"
            + "</Grant>"
            + "</AccessControlList>"
            + "</AccessControlPolicy>").getBytes()))
        .build();
    OkHttpClient httpClient = new OkHttpClient.Builder().followRedirects(false)
        .retryOnConnectionFailure(false)
        .cache(null)
        .build();

    Call c = httpClient.newCall(httpRequest);
    Response res = c.execute();
    System.out.println("Status:" + res.code());
    if (res.body() != null) {
        System.out.println("Content:" + res.body().string() + "\n");
    }
    res.close();
    System.out.println("SetObjectAcl successfully");
} catch (ObsException e) {
    System.out.println("SetObjectAcl failed");
}
```

```
        // Request failed. Print the HTTP status code.  
        System.out.println("HTTP Code:" + e.getResponseCode());  
        // Request failed. Print the server-side error code.  
        System.out.println("Error Code:" + e.getErrorCode());  
        // Request failed. Print the error details.  
        System.out.println("Error Message:" + e.getErrorMessage());  
        // Request failed. Print the request ID.  
        System.out.println("Request ID:" + e.getErrorRequestId());  
        System.out.println("Host ID:" + e.getErrorHostId());  
        e.printStackTrace();  
    } catch (Exception e) {  
        System.out.println("SetObjectAcl failed");  
        // Print other error information.  
        e.printStackTrace();  
    }  
}
```

12 Versioning (SDK for Java)

12.1 Overview (SDK for Java)

You can use versioning to store multiple versions of an object in a bucket.

When versioning is enabled for a bucket, OBS keeps multiple versions of an object in the bucket, allowing you to easily retrieve and restore historical versions or recover data in the event of accidental changes or application failures.

By default, versioning is disabled for new OBS buckets. In this case, if a newly uploaded object is using the name of the previously uploaded one, the new object will overwrite the previous one.

12.2 Configuring Versioning for a Bucket (SDK for Java)

Function

You can enable versioning to automatically maintain previous versions of an object. When versioning is enabled, you can access earlier versions of an object to recover your data in the event of accidental actions or application failures.

This API configures the versioning status for a bucket.

Restrictions

- To configure versioning for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketVersioning** in IAM or **PutBucketVersioning** in a bucket policy).

Method

obsClient.setBucketVersioning(final [SetBucketVersioningRequest request](#))

Request Parameters

Table 12-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetBucketVersioningRequest	Yes	Explanation: Request parameters for bucket versioning configuration. For details, see Table 12-2 .

Table 12-2 SetBucketVersioningRequest

Parameter	Type	Man dato ry (Yes /No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
status	VersioningSt atusEnum	Yes	Explanation: Versioning status of the bucket. Value range: See Table 12-3 .

Table 12-3 VersioningStatusEnum

Constant	Default Value	Description
SUSPENDED	Suspended	Versioning is suspended.
ENABLED	Enabled	Versioning is enabled.

Responses

Table 12-4 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example enables and suspends versioning of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketVersioningConfiguration;
import com.obs.services.model.VersioningStatusEnum;
public class SetBucketVersioning001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
```

```
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Enable versioning.
    obsClient.setBucketVersioning(
        "examplebucket", new BucketVersioningConfiguration(VersioningStatusEnum.ENABLED));
    // Suspend versioning.
    obsClient.setBucketVersioning(
        "examplebucket", new BucketVersioningConfiguration(VersioningStatusEnum.SUSPENDED));
    System.out.println("setBucketVersioning successfully");
} catch (ObsException e) {
    System.out.println("setBucketVersioning failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketVersioning failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

12.3 Obtaining the Versioning Status of a Bucket (SDK for Java)

Function

You can enable versioning to automatically maintain previous versions of an object. When versioning is enabled, you can access earlier versions of an object to recover your data in the event of accidental actions or application failures.

This API obtains the versioning status of a bucket.

Restrictions

- To view the versioning status of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketVersioning** in IAM or **GetBucketVersioning** in a bucket policy).

Method

obsClient.getBucketVersioning(final [BaseBucketRequest request](#))

Request Parameters

Table 12-5 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters for viewing the versioning status of a bucket. For details, see Table 12-6 .

Table 12-6 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 12-7 BucketVersioningConfiguration

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
status	VersioningStatusEnum	Explanation: Versioning status of the bucket. Value range: See Table 12-8 .

Table 12-8 VersioningStatusEnum

Constant	Default Value	Description
SUSPENDED	Suspended	Versioning is suspended.
ENABLED	Enabled	Versioning is enabled.

Code Examples

This example views the versioning status of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketVersioningConfiguration;
public class GetBucketVersioning001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
```

```
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // View the versioning status of a bucket.
    BucketVersioningConfiguration status = obsClient.getBucketVersioning("examplebucket");
    System.out.println("getBucketVersioning successfully");
    System.out.println("getVersioningStatus:" + status.getVersioningStatus());
} catch (ObsException e) {
    System.out.println("getBucketVersioning failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getBucketVersioning failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

12.4 Obtaining an Object Version (SDK for Java)

Function

You can call **ObsClient.getObject** to obtain an object version by specifying the version ID (**versionId**).

Restrictions

- To download an object, you must be the bucket owner or have the required permission (**obs:object:GetObject** in IAM or **GetObject** in a bucket policy).
- Objects in the storage class can be downloaded only when they are restored.

Method

obsClient.getObject([GetObjectRequest request](#))

Request Parameters

Table 12-9 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectRequest	Yes	Explanation: Request parameters for downloading an object. For details, see Table 12-10 .

Table 12-10 GetObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. If version ID is left blank, the latest version of the object will be downloaded. Value range: The value must contain 32 characters. Default value: None
rangeStart	Long	No	Explanation: Start position for object download. Value range: A non-negative integer. Default value: None
rangeEnd	Long	No	Explanation: End position for object download. Restrictions: The upper limit of the value is the object length minus 1, in bytes. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
IfModifiedSince	Date	No	Explanation: The object is returned if it has been modified since the specified time; otherwise, an error is returned. Default value: None
IfUnmodifiedSince	Date	No	Explanation: The object is returned if it has not been modified since the specified time; otherwise, an error is returned. Default value: None
replaceMetadata	ObjectRepleace Metadata	No	Explanation: Additional information about the object. For details, see Table 12-11.
progressListener	ProgressListener	No	Explanation: Data transmission listener for obtaining the download progress. For details, see Table 12-12.

Table 12-11 ObjectRepleaceMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: Expires header in the response. It specifies the cache expiration time of the web page when the object is downloaded. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object in a download. Default value: None

Table 12-12 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 12-13 . Default value: None

Table 12-13 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 12-14 . Default value: None

Table 12-14 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneousSpeed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 12-15 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-16 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 12-17 ObjectMetadata

Parameter	Type	Mandatory (Yes/No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the file data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 12-16. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Responses

Table 12-18 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.) - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 12-15.

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 12-17 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Code Examples

This example sets the version ID to obtain object versions and views the versioning status of object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsObject;
public class GetObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Set the version ID to obtain object versions.
            ObsObject obsObject = obsClient.getObject("examplebucket", "objectname", "versionid1");
            System.out.println("getObject successfully");
            obsObject.getObjectContent().close();
        } catch (ObsException e) {
            System.out.println("getObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("getObject failed");
            // Print other error information.
        }
    }
}
```

```
e.printStackTrace();  
    }  
}  
}
```

12.5 Copying an Object Version (SDK for Java)

Function

This API copies an object version in a specified bucket.

You can call **ObsClient.copyObject** to copy an object version by specifying the version ID (**versionId**).

Restrictions

- To copy an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- You must have the read permission on the source object.
- The object copy request carries the information about the source bucket and object to be copied in the header field. The message body cannot be carried.
- Cross-bucket replication in the same region is supported, but cross-region replication is not supported.
- An object copy can be up to 5 GB in size. If the source object exceeds 5 GB, you can only perform a [multipart copy](#).
- If the source object is in the Cold storage class, you must restore it first.

Method

obsClient.copyObject([CopyObjectRequest request](#))

Request Parameters

Table 12-19 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	CopyObjectRequest	Yes	Explanation: Request parameters for copying an object. For details, see Table 12-20 .

Table 12-20 CopyObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Target bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Target object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
acl	AccessControlList	No	Explanation: ACL that can be pre-defined during the object copy. For details, see Table 12-21. Default value: None
successRedirect-Location	String	No	Explanation: Address (URL) to which a successfully answered request is redirected. - If the value is valid and the request is successful, OBS returns status code 303. Location contains SuccessActionRedirect as well as the bucket name, object name, and object ETag. - If the value is invalid, OBS ignores this parameter. In such case, the Location header is the object address, and OBS returns a status code based on whether the operation succeeds or fails. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
sourceBucketName	String	Yes	Explanation: Name of the source bucket. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
sourceObjectKey	String	Yes	Explanation: Source object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
newObjectMetadata	ObjectMetadata	No	Explanation: User-defined metadata of the target object. For details, see Table 12-28. Restrictions: newObjectMetadata must be used together with replaceMetadata. Default value: None
replaceMetadata	boolean	No	Explanation: Whether to rewrite the metadata of the source object. Restrictions: replaceMetadata must be used together with newObjectMetadata. Value range: true: The metadata of the source object is rewritten. false: The metadata of the source object is not rewritten. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifModifiedSince	java.util.Date	No	Explanation: The source object is copied if it has been modified since the specified time; otherwise, an exception is thrown. Default value: None
ifUnmodifiedSince	java.util.Date	No	Explanation: If the source object has not been modified since the specified time, it is copied. Otherwise, an exception is thrown. Default value: None
ifMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is the same as the preset ETag, the object is returned. Otherwise, an error is returned. The ETag of the source object is the MD5 value of it. Restrictions: - If the object copy request includes ifUnmodifiedSince, ifMatchTag, ifModifiedSince, or ifNoneMatchTag, and the specified condition is not met, the copy will fail and an exception will be thrown with HTTP status code 412 precondition failed returned. ifModifiedSince and ifNoneMatchTag can be used together. So do ifUnmodifiedSince and ifMatchTag. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
ifNoneMatchTag	String	No	Explanation: Preset ETag. If the ETag of the object to be downloaded is different from the preset ETag, the object is returned. Otherwise, an error is returned. Value range: The value must contain 32 characters. Default value: None
versionId	String	No	Explanation: Version ID of the source object. Value range: The value must contain 32 characters. Default value: None

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 12-21 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Description
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 12-22 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 12-23 .

Table 12-22 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-23 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 12-25 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 12-24 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 12-24 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 12-25 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 12-26 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 12-27 . Default value: None

Table 12-26 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-27 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 12-28 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the file data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 12-29. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 12-29 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Responses

Table 12-30 CopyObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
lastModified	java.util.Date	Explanation: Last time the target object was modified. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of the target object. Base64-encoded, 128-bit MD5 value of an object. ETag is the unique identifier of the object contents and is used to determine whether the contents of an object are changed. For example, if the ETag value is A when an object is uploaded and is B when the object is downloaded, this indicates the contents of the object are changed. The ETag reflects changes only to the contents of an object, not its metadata. Objects created by the upload and copy operations have unique ETags after being encrypted using MD5. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
versionId	String	Explanation: Version ID of the target object. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
copySourceVersionId	String	Explanation: Version ID of the source object. Value range: The value must contain 32 characters. Default value: None
storageClass	StorageClassEnum	Explanation: Storage class of the target object. Value range: See Table 12-29 . Default value: None

Code Examples

This example sets a version ID to copy object versions of **sourceobjectname** from bucket **sourceexamplebucket** to another bucket.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.CopyObjectRequest;
public class CopyObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk,endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Copy object versions.
            CopyObjectRequest request = new CopyObjectRequest();
            request.setSourceBucketName("sourceexamplebucket");
            request.setSourceObjectKey("sourceobjectname");
            // Set the version ID of the object to be copied.
            request.setVersionId("versionid");
```

```
request.setDestinationBucketName("destexamplebucket");
request.setDestinationObjectKey("destobjectname");
obsClient.copyObject(request);
System.out.println("copyObject successfully");
} catch (ObsException e) {
    System.out.println("copyObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("copyObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

12.6 Restoring a Specific Cold Object Version (SDK for Java)

Function

This API downloads a Cold object. To download such an object, you must restore it first. For the options for the restore speed, see [Table 12-33](#).

You can call **ObsClient.restoreObject** to restore a Cold object version by specifying the **versionId**.

Restrictions

- To restore object, you must be the bucket owner or have the required permission (**obs:object:RestoreObject** in IAM or **RestoreObject** in a bucket policy.)
- The object specified in **ObsClient.restoreObject** must be in the Cold storage class. Otherwise, an exception will be thrown when you call this API.

Method

obsClient.restoreObject([RestoreObjectRequest request](#))

Request Parameters

Table 12-31 List of request parameters

Parameter	Type	Mandator y (Yes/No)	Description
request	RestoreObjectRequest	Yes	Explanation: Request parameters for restoring a Cold object version. For details, see Table 12-32 .

Table 12-32 RestoreObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (.), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Value range: The value must contain 32 characters. Default value: None. If this parameter is left blank, the latest version of the object is specified.
days	int	Yes	Explanation: After an object is restored, a Standard copy is generated for the object. This parameter specifies how long the Standard copy can be retained, that is, the validity period of the restored object. Restrictions: The value must be a positive integer. Value range: The value ranges from 1 to 30, in days. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
tier	RestoreTierEnum	No	Explanation: The restore option, which indicates the time spent on restoring the object. Value range: For details, see Table 12-33 . Default value: Standard

Table 12-33 RestoreTierEnum

Constant	Default Value	Description
EXPEDITED	Expedited	Objects can be restored at an expedited speed within 1 to 5 minutes.
STANDARD	Standard	Objects can be restored at a standard speed within 3 to 5 hours.

Responses

Table 12-34 RestoreObjectStatus

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example restores object version **objectname** in bucket **examplebucket** at an expedited speed and retains the restored object for one day.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsObject;
import com.obs.services.model.RestoreObjectRequest;
import com.obs.services.model.RestoreTierEnum;
public class RestoreObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Restore an object version at an expedited speed.
            RestoreObjectRequest request = new RestoreObjectRequest("examplebucket", "objectname", 1);
            request.setRestoreTier(RestoreTierEnum.EXPEDITED);
            request.setVersionId("versionid");
            obsClient.restoreObject(request);
            System.out.println("RestoreObject successfully");
        } catch (ObsException e) {
            System.out.println("RestoreObject failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        }
    }
}
```

```
    } catch (Exception e) {  
        System.out.println("RestoreObject failed");  
        // Print other error information.  
        e.printStackTrace();  
    }  
}
```

12.7 Listing Object Versions (SDK for Java)

Function

This API lists some or all of the object versions in a bucket. When listing the object versions, you can specify the criteria such as the prefix, number, and start position. Returned object versions are listed in alphabetical order by object name.

The returned results of **ObsClient.listVersions** include the object versions and delete markers.

Restrictions

- To list object versions in a bucket, you must be the bucket owner or have the required permission (**obs:bucket:ListBucketVersions** in IAM or **ListBucketVersions** in a bucket policy).
- A maximum of 1,000 object versions can be returned in a single request. If a bucket contains more than 1,000 object versions, you can use [pagination](#) to list all versions. If **ListVersionsResult.isTruncated** is **true**, only part of object versions are returned. In this case, you can use **ListVersionsResult.getNextKeyMarker** and **ListVersionsResult.getNextVersionIdMarker** to obtain the start position for the next listing.

Method

obsClient.listVersions([ListVersionsRequest request](#))

Request Parameters

Table 12-35 List of request parameters

Parameter	Type	Mandator y (Yes/No)	Description
request	ListVersionsRequest	Yes	Explanation: Request parameters for listing object versions in a bucket. For details, see Table 12-36 .

Table 12-36 ListVersionsRequest parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.). - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
prefix	String	No	Explanation: Prefix that the names of objects to list must contain. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
keyMarker	String	No	Explanation: Object name to start with when listing object versions in a bucket. All object versions following this parameter are listed in alphabetical order. Restrictions: This parameter is only available for listing objects with multiple versions. Value range: The value of nextKeyMarker in the response body of the last request. Default value: None
maxKeys	int	No	Explanation: The maximum number of object versions returned in the response in alphabetical order. Value range: The value ranges from 1 to 1000. If the specified value is beyond this range, only 1,000 objects are returned. Default value: 1000

Parameter	Type	Mandatory (Yes/No)	Description
delimiter	String	No	Explanation: This parameter is used to group object names. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one CommonPrefixes. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one CommonPrefixes. Assume that a bucket has objects abcd, abcde, and bbcde in it. If delimiter is set to d and prefix is set to a, objects abcd and abcde are grouped into a commonPrefix with abcd as the prefix. If only delimiter is set to d, objects abcd and abcde are grouped into a commonPrefix with abcd as the prefix, and bbcde is grouped separately into another commonPrefix with bbcd as the prefix. For a PFS, if this parameter is not specified, all the contents in the current directory are recursively listed by default, and subdirectories are also listed. In big data scenarios, PFSs usually have multiple directory levels and each directory level has a large number of files. In such case, you are advised to configure [delimiter=/] to list the contents in the current directory but exclude the contents in subdirectories, thereby accelerating the listing. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
versionIdMarker	String	No	Explanation: Version ID you want to start listing from. All object versions following the value specified by this parameter are listed in alphabetical order by object name and version ID. This parameter must be used together with keyMarker. keyMarker specifies the object name, and versionIdMarker specifies the version of the specified object. Restrictions: • This parameter is only available for listing objects with multiple versions. • If the object version specified by versionIdMarker and the object name specified by keyMarker do not match, versionIdMarker is invalid. Value range: Object version ID, that is, the value of nextVersionIdMarker in the response body of the last request. Default value: None

Responses

Table 12-37 ListVersionsResult

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Description
prefix	String	Explanation: Object name prefix. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you specify logs/ as the prefix, logs/day1, logs/day2, and logs/day3 will be returned. If this parameter is left blank and there are no other filtering criteria, all objects in the bucket will be returned. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionIdMarker	String	Explanation: Object version for the object specified by KeyMarker. All object versions following the value specified by this parameter are listed in alphabetical order by object name and version ID. This parameter must be used together with KeyMarker. KeyMarker specifies the object name, and versionIdMarker specifies the version of the specified object. Restrictions: • This parameter is only available for listing objects with multiple versions. • If the object version specified by versionIdMarker and the object name specified by keyMarker do not match, versionIdMarker is invalid. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
nextVersionIdMarker	String	Explanation: Version ID to start with in the next request for listing object versions. It must be used together with nextKeyMarker. If only part of the object versions are returned for the current request, this parameter is included in the response for your use in the subsequent request. Restrictions: This parameter is only available for listing objects with multiple versions. Value range: The value must contain 32 characters. Default value: None
objectSummaries	List<ObsObject>	Explanation: Object information. For details, see Table 12-38.
commonPrefixes	List<String>	Explanation: List of object name prefixes grouped according to the delimiter parameter (if specified) Value range: The value must contain 1 to 1,024 characters. Default value: None
truncated	boolean	Explanation: Whether all objects are returned in the response. A maximum of 1,000 objects can be listed at a time. If the number of objects is greater than 1,000, the objects beyond 1,000 cannot be returned. Value range: true: Not all objects are returned. false: All objects are returned. Default value: None

Parameter	Type	Description
keyMarker	String	Explanation: Object name to start listing from. All object versions following the value specified by this parameter are listed in alphabetical order by object name. Restrictions: This parameter is only available for listing objects with multiple versions. Value range: The value must contain 1 to 1,024 characters. Default value: None
maxKeys	int	Explanation: Maximum number of objects to list. No more than the specified number of objects can be returned in the response in alphabetical order. Value range: The value ranges from 1 to 1,000. If the specified value is beyond this range, only 1,000 objects are returned. Default value: 1000

Parameter	Type	Description
delimiter	String	Explanation: Object names are grouped by this parameter, which is often used with Prefix. If a prefix is specified, objects with the same string from the prefix to the first delimiter are grouped into one CommonPrefixes. If no prefix is specified, objects with the same string from the first character to the first delimiter are grouped into one CommonPrefixes. Assume that a bucket has objects abcd, abcde, and bbcde in it. If delimiter is set to d and prefix is set to a, files abcd and abcde are grouped into a CommonPrefixes with abcd as the prefix. If only delimiter is set to d, files abcd and abcde are grouped into a CommonPrefixes with abcd as the prefix, and bbcde is grouped separately into another CommonPrefixes with bbcde as the prefix. For a PFS, if this parameter is not specified, all the contents in the current directory are recursively listed by default, and subdirectories are also listed. In big data scenarios, PFSs usually have multiple directory levels and each directory level has a large number of files. In such case, you are advised to configure [delimiter=/] to list the contents in the current directory but exclude the contents in subdirectories, thereby accelerating the listing. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
nextKeyMarker	String	Explanation: Object name to start with for the next request for listing object versions. If only part of the object versions are returned for the current request, this parameter is included in the response for your use in the subsequent request. Restrictions: This parameter is only available for listing objects with multiple versions. Value range: The value must contain 1 to 1,024 characters. Default value: None
location	String	Explanation: Region where a bucket is located.
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
versions	VersionOrDeleteMarker[]	Explanation: Object version information. For details, see Table 12-43.

Table 12-38 ObsObject

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
owner	Owner	Explanation: Owner of the object. For details, see Table 12-39 .

Parameter	Type	Description
metadata	ObjectMetadata	Explanation: Object metadata. For details, see Table 12-40 .
objectContent	InputStream	Explanation: Object data stream. Default value: None

Table 12-39 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-40 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: - The object size in a single upload ranges from 0 to 5 GB. - To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml, the object is an application/xml file. If the suffix is .html, the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the file data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 12-42. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 12-41 ObjectTypeEnum

Constant	Default Value	Description
NORMAL	NORMAL	Normal objects
APPENDABLE	APPENDABLE	Appendable objects

Table 12-42 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 12-43 VersionOrDeleteMarker

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Description
key	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
isLatest	boolean	Explanation: Whether the object is the latest version. Value range: • true: The object is the latest version. • false: The object is not the latest version. Default value: false
lastModified	Date	Explanation: Time when the object was last modified. Value range: UTC time Default value: None

Parameter	Type	Description
owner	Owner	Explanation: User information, including the domain ID and name of the object owner. For details, see Table 12-39 .
etag	String	Explanation: Base64-encoded, 128-bit MD5 value of an object. ETag is the unique identifier of the object contents and is used to determine whether the contents of an object are changed. For example, if the ETag value is A when an object is uploaded and is B when the object is downloaded, this indicates the contents of the object are changed. The ETag reflects changes only to the contents of an object, not its metadata. Objects created by the upload and copy operations have unique ETags after being encrypted using MD5. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
size	long	Explanation: Object size, in bytes. Value range: The value ranges from 0 TB to 48.8 TB, in bytes. Default value: None
storageClass	StorageClassEnum	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 12-42 . Default value: None
objectType	ObjectTypeEnum	Explanation: Type of the object. Value range: Object type. For details, see Table 12-41 . Default value: None
isDeleteMarker	boolean	Explanation: Whether the object version is a delete marker. Value range: true false Default value: None

Code Example: Listing Object Versions in a Bucket

This example lists object versions in bucket **examplebucket**. A maximum of 1,000 object versions can be returned.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
public class ListVersions001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Listing object versions.
            ListVersionsResult result = obsClient.listVersions("examplebucket");
            System.out.println("listVersions successfully");
            for (VersionOrDeleteMarker v : result.getVersions()) {
                System.out.println("Key:" + v.getKey());
                System.out.println("Owner:" + v.getOwner());
                System.out.println("isDeleteMarker:" + v.isDeleteMarker());
            }
        } catch (ObsException e) {
            System.out.println("listVersions failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
            System.out.println("Host ID:" + e.getErrorHostId());
            e.printStackTrace();
        } catch (Exception e) {
            System.out.println("listVersions failed");
            // Print other error information.
            e.printStackTrace();
        }
    }
}
```

Code Example: Specifying the Version Count to List

This example lists a specified number of object versions in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
public class ListVersions002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
```

```
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
// advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
// ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // List a specified number of object versions.
    ListVersionsResult result = obsClient.listVersions("examplebucket", 100);
    System.out.println("listVersions successfully");
    for (VersionOrDeleteMarker v : result.getVersions()) {
        System.out.println("Key:" + v.getKey());
        System.out.println("Owner:" + v.getOwner());
        System.out.println("isDeleteMarker:" + v.isDeleteMarker());
    }
} catch (ObsException e) {
    System.out.println("listVersions failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listVersions failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Specifying the Version Prefix to List

This example specifies a prefix to list object versions in bucket **examplebucket**. A maximum of 100 object versions will be returned.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
public class ListVersions003 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
```



```
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // List object versions by specifying a prefix.
    // List 100 object versions with the specified prefix.
    ListVersionsRequest request = new ListVersionsRequest("examplebucket", 100);
    request.setPrefix("prefix");
    ListVersionsResult result = obsClient.listVersions(request);
    System.out.println("listVersions successfully");
    for (VersionOrDeleteMarker v : result.getVersions()) {
        System.out.println("Key:" + v.getKey());
        System.out.println("Owner:" + v.getOwner());
        System.out.println("isDeleteMarker:" + v.isDeleteMarker());
    }
} catch (ObsException e) {
    System.out.println("listVersions failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listVersions failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Specifying the Start Position to List

This example lists 100 object versions whose names follow **test** in lexicographical order in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
public class ListVersions004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");
```

```
// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // List objects from a specified start position.
    // List 100 object versions whose names following test in lexicographic order.
    ListVersionsRequest request = new ListVersionsRequest("examplebucket", 100);
    request.setKeyMarker("test");
    ListVersionsResult result = obsClient.listVersions(request);
    System.out.println("listVersions successfully");
    for (VersionOrDeleteMarker v : result.getVersions()) {
        System.out.println("Key:" + v.getKey());
        System.out.println("Owner:" + v.getOwner());
        System.out.println("isDeleteMarker:" + v.isDeleteMarker());
    }
} catch (ObsException e) {
    System.out.println("listVersions failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listVersions failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Using Pagination to List All Versions

This example lists 100 object versions whose names follow **test** in lexicographical order in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
public class ListVersions005 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
```

```
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // List all object versions using pagination.
    ListVersionsResult result;
    ListVersionsRequest request = new ListVersionsRequest("examplebucket", 100);
    do {
        result = obsClient.listVersions(request);
        System.out.println("listVersions successfully");
        for (VersionOrDeleteMarker v : result.getVersions()) {
            System.out.println("Key:" + v.getKey());
            System.out.println("Owner:" + v.getOwner());
            System.out.println("isDeleteMarker:" + v.isDeleteMarker());
        }
        request.setKeyMarker(result.getNextKeyMarker());
        request.setVersionIdMarker(result.getNextVersionIdMarker());
    } while (result.isTruncated());
} catch (ObsException e) {
    System.out.println("listVersions failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listVersions failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Listing All Object Versions in a Folder

There is no concept of folders in OBS. All elements stored in OBS buckets are objects. Folders are actually objects whose sizes are 0 and whose names end with a slash (/). You can set a folder name as a prefix to list objects in this folder. This example lists all object versions in a folder.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;

public class ListVersions006 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
```

```
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    //List all object versions in a folder.
    ListVersionsResult result;
    ListVersionsRequest request = new ListVersionsRequest("examplebucket", 100);
    // Set folder name dir/ as the prefix.
    request.setPrefix("dir/");
    do {
        result = obsClient.listVersions(request);
        System.out.println("listVersions successfully");
        for (VersionOrDeleteMarker v : result.getVersions()) {
            System.out.println("Key:" + v.getKey());
            System.out.println("Owner:" + v.getOwner());
            System.out.println("isDeleteMarker:" + v.isDeleteMarker());
        }
        request.setKeyMarker(result.getNextKeyMarker());
        request.setVersionIdMarker(result.getNextVersionIdMarker());
    } while (result.isTruncated());
} catch (ObsException e) {
    System.out.println("listVersions failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listVersions failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Listing All Object Versions in a Bucket by Folder Name

This example lists all object versions in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ListVersionsRequest;
import com.obs.services.model.ListVersionsResult;
import com.obs.services.model.VersionOrDeleteMarker;
public class ListVersions007 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
```

```
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // List all object versions in the root directory.
    ListVersionsRequest request = new ListVersionsRequest("examplebucket", 1000);
    request.setDelimiter("/");
    ListVersionsResult result = obsClient.listVersions(request);
    System.out.println("listVersions successfully");
    System.out.println("Objects in the root directory:");
    for (VersionOrDeleteMarker v : result.getVersions()) {
        System.out.println("Key:" + v.getKey());
        System.out.println("Owner:" + v.getOwner());
        System.out.println("isDeleteMarker:" + v.isDeleteMarker());
    }
    listVersionsByPrefix(obsClient, result);
} catch (ObsException e) {
    System.out.println("listVersions failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("listVersions failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Recursively Listing Object Versions in a Sub-folder

This example recursively lists object versions in a sub-folder.

```
// Recursively list object versions in a sub-folder.
static void listVersionsByPrefix(ObsClient obsClient, ListVersionsResult result) throws ObsException {
    for (String prefix : result.getCommonPrefixes()) {
        System.out.println("Objects in folder [" + prefix + "]:");
        ListVersionsRequest request = new ListVersionsRequest("examplebucket", 1000);
        request.setDelimiter("/");
        request.setPrefix(prefix);
        result = obsClient.listVersions(request);
        for (VersionOrDeleteMarker v : result.getVersions()) {
            System.out.println("Key:" + v.getKey());
            System.out.println("Owner:" + v.getOwner());
            System.out.println("isDeleteMarker:" + v.isDeleteMarker());
        }
        listVersionsByPrefix(obsClient, result);
    }
}
}
```

NOTE

- The code examples above do not involve scenarios where there are more than 1,000 object versions in a folder.
- Since all folder names end with a slash (/) and the objects and sub-folders to list are under the folder, **delimiter** is always a slash (/).
- In the returned result of each recursion, **ListVersionsResult.getVersions** includes the object versions under the folder, and **ListVersionsResult.getCommonPrefixes** includes the sub-folders under the folder.

12.8 Setting an ACL for an Object Version (SDK for Java)

Function

OBS allows the control of access permissions for objects. By default, only object creators have the read and write permissions on the object. You can set access control policies for objects. For example, if an object is configured with the public access policy, all users are allowed to read the object. If an object is encrypted with SSE-KMS, the ACL configured for it is not in effect in the cross-tenant case.

You can set an ACL when uploading an object or call an ACL API to modify or obtain the ACL of an existing object.

Restrictions

- To set an object ACL, you must be the bucket owner or have the required permission (**obs:object:PutObjectAcl** in IAM or **PutObjectAcl** in a bucket policy).
- An object ACL supports a maximum of 100 grants.

Method

obsClient.setObjectAcl([SetObjectAclRequest request](#))

Request Parameters

Table 12-44 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetObjectAclRequest	Yes	Explanation: Request parameters for setting an object ACL. For details, see Table 12-45 .

Table 12-45 SetObjectAclRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
acl	AccessControlList	No	Explanation: An ACL specified for the object. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 12-53 for the available options. - To use a user-defined ACL, see Table 12-46 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE

Table 12-46 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 12-47 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set<GrantAndPermission>	No	Explanation: Grantee information. For details, see Table 12-48 .

Table 12-47 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-48 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 12-49 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 12-52 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 12-49 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 12-50 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 12-51 . Default value: None

Table 12-50 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-51 GroupGrantee

Constant	Description
ALL_USERS	All users.

Constant	Description
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 12-52 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Table 12-53 Pre-defined ACL

Constant	Description
AccessControlList.REST_CANNED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControlList.REST_CANNED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControlList.REST_CANNED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControlList.REST_CANNED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControlList.REST_CAN_NED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Responses

Table 12-54 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example sets an ACL for the versioned object **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.GroupGrantee;
import com.obs.services.model.Owner;
import com.obs.services.model.Permission;
public class SetObjectAcl001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Define the ACL of the versioned object to Private.
            obsClient.setObjectAcl("examplebucket", "objectname", AccessControlList.REST_CANNED_PRIVATE,
"versionid");
            AccessControlList acl = new AccessControlList();
            Owner owner = new Owner();
            owner.setId("ownerid");
            acl.setOwner(owner);
            // Grant the read permission to all users.
            acl.grantPermission(GroupGrantee.ALL_USERS, Permission.PERMISSION_READ);
            // Set the ACL for the versioned object.
            obsClient.setObjectAcl("examplebucket", "objectname", acl, "versionid");
            System.out.println("setObjectAcl successfully");
        } catch (ObsException e) {
            System.out.println("setObjectAcl failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
        }
    }
}
```

```
// Request failed. Print the server-side error code.
System.out.println("Error Code:" + e.getErrorCode());
// Request failed. Print the error details.
System.out.println("Error Message:" + e.getErrorMessage());
// Request failed. Print the request ID.
System.out.println("Request ID:" + e.getErrorRequestId());
System.out.println("Host ID:" + e.getErrorHostId());
e.printStackTrace();
} catch (Exception e) {
    System.out.println("setObjectAcl failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

NOTE

The owner or grantee ID needed in the ACL indicates the account ID, which can be viewed on the **My Credential** page of OBS Console.

12.9 Obtaining the ACL of an Object Version (SDK for Java)

Function

OBS allows the control of access permissions for objects. By default, only object creators have the read and write permissions on the object. You can call an ACL API to modify or obtain the ACL of an existing object.

This API returns the ACL for a specific object version in a bucket.

Restrictions

- To obtain an object ACL, you must be the bucket owner or have the required permission (**obs:object:GetObjectAcl** in IAM or **GetObjectAcl** in a bucket policy).
- To call this API, you must have the read permission on the ACL of the object.

Method

obsClient.getObjectAcl([GetObjectAclRequest request](#))

Request Parameters

Table 12-55 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	GetObjectAclRequest	Yes	Explanation: Request parameters for obtaining the ACL of the object. For details, see Table 12-56 .

Table 12-56 GetObjectAclRequest parameters

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	Yes	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None

Responses

Table 12-57 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Description
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 12-58 .

Parameter	Type	Mandatory (Yes/No)	Description
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 12-59 .

Table 12-58 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-59 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 12-60 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 12-63 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 12-60 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 12-61 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 12-62 . Default value: None

Table 12-61 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 12-62 GroupGrantee

Constant	Description
ALL_USERS	All users.

Constant	Description
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 12-63 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.

Constant	Default Value	Description
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ, WRITE, READ_ACP, and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ, READ_ACP, and WRITE_ACP permissions for the object.

Code Examples

This example returns the ACL information of object version **objectname** in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
public class GetObjectAcl001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Obtain the ACL of the versioned object.
            AccessControlList acl = obsClient.getObjectAcl("examplebucket", "objectname", "versionid");
            System.out.println("getObjectAcl successfully");
            System.out.println(acl);
        } catch (ObsException e) {
            System.out.println("getObjectAcl failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
            System.out.println("Request ID:" + e.getErrorRequestId());
        }
    }
}
```

```
        System.out.println("Host ID:" + e.getErrorHostId());  
        e.printStackTrace();  
    } catch (Exception e) {  
        System.out.println("getObjectAcl failed");  
        // Print other error information.  
        e.printStackTrace();  
    }  
}
```

12.10 Deleting an Object Version (SDK for Java)

Function

This API deletes an object from a specific bucket.

You can call **ObsClient.deleteObject** to pass a version ID (**versionId**) to delete an object version.

Restrictions

- To delete an object, you must be the bucket owner or have the required permission (**obs:object:DeleteObject** in IAM or **DeleteObject** in a bucket policy).
- If versioning is not enabled for a bucket, deleted objects cannot be recovered.

Method

obsClient.deleteObject([DeleteObjectRequest request](#))

Request Parameters

Table 12-64 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	DeleteObjectRequest	Yes	Explanation: Request parameters for deleting an object. For details, see Table 12-65 .

Table 12-65 DeleteObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None

Responses

Table 12-66 DeleteObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
versionId	String	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
deleteMarker	boolean	Explanation: Whether the deleted object is a delete marker. Value range: • true: The deleted object is a delete marker. • false: The deleted object is not a delete marker. Default value: false
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Code Examples

This example passes a version ID (**versionId**) to delete an object version using **ObsClient.deleteObject**.

```
import com.obs.services.ObsClient;  
import com.obs.services.exception.ObsException;  
public class DeleteObject001 {
```

```
public static void main(String[] args) {
    // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
    // hard coding may result in leakage.
    // Obtain an AK/SK pair on the management console.
    String ak = System.getenv("ACCESS_KEY_ID");
    String sk = System.getenv("SECRET_ACCESS_KEY_ID");
    // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
    // advised not to use hard coding, which may result in information leakage.
    // Obtain an AK/SK pair and a security token using environment variables or import them in other
    // ways.
    // String securityToken = System.getenv("SECURITY_TOKEN");
    // Enter the endpoint corresponding to the region where the bucket is to be created.
    String endPoint = "https://your-endpoint";
    // Obtain an endpoint using environment variables or import it in other ways.
    //String endPoint = System.getenv("ENDPOINT");

    // Create an ObsClient instance.
    // Use the permanent AK/SK pair to initialize the client.
    ObsClient obsClient = new ObsClient(ak, sk, endPoint);
    // Use the temporary AK/SK pair and security token to initialize the client.
    // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

    try {
        // Delete the object version.
        obsClient.deleteObject("examplebucket", "objectname", "versionid");
        System.out.println("deleteObject successfully");
    } catch (ObsException e) {
        System.out.println("deleteObject failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("deleteObject failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

12.11 Batch Deleting Object Versions (SDK for Java)

Function

This API deletes objects in batches from a specific bucket. Deleted objects cannot be recovered.

In a batch delete operation, OBS concurrently deletes the specified objects and returns the deletion result of each object.

You can call **ObsClient.deleteObjects** to pass version IDs (**versionId**) to delete object versions.

Restrictions

- To delete objects in a batch, you must be the bucket owner or have the required permission (**obs:object:DeleteObject** in IAM or **DeleteObject** in a bucket policy).

- If versioning is not enabled for a bucket, deleted objects cannot be recovered.
- A maximum of 1,000 objects can be deleted at a time. If you send a request for deleting more than 1,000 objects, OBS returns an error message.
- After concurrent tasks are assigned, if an internal error occurs during cyclic deletion of multiple objects, an object may be deleted in the index data but still exist in the metadata.

Method

obsClient.deleteObjects([DeleteObjectsRequest deleteRequest](#))

Request Parameters

Table 12-67 List of request parameters

Parameter	Type	Mand atory (Yes/ No)	Description
deleteRequ est	DeleteObj ectsReque st	Yes	Explanation: Request parameters for deleting objects in batches. For details, see Table 12-68 .

Table 12-68 DeleteObjectsRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
keyAndVersions	List< Key AndVersion >	Yes	Explanation: List of objects to be deleted. For details, see Table 12-69 .

Parameter	Type	Mandatory (Yes/No)	Description
quiet	boolean	No	Explanation: Response mode to the request for deleting objects in a batch. Value range: • false: The detailed mode. Results of both successful and failed deletions are returned. • true: The quiet mode. Only results of failed deletions are returned. Default value: false

Table 12-69 KeyAndVersion

Parameter	Type	Mandatory (Yes/No)	Description
key	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None. If this parameter is left blank, the latest version of the object is deleted.

Responses

Table 12-70 DeleteObjectsResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
deletedObjectResults	List< DeleteObjectResult >	Explanation: Response results of the request for deleting objects in a batch. For details, see Table 12-71 .
errorResults	List< ErrorResult >	Explanation: List of objects that fail to be deleted. For details, see Table 12-72 .

Table 12-71 DeleteObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
versionId	String	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
deleteMarker	boolean	Explanation: Whether the deleted object is a delete marker. Value range: • true: The deleted object is a delete marker. • false: The deleted object is not a delete marker. Default value: false

Parameter	Type	Description
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Table 12-72 ErrorResult

Parameter	Type	Description
versionId	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
errorCode	String	Explanation: Error code for the failed deletion.

Parameter	Type	Description
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
message	String	Explanation: Error message for the failed deletion.

Code Examples

This example passes multiple version IDs (**versionId**) to batch delete object versions using **ObsClient.deleteObjects**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.DeleteObjectsRequest;
import com.obs.services.model.DeleteObjectsResult;
import com.obs.services.model.KeyAndVersion;
import java.util.ArrayList;
import java.util.List;
public class DeleteObjects001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Delete object versions in a batch.
            DeleteObjectsRequest request = new DeleteObjectsRequest("examplebucket");
```

```
request.setQuiet(false);
List<KeyAndVersion> toDelete = new ArrayList<KeyAndVersion>();
toDelete.add(new KeyAndVersion("objectname1", "versionid1"));
toDelete.add(new KeyAndVersion("objectname2", "versionid2"));
toDelete.add(new KeyAndVersion("objectname3", "versionid3"));
request.setKeyAndVersions(toDelete.toArray(new KeyAndVersion[toDelete.size()]));
DeleteObjectsResult result = obsClient.deleteObjects(request);
System.out.println("deleteObjects successfully");
System.out.println("getDeletedObjectResults:" + result.getDeletedObjectResults());
System.out.println("getErrorResults:" + result.getErrorResults());
} catch (ObsException e) {
    System.out.println("deleteObjects failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteObjects failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

13 Lifecycle Management (SDK for Java)

13.1 Overview (SDK for Java)

OBS allows you to set lifecycle rules for buckets to automatically transition the storage class of an object or delete expired objects, to effectively use storage features and optimize the storage space. You can set multiple lifecycle rules based on the prefix. A lifecycle rule must contain:

- Rule ID, which uniquely identifies the rule
- Prefix of objects that are under the control of this rule
- Transition policy of an object of the latest version, which can be specified in either mode:
 - a. How many days after the object is created
 - b. Transition date
- Expiration time of an object of the latest version, which can be specified in either mode:
 - a. How many days after the object is created
 - b. Expiration date
- Transition policy of a noncurrent object version, which can be specified in the following mode:
 - How many days after an object version becomes a noncurrent one
- Expiration time of a noncurrent object version, which can be specified in the following mode:
 - How many days after the object becomes a noncurrent object version
- Identifier specifying whether the setting is effective

13.2 Setting Lifecycle Rules (SDK for Java)

Function

This API configures lifecycle rules for a bucket to periodically delete objects in the bucket or transition objects between storage classes.

1. An object will be automatically deleted by the OBS server once it expires.
2. The time set in the transition policy of an object must be earlier than its expiration time, and the time set in the transition policy of a noncurrent object version must be earlier than its expiration time.
3. The configured expiration time and transition policy for a noncurrent object version can only take effect when versioning is enabled or suspended for the bucket where that object version is stored.

NOTE

- Objects in the Warm storage class have a minimum storage period of 30 days and those in the Cold storage class have a minimum storage period of 90 days. After an object is transitioned to the Cold storage class, if it stays in this storage class for less than 90 days, you still need to pay for a full 90 days.

Restrictions

- There is no limit on the number of lifecycle rules in a bucket, but the total size of XML descriptions about all lifecycle rules in a bucket cannot exceed 20 KB.
- A maximum of 20 lifecycle rules can be configured for a parallel file system.
- To configure a lifecycle rule for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutLifecycleConfiguration** in IAM or **PutLifecycleConfiguration** in a bucket policy).
- Expired objects will be permanently deleted and cannot be recovered.
- Multi-AZ redundancy is not available for storage. For this reason, buckets or objects with multi-AZ redundancy cannot be transitioned to the storage class based on a lifecycle rule.

Method

obsClient.setBucketLifecycle(final [SetBucketLifecycleRequest request](#))

Request Parameters

Table 13-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetBucketLifecycleRequest	Yes	Explanation: Request parameters for setting the lifecycle rule. For details, see Table 13-2 .

Table 13-2 SetBucketLifecycleRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: - A bucket name must be unique across all accounts and regions. - A bucket name: - Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. - Cannot be formatted as an IP address. - Cannot start or end with a hyphen (-) or period (.). - Cannot contain two consecutive periods (..), for example, my..bucket. - Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. - If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
lifecycleConfiguration	LifecycleConfiguration	Yes	Explanation: Lifecycle rules for the bucket. For details, see Table 13-3 .

Table 13-3 LifecycleConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
rules	List< Rule >	No	Explanation: List of lifecycle rules. For details, see Table 13-4 .

Table 13-4 Rule

Parameter	Type	Mandatory (Yes/No)	Description
id	String	No	Explanation: Lifecycle rule ID. Value range: The value must contain 1 to 255 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
prefix	String	Yes	Explanation: Object name prefix. It identifies the objects the rule applies to. You can leave this parameter blank to apply the rule to all objects in the bucket. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you set prefix to ExampleObject.jpg, the rule applies to object ExampleObject.jpg only. If you set prefix to logs/, the rule applies to the three objects with name starting with logs/. If you leave prefix blank, the rule applies to all objects in the bucket. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
enabled	boolean	Yes	Explanation: Whether the rule is enabled. Value range: true: The rule is enabled. false: The rule is disabled. Default value: None
expiration	Expiration	No	Explanation: Expiration time of an object. For details, see Table 13-5 . Default value: None
noncurrentVersionExpiration	NoncurrentVersionExpiration	No	Explanation: Expiration time of noncurrent object versions. For details, see Table 13-6 . Restrictions: - This parameter is only available for noncurrent object versions. - Versioning must be enabled (or suspended after being enabled) for the bucket. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
transitions	List< Transition >	No	Explanation: Policies for storage class transition, including transition time and the storage class after transition. For details, see Table 13-7. Restrictions: This parameter is only available for the current object version. Default value: None
noncurrentVersionTransitions	List< NoncurrentVersionTransition >	No	Explanation: Policies for storage class transition of noncurrent versions, including transition time and the storage class after transition. For details, see Table 13-9. Restrictions: • This parameter is only available for noncurrent object versions. • Versioning must be enabled (or suspended after being enabled) for the bucket. Default value: None

Table 13-5 Expiration

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	Yes if date is absent during object storage class transition Yes if date or expiredObjectDeleteMarker is absent during object deletion	Explanation: Number of days (since the last update was made to the object) after which the lifecycle rule takes effect (the object will be deleted). Restrictions: This parameter is only available for the current object version. Value range: A positive integer, in days. Default value: None
date	Date	Yes if days is absent during object storage class transition. Yes if days or expiredObjectDeleteMarker is absent during object deletion	Explanation: The value must conform to the ISO8601 standards and indicate UTC 00:00. For example, 2018-01-01T00:00:00.000Z indicates only objects that were last modified before the specified time are transitioned to the specified storage class or deleted. Default value: None
expiredObjectDeleteMarker	Boolean	Yes if days or date is absent	Explanation: Whether to delete expired delete markers. The value can be true or false. If tags are configured in a lifecycle rule, this element cannot be configured. Default value: None

Table 13-6 NoncurrentVersionExpiration

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	Yes	Explanation: Number of days an object is noncurrent before it expires. Restrictions: This parameter is only available for noncurrent object versions. Value range: A positive integer, in days. Default value: None

Table 13-7 Transition

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	This parameter is mandatory if date is not configured.	Explanation: Number of days after its creation when the object is transitioned. Restrictions: This parameter is only available for the current object version. Value range: A positive integer, in days. Default value: None
date	Date	This parameter is mandatory if days is not configured.	Explanation: Date when the object will be transitioned. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
storageClass	StorageClassEnum	Yes	Explanation: Storage class the object is transitioned to. Restrictions: Restrictions on storage class transitions: • Only transitions from the Standard storage class to the storage class are supported. To transition objects from to Standard, you must manually do it. • Only transitions from the Standard or storage class to the storage class are supported. To transition objects from to Standard or , you must first restore these objects and then manually transition their storage classes. • Multi-AZ redundancy is not available for storage. For this reason, buckets or objects with multi-AZ redundancy cannot be transitioned to the storage class based on a lifecycle rule. Value range: See Table 13-8. Default value: None

Table 13-8 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 13-9 NoncurrentVersionTransition

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	This parameter is mandatory if date is not configured.	Explanation: Number of days after its creation when the object is transitioned. Restrictions: This parameter is only available for noncurrent object versions. Value range: A positive integer, in days. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
storageClass	StorageClassEnum	Yes	Explanation: Storage class the object is transitioned to. Restrictions: Restrictions on storage class transitions: • Only transitions from the Standard storage class to the storage class are supported. To transition objects from to Standard, you must manually do it. • Only transitions from the Standard or storage class to the storage class are supported. To transition objects from to Standard or , you must first restore these objects and then manually transition their storage classes. • Multi-AZ redundancy is not available for storage. For this reason, buckets or objects with multi-AZ redundancy cannot be transitioned to the storage class based on a lifecycle rule. Value range: See Table 13-8 . Default value: None

 NOTE

Transitions, Expiration, NoncurrentVersionTransitions, AbortIncompleteMultipartUpload, and NoncurrentVersionExpiration must not be all left blank.

Responses

Table 13-10 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Example: Setting an Object Transition Policy

This example configures a transition policy for latest and historical object versions in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.LifecycleConfiguration;
import com.obs.services.model.StorageClassEnum;
public class SetBucketLifecycle001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
```

```
try {
    //Set an object transition policy.
    LifecycleConfiguration config = new LifecycleConfiguration();
    LifecycleConfiguration.Rule rule = config.new Rule();
    rule.setEnabled(true);
    rule.setId("rule1");
    rule.setPrefix("prefix");
    LifecycleConfiguration.Transition transition = config.new Transition();
    // Specify that objects whose names contain the specified prefix will be transitioned 30 days after
creation.
    transition.setDays(30);
    // Specify the storage class that the object will be transitioned to.
    transition.setObjectStorageClass(StorageClassEnum.WARM);
    // Specify when the objects whose names contain the specified prefix will be transitioned.
    // transition.setDate(new SimpleDateFormat("yyyy-MM-dd").parse("2018-10-31"));
    rule.getTransitions().add(transition);
    LifecycleConfiguration.NoncurrentVersionTransition noncurrentVersionTransition =
        config.new NoncurrentVersionTransition();
    // Specify that objects whose names contain the specified prefix will be transitioned 30 days after
being historical versions.
    noncurrentVersionTransition.setDays(30);
    // Specify the storage class of the historical object version after transition.
    noncurrentVersionTransition.setObjectStorageClass(StorageClassEnum.COLD);
    rule.getNoncurrentVersionTransitions().add(noncurrentVersionTransition);
    // Set the expiration time of fragments.
    LifecycleConfiguration.AbortIncompleteMultipartUpload abortIncompleteMultipartUpload =
config.new AbortIncompleteMultipartUpload();
    abortIncompleteMultipartUpload.setDaysAfterInitiation(7);
    rule.setAbortIncompleteMultipartUpload(abortIncompleteMultipartUpload);
    config.addRule(rule);
    obsClient.setBucketLifecycle("examplebucket", config);
    System.out.println("setBucketLifecycle successfully");
} catch (ObsException e) {
    System.out.println("setBucketLifecycle failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketLifecycle failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Setting the Object Expiration Time

This example configures the expiration time of latest and historical object versions in bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.LifecycleConfiguration;
public class SetBucketLifecycle002 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
```

```
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Set the object expiration time.
    LifecycleConfiguration config = new LifecycleConfiguration();
    LifecycleConfiguration.Rule rule = config.new Rule();
    rule.setEnabled(true);
    rule.setId("rule1");
    rule.setPrefix("prefix");
    LifecycleConfiguration.Expiration expiration = config.new Expiration();
    // Specify that objects whose names contain the specified prefix will expire 60 days after creation.
    expiration.setDays(60);
    // Specify when the objects whose names contain the specified prefix will expire.
    // expiration.setDate(new SimpleDateFormat("yyyy-MM-dd").parse("2018-12-31"));
    rule.setExpiration(expiration);
    LifecycleConfiguration.NoncurrentVersionExpiration noncurrentVersionExpiration =
        config.new NoncurrentVersionExpiration();
    // Specify that objects whose names contain the specified prefix will expire after changing into
historical versions for 60 days.
    noncurrentVersionExpiration.setDays(60);
    rule.setNoncurrentVersionExpiration(noncurrentVersionExpiration);
    config.addRule(rule);
    obsClient.setBucketLifecycle("examplebucket", config);
    System.out.println("setBucketLifecycle successfully");
} catch (ObsException e) {
    System.out.println("setBucketLifecycle failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketLifecycle failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

13.3 Obtaining Lifecycle Rules (SDK for Java)

Function

You can configure lifecycle rules to periodically delete objects or transition objects between storage classes.

This API returns the lifecycle rules of a bucket.

Restrictions

- To obtain the lifecycle configuration of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetLifecycleConfiguration** in IAM or **GetLifecycleConfiguration** in a bucket policy).

Method

obsClient.getBucketLifecycle(final [BaseBucketRequest request](#))

Request Parameters

Table 13-11 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters related to basic bucket information. For details, see Table 13-12 .

Table 13-12 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Table 13-13 GetBucketLifecycleRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 13-14 LifecycleConfiguration

Parameter	Type	Description
rules	List< Rule >	Explanation: List of lifecycle rules. For details, see Table 13-15 .

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Table 13-15 Rule

Parameter	Type	Mandatory (Yes/No)	Description
id	String	No	Explanation: Lifecycle rule ID. Value range: The value must contain 1 to 255 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
prefix	String	Yes	Explanation: Object name prefix. It identifies the objects the rule applies to. You can leave this parameter blank to apply the rule to all objects in the bucket. Assume that you have the following objects: logs/day1, logs/day2, logs/day3, and ExampleObject.jpg. If you set prefix to ExampleObject.jpg, the rule applies to object ExampleObject.jpg only. If you set prefix to logs/, the rule applies to the three objects with name starting with logs/. If you leave prefix blank, the rule applies to all objects in the bucket. Value range: The value must contain 1 to 1,024 characters. Default value: None
enabled	boolean	Yes	Explanation: Whether the rule is enabled. Value range: true: The rule is enabled. false: The rule is disabled. Default value: None
expiration	Expiration	No	Explanation: Expiration time of an object. For details, see Table 13-16. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
noncurrentVersionExpiration	NoncurrentVersionExpiration	No	Explanation: Expiration time of noncurrent object versions. For details, see Table 13-17. Restrictions: - This parameter is only available for noncurrent object versions. - Versioning must be enabled (or suspended after being enabled) for the bucket. Default value: None
transitions	List< Transition >	No	Explanation: Policies for storage class transition, including transition time and the storage class after transition. For details, see Table 13-18. Restrictions: This parameter is only available for the current object version. Default value: None
noncurrentVersionTransitions	List< NoncurrentVersionTransition >	No	Explanation: Policies for storage class transition of noncurrent versions, including transition time and the storage class after transition. For details, see Table 13-20. Restrictions: - This parameter is only available for noncurrent object versions. - Versioning must be enabled (or suspended after being enabled) for the bucket. Default value: None

Table 13-16 Expiration

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	Yes if date is absent during object storage class transition Yes if date or expiredObjectDeleteMarker is absent during object deletion	Explanation: Number of days (since the last update was made to the object) after which the lifecycle rule takes effect (the object will be deleted). Restrictions: This parameter is only available for the current object version. Value range: A positive integer, in days. Default value: None
date	Date	Yes if days is absent during object storage class transition. Yes if days or expiredObjectDeleteMarker is absent during object deletion	Explanation: The value must conform to the ISO8601 standards and indicate UTC 00:00. For example, 2018-01-01T00:00:00.000Z indicates only objects that were last modified before the specified time are transitioned to the specified storage class or deleted. Default value: None
expiredObjectDeleteMarker	Boolean	Yes if days or date is absent	Explanation: Whether to delete expired delete markers. The value can be true or false. If tags are configured in a lifecycle rule, this element cannot be configured. Default value: None

Table 13-17 NoncurrentVersionExpiration

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	Yes	Explanation: Number of days an object is noncurrent before it expires. Restrictions: This parameter is only available for noncurrent object versions. Value range: A positive integer, in days. Default value: None

Table 13-18 Transition

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	This parameter is mandatory if date is not configured.	Explanation: Number of days after its creation when the object is transitioned. Restrictions: This parameter is only available for the current object version. Value range: A positive integer, in days. Default value: None
date	Date	This parameter is mandatory if days is not configured.	Explanation: Date when the object will be transitioned. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
storageClass	StorageClassEnum	Yes	Explanation: Storage class the object is transitioned to. Restrictions: Restrictions on storage class transitions: - Only transitions from the Standard storage class to the storage class are supported. To transition objects from to Standard, you must manually do it. - Only transitions from the Standard or storage class to the storage class are supported. To transition objects from to Standard or , you must first restore these objects and then manually transition their storage classes. - Multi-AZ redundancy is not available for storage. For this reason, buckets or objects with multi-AZ redundancy cannot be transitioned to the storage class based on a lifecycle rule. Value range: See Table 13-19. Default value: None

Table 13-19 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.

Constant	Default Value	Description
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 13-20 NoncurrentVersionTransition

Parameter	Type	Mandatory (Yes/No)	Description
days	Integer	This parameter is mandatory if date is not configured.	Explanation: Number of days after its creation when the object is transitioned. Restrictions: This parameter is only available for noncurrent object versions. Value range: A positive integer, in days. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
storageClass	Storage Class Enum	Yes	Explanation: Storage class the object is transitioned to. Restrictions: Restrictions on storage class transitions: • Only transitions from the Standard storage class to the storage class are supported. To transition objects from to Standard, you must manually do it. • Only transitions from the Standard or storage class to the storage class are supported. To transition objects from to Standard or , you must first restore these objects and then manually transition their storage classes. • Multi-AZ redundancy is not available for storage. For this reason, buckets or objects with multi-AZ redundancy cannot be transitioned to the storage class based on a lifecycle rule. Value range: See Table 13-19 . Default value: None

Code Examples

This example returns the lifecycle configuration of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.LifecycleConfiguration;
public class GetBucketLifecycle001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        //String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
```

```
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // View the lifecycle rules
    LifecycleConfiguration config = obsClient.getBucketLifecycle("examplebucket");
    System.out.println("getBucketLifecycle successfully");
    for (LifecycleConfiguration.Rule rule : config.getRules()) {
        System.out.println(rule.getId());
        System.out.println(rule.getPrefix());
        for (LifecycleConfiguration.Transition transition : rule.getTransitions()) {
            System.out.println(transition.getDays());
            System.out.println(transition.getStorageClass());
        }
        System.out.println(rule.getExpiration() != null ? rule.getExpiration().getDays() : "");
        for (LifecycleConfiguration.NoncurrentVersionTransition noncurrentVersionTransition :
            rule.getNoncurrentVersionTransitions()) {
            System.out.println(noncurrentVersionTransition.getDays());
            System.out.println(noncurrentVersionTransition.getStorageClass());
        }
        System.out.println(
            rule.getNoncurrentVersionExpiration() != null
                ? rule.getNoncurrentVersionExpiration().getDays()
                : "");
    }
} catch (ObsException e) {
    System.out.println("getBucketLifecycle failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getBucketLifecycle failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

13.4 Deleting Lifecycle Rules (SDK for Java)

Function

You can configure lifecycle rules to periodically delete objects or transition objects between storage classes.

This API deletes the lifecycle configuration of a bucket.

Restrictions

- To delete the lifecycle configuration of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutLifecycleConfiguration** in IAM or **PutLifecycleConfiguration** in a bucket policy).
- You cannot use this API to delete a specified lifecycle rule. When you call this API, all lifecycle rules in the bucket are deleted.

Method

obsClient.deleteBucketLifecycle(final [BaseBucketRequest request](#))

Request Parameters

Table 13-21 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters related to basic bucket information. For details, see Table 13-22 .
request	DeleteBucketLifecycleRequest	No Either DeleteBucketLifecycleRequest or BaseBucketRequest must be used.	Explanation: Request parameters for deleting the lifecycle configuration of a bucket. For details, see Table 13-23 . Restrictions: It is supported in 3.25.10 and later versions.

Table 13-22 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Table 13-23 DeleteBucketLifecycleRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 13-24 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes the lifecycle configurations of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class DeleteBucketLifecycle001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Delete the lifecycle configurations.
```

```
        obsClient.deleteBucketLifecycle("examplebucket");
        System.out.println("deleteBucketLifecycle successfully");
    } catch (ObsException e) {
        System.out.println("deleteBucketLifecycle failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("deleteBucketLifecycle failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

14 Cross-Origin Resource Sharing (SDK for Java)

14.1 Overview (SDK for Java)

Cross-origin access refers to access between different domains. Restricting cross-origin access is a browser policy for security purposes, that is, the same-origin policy.

Due to this same-origin policy, JavaScript in origin A cannot operate objects in origin B or C.

The same-origin policy requires the protocols, domain names (or IP addresses), and ports are all the same. If the protocols, domain names, and ports (if specified) of the two web pages are the same, the two web pages are in the same origin.

Cross-origin resource sharing (CORS) allows web application programs in one origin to access resources in another.

OBS supports CORS rules that allow the resources in OBS to be requested by other domains.

14.2 Configuring a CORS Rule (SDK for Java)

Function

Cross-origin resource sharing (CORS) is a mechanism defined by the World Wide Web Consortium (W3C) that allows a web application program in one domain to access resources located in another one. For general web page requests, website scripts and contents in one domain cannot interact with those in another because of Same Origin Policies (SOPs). OBS supports CORS rules that allow the resources in OBS to be requested by other domains.

You can call **ObsClient.setBucketCors** to set CORS rules for a bucket. The configured CORS rules follow the principle of new ones overwriting old ones.

Restrictions

- To configure CORS for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketCORS** in IAM or **PutBucketCORS** in a bucket policy).

Method

obsclient.setBucketCors(final [SetBucketCorsRequest request](#))

Request Parameters

Table 14-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetBucketCorsRequest	Yes	Request parameters for setting a CORS rule. For details, see Table 14-2 .

Table 14-2 SetBucketCorsRequest

Parameter	Type	Man dato ry (Yes /No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
bucketCors	BucketCors	Yes	Explanation: List of CORS rules of a bucket. For details, see Table 14-3 .

Table 14-3 BucketCors

Parameter	Type	Mandatory (Yes/No)	Description
rules	List< BucketCorsRule >	Yes	Explanation: List of CORS rules of a bucket. For details, see Table 14-4 .

Table 14-4 BucketCorsRule

Parameter	Type	Mandatory (Yes/No)	Description
id	String	No	Explanation: CORS rule ID. Value range: The value must contain 1 to 255 characters. Default value: None
allowed Method	List<String>	Yes	Explanation: The allowed HTTP methods for a cross-origin request, indicating the operation types for buckets and objects. Value range: The following HTTP methods are supported: • GET • PUT • HEAD • POST • DELETE Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
allowedOrigin	List<String>	Yes	Explanation: The origin from which the requests can access the bucket. Restrictions: Domain name of the origin. Each origin can contain only one wildcard character (*), for example, https://*.vbs.example.com. Default value: None
allowedHeader	List<String>	No	Explanation: The allowed cross-origin request headers. Only CORS requests matching the allowed headers are valid. Restrictions: Each header can contain only one wildcard character (*). Spaces, ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None
maxAgeSeconds	int	No	Explanation: Duration your client can cache the response for a cross-origin request. Restrictions: Each bucket CORS rule can contain only one maxAgeSeconds. Value range: An integer greater than or equal to 0, in seconds. Default value: 100

Parameter	Type	Mandatory (Yes/No)	Description
exposeHeader	List<String>	No	Explanation: The CORS-allowed additional headers in the response. These headers provide additional information to clients. By default, your browser can only access headers Content-Length and Content-Type . If your browser needs to access other headers, add them to the list of the allowed additional headers. Restrictions: Spaces, wildcard characters (*), ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None

Responses

Table 14-5 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures CORS rules for bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketCors;
import com.obs.services.model.BucketCorsRule;
import com.obs.services.model.DeleteObjectsRequest;
import com.obs.services.model.DeleteObjectsResult;
import com.obs.services.model.KeyAndVersion;
import java.util.ArrayList;
import java.util.List;
public class SetBucketCors001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            //Configure CORS rules.
            BucketCors cors = new BucketCors();
            List<BucketCorsRule> rules = new ArrayList<BucketCorsRule>();
            BucketCorsRule rule = new BucketCorsRule();
            ArrayList<String> allowedOrigin = new ArrayList<String>();
            // Specify the origin of the cross-origin request.
            allowedOrigin.add( "http://www.a.com");
            allowedOrigin.add( "http://www.b.com");
            rule.setAllowedOrigin(allowedOrigin);
            ArrayList<String> allowedMethod = new ArrayList<String>();
            // Specify the request method, which can be GET, PUT, DELETE, POST, or HEAD.
            allowedMethod.add("GET");
            allowedMethod.add("HEAD");
            allowedMethod.add("PUT");
            rule.setAllowedMethod(allowedMethod);
            ArrayList<String> allowedHeader = new ArrayList<String>();
            // Specify whether headers specified in Access-Control-Request-Headers in the OPTIONS request
            // can be used.
            allowedHeader.add("x-obs-header");
            rule.setAllowedHeader(allowedHeader);
            ArrayList<String> exposeHeader = new ArrayList<String>();
            // Specify response headers that users can access using application programs.
            exposeHeader.add("x-obs-expose-header");
            rule.setExposeHeader(exposeHeader);
            // Specify the browser's cache time of the returned results of OPTIONS requests for specific
            // resources, in seconds.
            rule.setMaxAgeSecond(10);
            rules.add(rule);
            cors.setRules(rules);
            obsClient.setBucketCors("examplebucket", cors);
            System.out.println("setBucketCors successfully");
        } catch (ObsException e) {
```

```
System.out.println("setBucketCors failed");  
// Request failed. Print the HTTP status code.  
System.out.println("HTTP Code:" + e.getResponseCode());  
// Request failed. Print the server-side error code.  
System.out.println("Error Code:" + e.getErrorCode());  
// Request failed. Print the error details.  
System.out.println("Error Message:" + e.getErrorMessage());  
// Request failed. Print the request ID.  
System.out.println("Request ID:" + e.getErrorRequestId());  
System.out.println("Host ID:" + e.getErrorHostId());  
e.printStackTrace();  
} catch (Exception e) {  
    System.out.println("setBucketCors failed");  
    // Print other error information.  
    e.printStackTrace();  
}  
}
```

14.3 Obtaining a CORS Rule (SDK for Java)

Function

CORS is a browser-standard mechanism defined by the W3C. It allows a web client in one origin to interact with resources in another. For general web page requests, website scripts and contents in one origin cannot interact with those in another due to SOPs. OBS supports CORS, allowing the resources in OBS to be requested across origins.

This API returns the CORS configuration of a bucket.

Restrictions

- To obtain the CORS configuration of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketCORS** in IAM or **GetBucketCORS** in a bucket policy).

Method

obsclient.getBucketCors(final [BaseBucketRequest request](#))

Request Parameters

Table 14-6 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters for obtaining a CORS rule. For details, see Table 14-7 .

Table 14-7 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 14-8 BucketCors

Parameter	Type	Mandatory (Yes/No)	Description
rules	List< BucketCorsRule >	Yes	Explanation: List of CORS rules of a bucket. For details, see Table 14-9 .

Table 14-9 BucketCorsRule

Parameter	Type	Mandatory (Yes/No)	Description
id	String	No	Explanation: CORS rule ID. Value range: The value must contain 1 to 255 characters. Default value: None
allowed Method	List<String>	Yes	Explanation: The allowed HTTP methods for a cross-origin request, indicating the operation types for buckets and objects. Value range: The following HTTP methods are supported: • GET • PUT • HEAD • POST • DELETE Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
allowedOrigin	List<String>	Yes	Explanation: The origin from which the requests can access the bucket. Restrictions: Domain name of the origin. Each origin can contain only one wildcard character (*), for example, https://*.vbs.example.com. Default value: None
allowedHeader	List<String>	No	Explanation: The allowed cross-origin request headers. Only CORS requests matching the allowed headers are valid. Restrictions: Each header can contain only one wildcard character (*). Spaces, ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None
maxAgeSeconds	int	No	Explanation: Duration your client can cache the response for a cross-origin request. Restrictions: Each bucket CORS rule can contain only one maxAgeSeconds. Value range: An integer greater than or equal to 0, in seconds. Default value: 100

Parameter	Type	Mandatory (Yes/No)	Description
exposeHeader	List<String>	No	Explanation: The CORS-allowed additional headers in the response. These headers provide additional information to clients. By default, your browser can only access headers Content-Length and Content-Type. If your browser needs to access other headers, add them to the list of the allowed additional headers. Restrictions: Spaces, wildcard characters (*), ampersands (&), colons (:), and less-than signs (<) are not allowed. Default value: None

Code Examples

This example returns the CORS configuration of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketCors;
import com.obs.services.model.BucketCorsRule;
public class GetBucketCors001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Obtain the CORS rules.
            BucketCors cors = obsClient.getBucketCors("examplebucket");
            for(BucketCorsRule rule : cors.getRules()){
                System.out.println("Id:" + rule.getId());
                System.out.println("MaxAgeSecond:" + rule.getMaxAgeSecond());
                System.out.println("AllowedHeader:" + rule.getAllowedHeader());
                System.out.println("AllowedOrigin:" + rule.getAllowedOrigin());
                System.out.println("AllowedMethod:" + rule.getAllowedMethod());
            }
        }
    }
}
```



```
        System.out.println("ExposeHeader:" + rule.getExposeHeader());
    }
    System.out.println("getBucketCors successfully");
} catch (ObsException e) {
    System.out.println("getBucketCors failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getBucketCors failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

14.4 Deleting a CORS Rule (SDK for Java)

Function

CORS is a browser-standard mechanism defined by the W3C. It allows a web client in one origin to interact with resources in another. For general web page requests, website scripts and contents in one origin cannot interact with those in another due to SOPs. OBS supports CORS, allowing the resources in OBS to be requested across origins.

This API deletes the CORS rules of a bucket.

Restrictions

- To delete the CORS configuration of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketCORS** in IAM or **PutBucketCORS** in a bucket policy).

Method

obsclient.deleteBucketCors(final [BaseBucketRequest request](#))

Request Parameters

Table 14-10 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters for deleting a CORS rule. For details, see Table 14-11 .

Table 14-11 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 14-12 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes CORS rules of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class DeleteBucketCors001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Delete CORS rules.
```

```
        obsClient.deleteBucketCors("examplebucket");
        System.out.println("deleteBucketCors successfully");
    } catch (ObsException e) {
        System.out.println("deleteBucketCors failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("deleteBucketCors failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

15 Logging (SDK for Java)

15.1 Overview (SDK for Java)

OBS allows you to configure access logging for buckets. After the configuration, access to buckets will be recorded in logs. These logs will be saved in specific buckets in OBS.

You can enable OBS logging for bucket analysis or audit purposes. With access logs, a bucket owner can analyze the characteristics, types, or trends of requests sent to the bucket.

With logging enabled, OBS automatically logs access requests for the bucket and writes the generated log files into a specified bucket.

You need to specify a bucket for storing log files when enabling logging for a bucket. Log files can be stored in any bucket you own in the region where the logged bucket is, including the logged bucket itself.

For more information, see "Logging" in the *User Guide (Ally Region)*.

15.2 Configuring Logging for a Bucket (SDK for Java)

Function

This API enables logging for a bucket (source) and configures another bucket (target) to store the log files. When a bucket is created, logging is not enabled by default. You can call this API to enable logging for the bucket. With logging enabled, a log message is generated for each operation on the bucket. Multiple log messages are packed into a file. The target bucket for storing log files must be specified when logging is enabled. It can be the bucket logging is enabled for, or any other bucket you have access to. If you specify another bucket for storing logs, the bucket must be in the same region as the logged bucket. You can also specify access permissions and name prefixes for log files.

Restrictions

- The source and target buckets must be in the same region.
- A bucket in the Warm or Cold storage class cannot be used as a target bucket.
- OBS creates log files and uploads them to the bucket. Before enabling logging for a bucket, you need to create an IAM agency to delegate OBS to upload log files to the specified bucket.
- To configure logging for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketLogging** in IAM or **PutBucketLogging** in a bucket policy).

Method

obsClient.setBucketLogging(final [SetBucketLoggingRequest request](#))

Request Parameters

Table 15-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetBucketLoggingRequest	Yes	Explanation: Request parameters for configuring logging for a bucket. For details, see Table 15-2 .

Table 15-2 SetBucketLoggingRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Name of the source bucket. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
loggingConfiguration	BucketLoggingConfiguration	Yes	Explanation: Bucket logging configurations. For details, see Table 15-3 .

Table 15-3 BucketLoggingConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
agency	String	Yes if you configure bucket logging	Explanation: Name of the IAM agency created by the owner of the target bucket for OBS. You can select an existing IAM agency or create one. Restrictions: By default, the IAM agency only requires the PutObject permission to upload logs to the target bucket. If default encryption is enabled for the target bucket, the agency also requires the KMS Administrator permission in the region where the target bucket is located. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
targetBucketName	String	No	Explanation: Name of the bucket for storing log files. Restrictions: • This bucket must be in the same region as the bucket with logging enabled. • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in

Parameter	Type	Mandatory (Yes/No)	Description
			the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
logfilePrefix	String	No	Explanation: Name prefix for log files stored in the target bucket. Value range: The value must contain 1 to 1,024 characters. Default value: None
targetGrantsList	List< GrantAndPermission >	No	Explanation: Permission information list of grantees, which defines grantees and their permissions for log files. For details, see Table 15-4 .

Table 15-4 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 15-5 .

Parameter	Type	Mandatory (Yes/No)	Description
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 15-8 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 15-5 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 15-6 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 15-7 . Default value: None

Table 15-6 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 15-7 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 15-8 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.

Constant	Default Value	Description
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Responses

Table 15-9 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Example: Enabling Bucket Logging

This example configures logging for bucket **examplebucket**, with **your agency** as the agency, **targetprefix** as the prefix for generated log files, and **targetbucketname** as the bucket for storing log files.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketLoggingConfiguration;
public class SetBucketLogging001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
```

```
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Enable bucket logging.
    BucketLoggingConfiguration config = new BucketLoggingConfiguration();
    // Set an agency. You need to create one on IAM.
    config.setAgency("your agency");
    config.setTargetBucketName("targetbucketname");
    config.setLogfilePrefix("targetprefix");
    obsClient.setBucketLogging("examplebucket", config);
    System.out.println("setBucketLogging successfully");
} catch (ObsException e) {
    System.out.println("setBucketLogging failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketLogging failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Granting the Read Permission on a Log Object

This example configures logging for bucket **examplebucket**, with **your agency** as the agency, **targetprefix** as the prefix for generated log files, and **targetbucketname** as the bucket for storing log files, and then grants all users the read permission for the logs.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketLoggingConfiguration;
import com.obs.services.model.GrantAndPermission;
import com.obs.services.model.GroupGrantee;
import com.obs.services.model.Permission;
public class SetBucketLogging002
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
    }
}
```

```
try {
    // Configure access to logs.
    String targetBucket = "targetbucketname";
    // Configure logging for the bucket.
    BucketLoggingConfiguration config = new BucketLoggingConfiguration();
    // Set an agency. You need to create one on IAM.
    config.setAgency("your agency");
    config.setTargetBucketName(targetBucket);
    config.setLogFilePrefix("prefix");
    // Grant all users the READ permission for the logs.
    GrantAndPermission grant1 = new GrantAndPermission(GroupGrantee.ALL_USERS,
Permission.PERMISSION_READ);
    config.setTargetGrants(new GrantAndPermission[]{grant1});
    obsClient.setBucketLogging("examplebucket", config);
    System.out.println("setBucketLogging successfully");
} catch (ObsException e) {
    System.out.println("setBucketLogging failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketLogging failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Disabling Bucket Logging

This example disables the logging for bucket **examplebucket** by clearing the logging configurations of the bucket using **ObsClient.setBucketLogging**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketLoggingConfiguration;
public class SetBucketLogging003
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Leave the logging configurations in blank.

```



```
obsClient.setBucketLogging("examplebucket", new BucketLoggingConfiguration());
System.out.println("setBucketLogging successfully");
} catch (ObsException e) {
    System.out.println("setBucketLogging failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketLogging failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

15.3 Obtaining the Logging Configuration of a Bucket (SDK for Java)

Function

This API returns the logging configuration of a bucket.

Restrictions

- To obtain the logging configuration of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketLogging** in IAM or **GetBucketLogging** in a bucket policy).

Method

obsClient.getBucketLogging(final [BaseBucketRequest request](#))

Request Parameters

Table 15-10 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters for obtaining the logging configurations of a bucket. For details, see Table 15-11 .

Table 15-11 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 15-12 BucketLoggingConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
agency	String	Yes if you configure bucket logging	Explanation: Name of the IAM agency created by the owner of the target bucket for OBS. You can select an existing IAM agency or create one. Restrictions: By default, the IAM agency only requires the PutObject permission to upload logs to the target bucket. If default encryption is enabled for the target bucket, the agency also requires the KMS Administrator permission in the region where the target bucket is located. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
targetBucket Name	String	No	Explanation: Name of the bucket for storing log files. Restrictions: • This bucket must be in the same region as the bucket with logging enabled. • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
logfilePrefix	String	No	Explanation: Name prefix for log files stored in the target bucket. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
targetGrants List	List< GrantAndPermission >	No	Explanation: Permission information list of grantees, which defines grantees and their permissions for log files. For details, see Table 15-13 .

Table 15-13 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 15-14 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 15-17 . Default value: None
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 15-14 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 15-15 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 15-16 . Default value: None

Table 15-15 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 15-16 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 15-17 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.

Constant	Default Value	Description
PERMISSION_WRITE_ACL	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Code Examples

This example returns the logging configuration of bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.BucketLoggingConfiguration;
public class GetBucketLogging001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
```



```
try {
    // View the bucket logging configuration.
    BucketLoggingConfiguration config = obsClient.getBucketLogging("examplebucket");
    System.out.println("TargetBucketName:" + config.getTargetBucketName());
    System.out.println("LogfilePrefix:" + config.getLogfilePrefix());
    System.out.println("getBucketLogging successfully");
} catch (ObsException e) {
    System.out.println("getBucketLogging failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getBucketLogging failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

16 Static Website Hosting (SDK for Java)

16.1 Overview (SDK for Java)

To host your static website on OBS, you can upload static website files to your bucket as objects, configure the public read permission for the objects, and then configure static website hosting for your bucket.

After this, when third-party users access your websites, they actually access the objects in your bucket in OBS.

When using static website hosting, you can configure request redirection to redirect specific or all requests.

For more information, see "Static Website Hosting" in the *User Guide (Ally Region)*.

16.2 Hosting Website Files in a Bucket (SDK for Java)

Function

To host your static website on OBS, you can upload static website files to your bucket as objects, configure the public read permission for the objects, and then configure static website hosting for your bucket. After this, when third-party users access your website, they are accessing the objects in your bucket.

Restrictions

- To upload an object, you must be the bucket owner or have the required permission (**obs:object:PutObject** in IAM or **PutObject** in a bucket policy).
- To configure an ACL for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketAcl** in IAM or **PutBucketAcl** in a bucket policy).

Method

1. Uploading an object:
obsClient.putObject([PutObjectRequest request](#))
2. Configuring an object ACL:
obsClient.setObjectAcl([SetObjectAclRequest acl](#))

Request Parameters for Uploading an Object

Table 16-1 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	PutObjectRequest	Yes	Explanation: Request parameters for uploading an object. For details, see Table 16-2 .

Table 16-2 PutObjectRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
metadata	ObjectMetadata	No	Explanation: Object metadata. For details, see Table 16-14. Default value: None
acl	AccessControlList	No	Explanation: ACL that can be pre-defined when an object is created. Refer to Table 16-7 to choose the option you need. Value range: - To use a pre-defined ACL, see Table 16-16 for the available options. - To use a user-defined ACL, see Table 16-7 to configure the required parameters. Default value: None
input	java.io.InputStream	No	Explanation: Data stream of the object to be uploaded. Default value: None
file	java.io.File	No	Explanation: File stream of the object to be uploaded. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
extensionPermissionMap	Map<ExtensionObjectPermissionEnum, Set<String>>	No	Explanation: A permission map for granting bucket ACL permissions to one or more accounts. ExtensionObjectPermissionEnum specifies the permissions to grant, and Set<String> describes the list of account IDs (indicated by domain_id) the granted permissions apply to. Value range: - For details about the available permissions, see Table 16-6. - To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
expires	int	No	Explanation: Expiration time of the object. The calculation starts from the time when the object was last modified. Once the object expires, it is automatically deleted. Restrictions: This parameter can be configured only during object upload and cannot be modified by calling a metadata API. Value range: An integer greater than 0, in days. Default value: None
progressListener	Progress Listener	No	Explanation: Upload progress. For details, see Table 16-3.

Parameter	Type	Mandatory (Yes/No)	Description
encodeHeaders	boolean	No	Explanation: Whether to enable OBS to automatically encode request headers. Value range: true: Encoding with SDK is enabled. false: Encoding with SDK is disabled. Default value: true

Table 16-3 ProgressListener

Method	Return Value Type	Mandatory (Yes/No)	Description
progressChanged	void	Yes	Explanation: Used for obtaining the progress. For details, see Table 16-4 . Default value: None

Table 16-4 progressChanged

Parameter	Type	Mandatory (Yes/No)	Description
status	ProgressStatus	Yes	Explanation: Progress data. For details, see Table 16-5 . Default value: None

Table 16-5 ProgressStatus

Method	Return Value Type	Description
getAverageSpeed()	double	Average transmission rate.
getInstantaneous-Speed()	double	Instantaneous transmission rate.
getTransferPercentage()	int	Transmission progress, in percentage.
getNewlyTransferredBytes()	long	Number of the newly transmitted bytes.
getTransferredBytes() ()	long	Number of bytes that have been transmitted.
getTotalBytes()	long	Number of the bytes to be transmitted.

Table 16-6 ExtensionObjectPermissionEnum

Constant	Description
GRANT_READ	Grants a specific tenant the permissions to read the object and object metadata.
GRANT_READ_ACP	Grants a specific tenant the permissions to obtain the object ACL.
GRANT_WRITE_ACP	Grants a specific tenant the permissions to write the object ACL.
GRANT_FULL_CONTROL	Grants a specific tenant the permissions to read the content, metadata, and ACL of the object and write the object ACL.

Constant	Default Value
OBS_KMS	kms

Constant	Default Value
KMS	kms
AES256	AES256

Constant	Default Value
AES256	AES256

Table 16-7 AccessControlList

Parameter	Type	Mandatory (Yes/No)	Type
owner	Owner	No	Explanation: Bucket owner information. For details, see Table 16-8 .
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false
grants	Set< GrantAndPermission >	No	Explanation: Grantee information. For details, see Table 16-9 .

Table 16-8 Owner

Parameter	Type	Mandatory (Yes/No)	Description
id	String	Yes	Explanation: Account (domain) ID of the bucket owner. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
display Name	String	No	Explanation: Account name of the owner. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 16-9 GrantAndPermission

Parameter	Type	Mandatory (Yes/No)	Description
grantee	GranteeInterface	Yes	Explanation: Grantees (users or user groups). For details, see Table 16-11 .
permission	Permission	Yes	Explanation: Permissions to grant. Value range: See Table 16-10 . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
delivered	boolean	No	Explanation: Whether the bucket ACL is applied to all objects in the bucket. Value range: true: The bucket ACL is applied to all objects in the bucket. false: The bucket ACL is not applied to any objects in the bucket. Default value: false

Table 16-10 Permission

Constant	Default Value	Description
PERMISSION_READ	READ	Read permission. A grantee with this permission for a bucket can obtain the list of objects, multipart uploads, bucket metadata, and object versions in the bucket. A grantee with this permission for an object can obtain the object content and metadata.
PERMISSION_WRITE	WRITE	Write permission. A grantee with this permission for a bucket can upload, overwrite, and delete any object or part in the bucket. This permission is not available for objects.
PERMISSION_READ_ACP	READ_ACP	Permission to read an ACL. A grantee with this permission can obtain the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default.

Constant	Default Value	Description
PERMISSION_WRITE_ACP	WRITE_ACP	Permission to modify an ACL. A grantee with this permission can update the ACL of a bucket or object. A bucket or object owner has this permission for their bucket or object by default. This permission allows the grantee to change the access control policies, meaning the grantee has full control over a bucket or object.
PERMISSION_FULL_CONTROL	FULL_CONTROL	Full control access, including read and write permissions for a bucket and its ACL, or for an object and its ACL. A grantee with this permission for a bucket has READ , WRITE , READ_ACP , and WRITE_ACP permissions for the bucket. A grantee with this permission for an object has READ , READ_ACP , and WRITE_ACP permissions for the object.

Table 16-11 GranteeInterface

Parameter	Type	Mandatory (Yes/No)	Description
CanonicalGrantee	CanonicalGrantee	Yes	Explanation: Grantee (user) information. For details, see Table 16-12 .
GroupGrantee	GroupGrantee	Yes	Explanation: Grantee (user group) information. Value range: See Table 16-13 . Default value: None

Table 16-12 CanonicalGrantee

Parameter	Type	Mandatory (Yes/No)	Description
grantId	String	Yes if Type is set to Grantee User	Explanation: Account (domain) ID of the grantee. Value range: To obtain the account ID, see How Do I Get My Account ID and User ID? Default value: None
displayName	String	No	Explanation: Account name of the grantee. Value range: To obtain the account name, see How Do I Get My Account ID and User ID? Default value: None

Table 16-13 GroupGrantee

Constant	Description
ALL_USERS	All users.
AUTHENTICATED_USERS	Authorized users. This constant is deprecated.
LOG_DELIVERY	Log delivery group. This constant is deprecated.

Table 16-14 ObjectMetadata

Parameter	Type	Man dator y (Yes/ No)	Description
contentLength	Long	No	Explanation: Object size. Restrictions: • The object size in a single upload ranges from 0 to 5 GB. • To upload files larger than 5 GB, multipart uploads should be used. Default value: If this parameter is not specified, the SDK automatically calculates the size of the object.
contentType	String	No	Explanation: MIME type of the object file. MIME type is a standard way of describing a data type and is used by the browser to decide how to display data. Value range: See What Is Content-Type (MIME)? (Java SDK) Default value: If this parameter is not specified, the SDK determines the file type based on the suffix of the object name and assigns a value to the parameter. For example, if the suffix of the object name is .xml , the object is an application/xml file. If the suffix is .html , the object is a text/html file.
contentEncoding	String	No	Explanation: Content-Encoding header in the response. It specifies which encoding is applied to the object. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentDisposition	String	No	Explanation: Provides a default file name for the requested object. When the object with the default file name is being downloaded or accessed, the content is displayed as part of a web page in the browser or as an attachment in a download dialog box. Default value: None
cacheControl	String	No	Explanation: Cache-Control header in the response. It specifies the cache behavior of the web page when an object is downloaded. Default value: None
contentType	String	No	Explanation: Language or language combination for visitors to customize and use. For details, see the definition of ContentLanguage in the HTTP protocol. Default value: None
expires	String	No	Explanation: The time a cached web page object expires. Restrictions: The time must be in the GMT format. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
contentMd5	String	No	Explanation: Base64-encoded MD5 value of the file data. It is provided for the OBS server to verify data integrity. The OBS server will compare this MD5 value with the MD5 value calculated based on the file data. If the two values are not the same, HTTP status code 400 is returned. Restrictions: - The MD5 value of the file must be Base64 encoded. - If the MD5 value is not specified, the OBS server will not verify the MD5 value of the object. Value range: Base64-encoded 128-bit MD5 value of the request body calculated according to RFC 1864. Example: n58IG6hfM7vql4K0vnWpog== Default value: None
storageClasses	StorageClassEnum	No	Explanation: Storage class of an object that can be specified at object creation. If you do not specify this header, the object inherits the storage class of the bucket. Value range: See Table 16-15. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
webSiteRedirectLocation	String	No	Explanation: If the bucket is configured with website hosting, the request for obtaining the object can be redirected to another object in the bucket or an external URL. This parameter specifies the address the request for the object is redirected to. The request is redirected to an object anotherPage.html in the same bucket: WebsiteRedirectLocation:/anotherPage.html The request is redirected to an external URL http://www.example.com/: WebsiteRedirectLocation:http://www.example.com/ Restrictions: - The value must start with a slash (/), http://, or https:// and cannot exceed 2 KB. - OBS only supports redirection for objects in the root directory of a bucket. Default value: None
nextPosition	long	No	Explanation: Start position for the next append upload. Value range: 0 to the object length, in bytes. Default value: None
appendable	boolean	No	Explanation: Whether the object is appendable. Value range: true: The object is appendable. false: The object is not appendable. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
userMetadata	Map<String, Object>	No	Explanation: User-defined metadata of the object. To define it, you can add a header starting with x-obs-meta- in the request. In Map, the String key indicates the name of the user-defined metadata that starts with x-obs-meta-, and the Object value indicates the value of the user-defined metadata. To obtain the user-defined metadata of an object, use ObsClient.getObjectMetadata. For details, see Obtaining Object Metadata (SDK for Java). Restrictions: • An object can have multiple pieces of metadata. The size of the metadata cannot exceed 8 KB in total. • When you call ObsClient.getObject to download an object, its user-defined metadata will also be downloaded. Default value: None

Table 16-15 StorageClassEnum

Constant	Default Value	Description
STANDARD	STANDARD	Standard storage class
WARM	WARM	Warm storage class.
COLD	COLD	Cold storage class.
INTELLIGENT_TIERING	INTELLIGENT_TIERING	The Intelligent Tiering storage class is designed to optimize storage costs by automatically moving data to a more economical access tier when data access patterns change. This storage class is ideal for data with constantly changing or unpredictable access patterns.

Table 16-16 Pre-defined ACL

Constant	Description
AccessControllist.REST_CAN_NED_PRIVATE	Private read/write. A bucket or object can only be accessed by its owner.
AccessControllist.REST_CAN_NED_PUBLIC_READ	Public read. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, bucket metadata, and object versions in the bucket. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE	Public read/write. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart uploads, upload parts, assemble parts, copy parts, and abort multipart upload tasks. If this permission is granted on an object, anyone can read the content and metadata of the object.
AccessControllist.REST_CAN_NED_PUBLIC_READ_DELIVERED	Public read on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart tasks, and bucket metadata, and can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.
AccessControllist.REST_CAN_NED_PUBLIC_READ_WRITE_DELIVERED	Public read/write on a bucket as well as objects in the bucket. If this permission is granted on a bucket, anyone can read the object list, multipart uploads, and bucket metadata, and can upload or delete objects, initiate multipart upload tasks, upload parts, assemble parts, copy parts, and abort multipart uploads. They can also read the content and metadata of the objects in the bucket. This permission cannot be granted on objects.

Constant	Description
AccessControlList.REST_CANNED_BUCKET_OWNER_FULL_CONTROL	If this permission is granted on an object, only the bucket and object owners have the full control over the object. By default, if you upload an object to a bucket owned by another user, the bucket owner does not have the permissions on your object. After you grant this permission to the bucket owner, the bucket owner can have full control over your object. For example, if user A uploads object x to user B's bucket, user B does not have the control over object x. If user A sets bucket-owner-full-control for object x, user B then has the control over object x.

Request Parameters for Setting an Object ACL

Table 16-17 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	SetObjectACLRequest	Yes	Explanation: Request parameters for setting an object ACL. For details, see Table 16-18 .

Table 16-18 SetObjectAclRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
objectKey	String	Yes	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None
versionId	String	No	Explanation: Object version ID. Value range: The value must contain 32 characters. Default value: None
acl	AccessControlList	No	Explanation: An ACL specified for the object. You can use either a pre-defined or a user-defined ACL. Value range: - To use a pre-defined ACL, see Table 16-16 for the available options. - To use a user-defined ACL, see Table 16-7 to configure the required parameters. Default value: AccessControlList.REST_CANNED_PRIVATE

Responses for Uploading an Object

Table 16-19 PutObjectResult

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: Response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None
storageClass	StorageClassEnum	Explanation: Object storage class. If the storage class is Standard, leave this parameter blank. Value range: See Table 16-15. Default value: None
versionId	String	Explanation: Object version ID. If versioning is enabled for the bucket, the object version number will be returned. Value range: The value must contain 32 characters. Default value: None

Parameter	Type	Description
etag	String	Explanation: ETag of an object, which is a Base64-encoded 128-bit MD5 digest. ETag is the unique identifier of the object content. It can be used to determine whether the object content is changed. For example, if the ETag is A when an object is uploaded and is B when the object is downloaded, the object content is changed. The ETag reflects changes only to the contents of the object, not its metadata. An object created by an upload or copy operation has a unique ETag. Restrictions: If an object is encrypted using server-side encryption, the ETag is not the MD5 value of the object. Value range: The value must contain 32 characters. Default value: None
objectKey	String	Explanation: Object name. An object is uniquely identified by an object name in a bucket. An object name is a complete path that does not contain the bucket name. Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Description
bucketName	String	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses for Setting an Object ACL

Table 16-20 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None

Parameter	Type	Description
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example configures static website hosting for bucket **examplebucket** by specifying the MIME type and setting the object ACL to public read.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.AccessControlList;
import com.obs.services.model.ObjectMetadata;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject001
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // Sample code is as follows:
            // Upload an object and set the MIME type for the object.
            PutObjectRequest request = new PutObjectRequest();
            request.setBucketName("examplebucket");
            request.setObjectKey("test.html");
            request.setFile(new File("localfile.html"));
            ObjectMetadata metadata = new ObjectMetadata();
            metadata.setContentType("text/html");
            request.setMetadata(metadata);
            obsClient.putObject(request);
            // Set the object ACL to public read.
            obsClient.setObjectAcl("examplebucket", "test.html",
            AccessControlList.REST_CANNED_PUBLIC_READ);
            System.out.println("putObject successfully");
        } catch (ObsException e) {
```

```
        System.out.println("putObject failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code:" + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message:" + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("putObject failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

NOTE

In this code example, you can use **`https://bucketname.your-endpoint/test.html`** to access the hosted file in the browser.

16.3 Configuring Static Website Hosting (SDK for Java)

Function

You can host static website resources such as HTML web pages, flash files, or audio and video files in an OBS bucket, so that you can provide these hosted resources using the bucket's website endpoint to end users. Typical use cases include:

- Redirecting all requests to another website
- Redirecting specific requests to another website

This API configures static website hosting for a bucket.

Restrictions

- Periods (.) should be avoided in the target bucket name, or there may be certificate verification failures on the client when you use HTTPS for access.
- The request body of the website configuration cannot exceed 10 KB.
- To configure static website hosting for a bucket, you must be the bucket owner or have the required permission (**obs:bucket:PutBucketWebsite** in IAM or **PutBucketWebsite** in a bucket policy).

Method

obsClient.setBucketWebsite(final [SetBucketWebsiteRequest request](#))

Request Parameters

Table 16-21 List of request parameters

Parameter	Type	Mandato ry (Yes/No)	Description
request	SetBucketWebsite Request	Yes	Explanation: Request parameters for configuring website hosting. For details, see Table 16-22 .

Table 16-22 SetBucketWebsiteRequest

Parameter	Type	Man dato ry (Yes /No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Value range: The value must contain 3 to 63 characters.
websiteConfig	WebsiteConf iguration	Yes	Explanation: Static website hosting configurations. For details, see Table 16-23 .

Table 16-23 WebsiteConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
suffix	String	No	Explanation: Suffix that is appended to the request for a directory. For example, if the suffix is index.html and you request samplebucket/images/, the returned data will be for the object named images/index.html in the bucket samplebucket. Restrictions: • key, suffix, and routeRules must be used together and they cannot be used with redirectAllRequestsTo. • When key, suffix, and routeRules are used together, routeRules can be left blank. • If you configure key, suffix, and routeRules, they must not be all left blank. Value range: This parameter can neither be left blank nor contain slashes (/). Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
key	String	No	Explanation: Object name to use when a 4XX error occurs. This parameter specifies the web page to display when an error occurs. Restrictions: • key, suffix, and routeRules must be used together and they cannot be used with redirectAllRequestsTo. • When key, suffix, and routeRules are used together, routeRules can be left blank. • If you configure key, suffix, and routeRules, they must not be all left blank. Value range: The value must contain 1 to 1,024 characters. Default value: None
routeRules	List< Route Rule >	No	Explanation: List of routing rules. For details, see Table 16-26. Restrictions: • key, suffix, and routeRules must be used together and they cannot be used with redirectAllRequestsTo. • When key, suffix, and routeRules are used together, routeRules can be left blank. • If you configure key, suffix, and routeRules, they must not be all left blank. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
redirectAllRequestsTo	RedirectAllRequest	No	Explanation: Redirection rules for all requests. For details, see Table 16-24 . Restrictions: • key, suffix, and routeRules must be used together and they cannot be used with redirectAllRequestsTo. • If you configure redirectAllRequestsTo, it must not be left blank. Default value: None

Table 16-24 RedirectAllRequest

Parameter	Type	Mandatory (Yes/No)	Description
hostName	String	Yes	Explanation: Host name used for redirection, for example, www.example.com . Restrictions: The host name must comply with the host name rules. Default value: None
protocol	ProtocolEnum	No	Explanation: Protocol used for redirection. Value range: See Table 16-25 . Default value: None

Table 16-25 ProtocolEnum

Constant	Default Value	Description
HTTP	http	HTTP protocol used for redirection.
HTTPS	https	HTTPS protocol used for redirection.

Table 16-26 RouteRule

Parameter	Type	Man dator y (Yes/ No)	Description
condition	RouteRuleCondition	No	Explanation: Conditions of a redirection rule. For details, see Table 16-27 .
redirect	Redirect	Yes	Explanation: Details about the redirection. For details, see Table 16-28 .

Table 16-27 RouteRuleCondition

Parameter	Type	Mandatory (Yes/No)	Description
keyPrefixEquals	String	No	Explanation: Object name prefix for the redirection to take effect. If the name prefix of the requested object is the same as the value specified for this parameter, the redirection rule takes effect. For example, to redirect the requests for the object ExamplePage.html, set keyPrefixEquals to ExamplePage.html. Restrictions: This parameter cannot be used together with httpErrorCodeReturnedEquals. Value range: The value must contain 1 to 1,024 characters. Default value: None
httpErrorCodeReturnedEquals	String	No	Explanation: HTTP error code for the redirection to take effect. If there is an error, and the error code returned is the same as the value specified for this parameter, the redirection rule takes effect. For example, if you want to redirect requests to NotFound.html when HTTP error code 404 is returned, set httpErrorCodeReturnedEquals to 404 in RouteRuleCondition, and set replaceKeyWith to NotFound.html in Redirect. Restrictions: This parameter cannot be used together with keyPrefixEquals. Default value: None

Table 16-28 Redirect

Parameter	Type	Mandatory (Yes/No)	Description
Protocol	ProtocolEnum	No	Explanation: Protocol used for redirection. Value range: See Table 16-25 . Default value: None
hostName	String	No	Explanation: Host name used for redirection. Default value: None
replaceKeyPrefix-With	String	No	Explanation: Object name prefix used in the redirection request. Restrictions: This parameter cannot be used together with replaceKeyWith . Value range: The value must contain 1 to 1,024 characters. Default value: None
replaceKeyWith	String	No	Explanation: Object name used in the redirection request. Restrictions: This parameter cannot be used together with replaceKeyPrefixWith . Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
httpRedirectCode	String	No	Explanation: HTTP status code in the response to the redirect request. Default value: None

Responses

Table 16-29 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Example: Configuring the Default Home Page and Error Pages

This example configures the default homepage and error pages for bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.WebsiteConfiguration;
public class SetBucketWebsite001
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
```

```
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Sample code is as follows:
    WebsiteConfiguration config = new WebsiteConfiguration();
    // Configure the default homepage.
    config.setSuffix("index.html");
    // Configure the error pages.
    config.setKey("error.html");
    obsClient.setBucketWebsite("examplebucket", config);
    System.out.println("setBucketWebsite successfully");
} catch (ObsException e) {
    System.out.println("setBucketWebsite failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketWebsite failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Configuring a Redirection Rule

This example configures a redirection rule for bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ProtocolEnum;
import com.obs.services.model.Redirect;
import com.obs.services.model.RouteRule;
import com.obs.services.model.RouteRuleCondition;
import com.obs.services.model.WebsiteConfiguration;
public class SetBucketWebsite002
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
```

advised not to use hard coding, which may result in information leakage.

```
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Sample code is as follows:
    WebsiteConfiguration config = new WebsiteConfiguration();
    // Configure the default homepage.
    config.setSuffix("index.html");
    // Configure the error pages.
    config.setKey("error.html");
    RouteRule rule = new RouteRule();
    Redirect r = new Redirect();
    r.setHostName("www.example.com");
    r.setHttpRedirectCode("305");
    r.setRedirectProtocol(ProtocolEnum.HTTP);
    r.setReplaceKeyPrefixWith("replacekeyprefix");
    rule.setRedirect(r);
    RouteRuleCondition condition = new RouteRuleCondition();
    condition.setHttpErrorCodeReturnedEquals("404");
    condition.setKeyPrefixEquals("keyprefix");
    rule.setCondition(condition);
    config.getRouteRules().add(rule);
    obsClient.setBucketWebsite("examplebucket", config);
    System.out.println("setBucketWebsite successfully");
} catch (ObsException e) {
    System.out.println("setBucketWebsite failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketWebsite failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

Code Example: Configuring Redirection for All Requests

This example configures redirection of all requests for bucket **examplebucket**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ProtocolEnum;
import com.obs.services.model.RedirectAllRequest;
import com.obs.services.model.WebsiteConfiguration;
public class SetBucketWebsite003
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
```

```
hard coding may result in leakage.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is to be created.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use the permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use the temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Sample code is as follows:
    WebsiteConfiguration config = new WebsiteConfiguration();
    RedirectAllRequest redirectAll = new RedirectAllRequest();
    redirectAll.setHostName("www.example.com");
    redirectAll.setRedirectProtocol(ProtocolEnum.HTTP);
    config.setRedirectAllRequestsTo(redirectAll);
    obsClient.setBucketWebsite("examplebucket", config);
    System.out.println("setBucketWebsite successfully");
} catch (ObsException e) {
    System.out.println("setBucketWebsite failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("setBucketWebsite failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

16.4 Obtaining Static Website Hosting Configurations (SDK for Java)

Function

You can host static website resources such as HTML web pages, flash files, or audio and video files in an OBS bucket, so that you can provide these hosted resources using the bucket's website endpoint to end users. Typical use cases include:

- Redirecting all requests to another website
- Redirecting specific requests to another website

This API returns the static website hosting configurations of the bucket.

Restrictions

- To obtain the static website hosting configurations of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:GetBucketWebsite** in IAM or **GetBucketWebsite** in a bucket policy).

Method

obsClient.getBucketWebsite(final [BaseBucketRequest request](#))

Request Parameters

Table 16-30 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters related to basic bucket information. For details, see Table 16-31 .

Table 16-31 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 16-32 WebsiteConfiguration

Parameter	Type	Mandatory (Yes/No)	Description
suffix	String	No	Explanation: Suffix that is appended to the request for a directory. For example, if the suffix is index.html and you request samplebucket/images/ , the returned data will be for the object named images/index.html in the bucket samplebucket . Value range: This parameter can neither be left blank nor contain slashes (/). Default value: None
key	String	No	Explanation: Object name to use when a 4XX error occurs. This parameter specifies the web page to display when an error occurs. Value range: The value must contain 1 to 1,024 characters. Default value: None
redirectAllRequestsTo	RedirectAllRequest	No	Explanation: Redirection rules for all requests. For details, see Table 16-33 .
routeRules	List< RouteRule >	No	Explanation: List of routing rules. For details, see Table 16-35 .

Table 16-33 RedirectAllRequest

Parameter	Type	Man dato ry (Yes/ No)	Description
hostName	String	Yes	Explanation: Host name used for redirection, for example, www.example.com . Restrictions: The host name must comply with the host name rules. Default value: None
protocol	ProtocolEnum	No	Explanation: Protocol used for redirection. Value range: See Table 16-34 . Default value: None

Table 16-34 ProtocolEnum

Constant	Default Value	Description
HTTP	http	HTTP protocol used for redirection.
HTTPS	https	HTTPS protocol used for redirection.

Table 16-35 RouteRule

Parameter	Type	Man dator y (Yes/ No)	Description
condition	RouteRuleCondition	No	Explanation: Conditions of a redirection rule. For details, see Table 16-36 .
redirect	Redirect	Yes	Explanation: Details about the redirection. For details, see Table 16-37 .

Table 16-36 RouteRuleCondition

Parameter	Type	Mandatory (Yes/No)	Description
keyPrefixEquals	String	No	Explanation: Object name prefix for the redirection to take effect. If the name prefix of the requested object is the same as the value specified for this parameter, the redirection rule takes effect. For example, to redirect the requests for the object ExamplePage.html, set keyPrefixEquals to ExamplePage.html. Restrictions: This parameter cannot be used together with httpErrorCodeReturnedEquals. Value range: The value must contain 1 to 1,024 characters. Default value: None
httpErrorCodeReturnedEquals	String	No	Explanation: HTTP error code for the redirection to take effect. If there is an error, and the error code returned is the same as the value specified for this parameter, the redirection rule takes effect. For example, if you want to redirect requests to NotFound.html when HTTP error code 404 is returned, set httpErrorCodeReturnedEquals to 404 in RouteRuleCondition, and set replaceKeyWith to NotFound.html in Redirect. Restrictions: This parameter cannot be used together with keyPrefixEquals. Default value: None

Table 16-37 Redirect

Parameter	Type	Mandatory (Yes/No)	Description
Protocol	ProtocolEnum	No	Explanation: Protocol used for redirection. Value range: See Table 16-34 . Default value: None
hostName	String	No	Explanation: Host name used for redirection. Default value: None
replaceKeyPrefix-With	String	No	Explanation: Object name prefix used in the redirection request. Restrictions: This parameter cannot be used together with replaceKeyWith . Value range: The value must contain 1 to 1,024 characters. Default value: None
replaceKeyWith	String	No	Explanation: Object name used in the redirection request. Restrictions: This parameter cannot be used together with replaceKeyPrefixWith . Value range: The value must contain 1 to 1,024 characters. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
httpRedirectCode	String	No	Explanation: HTTP status code in the response to the redirect request. Default value: None

Code Examples

This example returns the website configuration of bucket **examplebucket** using **obsClient.getBucketWebsite**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.RouteRule;
import com.obs.services.model.WebsiteConfiguration;
public class GetBucketWebsite001
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

        try {
            // View the website configuration of the bucket.
            WebsiteConfiguration config = obsClient.getBucketWebsite("examplebucket");
            System.out.println("Key:" + config.getKey());
            System.out.println("Suffix:" + config.getSuffix());
            for(RouteRule rule : config.getRouteRules()){
                System.out.println("rule:" + rule);
            }
            System.out.println("getBucketWebsite successfully");
        } catch (ObsException e) {
            System.out.println("getBucketWebsite failed");
            // Request failed. Print the HTTP status code.
            System.out.println("HTTP Code:" + e.getResponseCode());
            // Request failed. Print the server-side error code.
            System.out.println("Error Code:" + e.getErrorCode());
            // Request failed. Print the error details.
            System.out.println("Error Message:" + e.getErrorMessage());
            // Request failed. Print the request ID.
        }
    }
}
```

```
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
        e.printStackTrace();
    } catch (Exception e) {
        System.out.println("getBucketWebsite failed");
        // Print other error information.
        e.printStackTrace();
    }
}
```

16.5 Deleting Static Website Hosting Configurations (SDK for Java)

Function

You can host static website resources such as HTML web pages, flash files, or audio and video files in an OBS bucket, so that you can provide these hosted resources using the bucket's website endpoint to end users. Typical use cases include:

- Redirecting all requests to another website
- Redirecting specific requests to another website

This API deletes the static website hosting configurations of a bucket.

Restrictions

- To delete the static website hosting configurations of a bucket, you must be the bucket owner or have the required permission (**obs:bucket:DeleteBucketWebsite** in IAM or **DeleteBucketWebsite** in a bucket policy).

Method

obsClient.deleteBucketWebsite(final [BaseBucketRequest request](#))

Request Parameters

Table 16-38 List of request parameters

Parameter	Type	Mandatory (Yes/No)	Description
request	BaseBucketRequest	Yes	Explanation: Request parameters related to basic bucket information. For details, see Table 16-39 .

Table 16-39 BaseBucketRequest

Parameter	Type	Mandatory (Yes/No)	Description
bucketName	String	Yes	Explanation: Bucket name. Restrictions: • A bucket name must be unique across all accounts and regions. • A bucket name: – Must be 3 to 63 characters long and start with a digit or letter. Lowercase letters, digits, hyphens (-), and periods (.) are allowed. – Cannot be formatted as an IP address. – Cannot start or end with a hyphen (-) or period (.). – Cannot contain two consecutive periods (..), for example, my..bucket. – Cannot contain periods (.) and hyphens (-) adjacent to each other, for example, my-.bucket or my.-bucket. • If you repeatedly create buckets of the same name in the same region, no error will be reported and the bucket attributes comply with those set in the first creation request. Default value: None

Responses

Table 16-40 Common response headers

Parameter	Type	Description
statusCode	int	Explanation: HTTP status code. Value range: A status code is a group of digits that can be 2xx (indicating successes) or 4xx or 5xx (indicating errors). It indicates the status of a response. Default value: None
responseHeaders	Map<String, Object>	Explanation: HTTP response header list, composed of tuples. In a tuple, the String key indicates the name of the header, and the Object value indicates the value of the header. Default value: None

Code Examples

This example deletes the website configuration of bucket **examplebucket** using **obsClient.deleteBucketWebsite**.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
public class DeleteBucketWebsite001
{
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is to be created.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");

        // Create an ObsClient instance.
        // Use the permanent AK/SK pair to initialize the client.
        ObsClient obsClient = new ObsClient(ak, sk, endPoint);
        // Use the temporary AK/SK pair and security token to initialize the client.
        // ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);
    }
}
```

```
try {
    // Delete the website configuration of the bucket.
    obsClient.deleteBucketWebsite("examplebucket");
    System.out.println("deleteBucketWebsite successfully");
} catch (ObsException e) {
    System.out.println("deleteBucketWebsite failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("deleteBucketWebsite failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

17 Client-Side Encryption (SDK for Java)

17.1 Overview (SDK for Java)

Client-side encryption is a process where data is encrypted using the selected encryption method and information on your local PC before it is transmitted to an OBS server. During this process, the encryption method used and necessary information required for decryption will be stored in object metadata. During a download, the OBS SDK decrypts the data based on the key provided and the information stored in object metadata, and then returns the decrypted data.



WARNING

- OBS does not store your master key in any way. Keep your master key correct and intact. If your master key is lost or mistakenly used, your data cannot be decrypted.
 - Do not modify the decryption information stored in object metadata when moving or replicating an encrypted object or modifying the object metadata. If you do so, your data cannot be decrypted.
 - For security purposes, later SDK versions will use RSA/ECB/OAEPWITHSHA-256ANDMGF1PADDING in place of RSA for encryption and, therefore, will be incompatible with earlier versions.
 - You are advised to add the SDK version information to **master-key-info** to avoid data encryption or decryption failures caused by changes of encryption algorithms and key length restrictions.
 - Consider both security and performance when determining the RSA key size. Later SDK versions will require the key size be larger than 3,072 bits.
 - Consider both security and performance when configuring **secureRandom**.
 - RSA keys in PKCS #8 format are recommended.
-

Encryption Process and Cipher Suites

The OBS SDK for Java offers two cipher suite generators: CTRCipherGenerator based on AES-CTR, and CtrRSACipherGenerator based on RSA and AES-CTR.

If CTRCipherGenerator is used to upload objects, you need to provide a data key. The SDK then randomly generates an initial value for each object and uses the data key and initial value to encrypt the object. After the encryption, the SDK uploads the encrypted object to OBS and stores its initial value in object metadata. To download this object, you need to use the same data key used for upload. The SDK automatically obtains the initial value from object metadata, uses the data key provided and initial value to decrypt the object, and returns the decrypted data. If a different data key is provided for download, the SDK returns an unavailable decrypted file.

If CtrRSACipherGenerator is used to upload objects, you need to provide an RSA public key. The SDK then randomly generates a data key and initial value for each object and uses them to encrypt the object. After that, the SDK uploads the encrypted object to OBS and then uses the provided RSA key to encrypt the data key. The encrypted data key and initial value are stored in the object metadata. To download this object, you need to provide the corresponding RSA private key. The SDK then automatically obtains the data key and initial value stored in the object metadata and uses the provided RSA private key to decrypt the data key. If the provided private key does not match the public key used for upload, an error will be reported. After the data key is decrypted, the SDK uses the decrypted data key and initial value to decrypt the object and returns the decrypted data.

API Changes

CryptoObsClient is inherited from ObsClient. Except the APIs listed in the following table, all other APIs of CryptoObsClient are the same as those of ObsClient.

Table 17-1 CryptoObsClient

API	CryptoObsClient API Action
putObject	Encrypts streams or files and then upload them to OBS.
getObject	Returns the decrypted data streams.

Decryption Information in Metadata

The SDK saves the information required for decryption in the user-defined metadata of an object and does not back it up. If you modify the stored information, data cannot be decrypted. The following table describes the information required for decryption.

Table 17-2

Parameter	Mandatory (Yes/No)	Description
encrypted-algorithm	Yes	Information about a cipher suite
encrypted-object-key	Yes when CtrRSACipherGenerator is used	Data key encrypted using an RSA key
encrypted-start	Yes	String of the Base64-encoded initial value used for encryption
master-key-info	No	Information about encryption keys
plaintext-sha256	No	SHA-256 value of the object before being encrypted (not available for streaming uploads)
plaintext-content-length	No	Length of the object before being encrypted (not available for streaming uploads)
encrypted-sha256	No	SHA-256 value of the object after being encrypted (not available for streaming uploads)

17.2 Client-Side Encryption APIs (SDK for Java)

Initializing CryptoCipher

OBS SDK for Java provides two cipher suites for you to choose from.

CtrRSACipherGenerator is inherited from **CTRCipherGenerator** but only requires you to provide an RSA public or private key to encrypt or decrypt the randomly generated data key.

CTRCipherGenerator requires only one data key. This key is used to encrypt all objects.

Table 17-3 CtrRSACipherGenerator parameters

Parameter	Type	Mandatory (Yes/No)	Description
privateKey	PrivateKey	Yes when decrypting objects (for example, getObject)	Explanation: RSA private key. Default value: None
publicKey	PublicKey	Yes for encryption (for example, putObject)	Explanation: RSA public key. Default value: None
masterKeyInfo	String	No	Explanation: Key information, which is stored in the user-defined metadata of objects to help you identify keys. You need to maintain the mappings between keys and masterKeyInfo on your own. Default value: None
secureRandom	SecureRandom	Yes	Explanation: Secure random number generator, which is used to randomly generate cryptoKeyBytes and cryptoIvBytes . Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
needSha256	boolean	No	Explanation: Whether to verify the SHA-256 value of the encrypted data and to add SHA-256 values calculated before and after the encryption to the user-defined metadata. NOTE To reduce memory overheads, the SDK uses streaming computing, which means that a file needs to be read and encrypted twice in a common upload and three times in a resumable upload. Value range: If need_sha256 is set to true, the SDK automatically calculates SHA-256 before and after the object is encrypted and saves the two values to the user-defined metadata of the object. In addition, the SDK adds the SHA-256 value of the encrypted object to the request header when sending a request to the server. After receiving the request, the server calculates the SHA-256 value of the object and checks the consistency between the calculated and the received values. If they are inconsistent, an error message is returned. false: The SHA-256 value of the encrypted data is not verified. Default value: false

Table 17-4 CTRCipherGenerator parameters

Parameter	Type	Mandatory (Yes/No)	Description
masterKeyInfo	String	No	Explanation: Key information, which is stored in the user-defined metadata of objects to help you identify cryptoKeyBytes . You need to maintain the mappings between cryptoKeyBytes and masterKeyInfo on your own. Default value: None
cryptoKeyBytes	byte[]	Yes	Explanation: Data key used for encrypting data. Restrictions: The value must be 32 bytes long. Default value: None
cryptolvBytes	byte[]	No	Explanation: Initial value used for encrypting data. Restrictions: • The value must be 16 bytes long. • If this parameter is specified, all objects are encrypted using the specified initial value. If this parameter is not specified, the SDK randomly generates an initial value for each object. Default value: None
secureRandom	SecureRandom	Yes	Explanation: Secure random number generator, for use when either cryptoKeyBytes or cryptolvBytes is missing. Default value: None

Parameter	Type	Mandatory (Yes/No)	Description
needSha256	boolean	No	Explanation: Whether to verify the SHA-256 value of the encrypted data and to add SHA-256 values calculated before and after the encryption to the user-defined metadata. NOTE To reduce memory overheads, the SDK uses streaming computing, which means that a file needs to be read and encrypted twice in a common upload and three times in a resumable upload. Value range: If need_sha256 is set to true, the SDK automatically calculates SHA-256 before and after the object is encrypted and saves the two values to the user-defined metadata of the object. In addition, the SDK adds the SHA-256 value of the encrypted object to the request header when sending a request to the server. After receiving the request, the server calculates the SHA-256 value of the object and checks the consistency between the calculated and the received values. If they are inconsistent, an error message is returned. false: The SHA-256 value of the encrypted data is not verified. Default value: false

Initializing CryptoObsClient

[CryptoObsClient](#) is inherited from [ObsClient](#). For its configuration details, see [Creating and Configuring an OBS Client \(SDK for Java\)](#).

Method

```
CryptoObsClient(String accessKey, String secretKey, String endPoint,
    CTCipherGenerator ctrCipherGenerator)
```

Table 17-5 Parameters for initializing CryptoObsClient

Parameter	Description	Recommended Value
accessKey	Explanation: Access key ID (AK). Default value: An empty string, indicating an anonymous user.	N/A
secretKey	Explanation: Secret access key (SK). Default value: An empty string, indicating an anonymous user.	N/A
endPoint	Explanation: OBS server address. It consists of a protocol type, domain name, and port number, for example, https://your-endpoint:443 . For security purposes, you are advised to use HTTPS. Default value: None	N/A
ctrCipherGenerator	Explanation: Cipher suite used by the client. Value range: • CtrRSACipherGenerator • CTRCipherGenerator Default value: None	N/A

APIs That Support Client-Side Encryption

Table 17-6 APIs that support client-side encryption

Method	Description
putObject	Uploads an object.
getObject	Downloads an object.

Code Examples

CtrRSACipherGenerator:

```
import com.obs.services.ObsConfiguration;
import com.obs.services.crypto.CTRCipherGenerator;
import com.obs.services.crypto.CryptoObsClient;
import com.obs.services.crypto.CtrRSACipherGenerator;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObsObject;
import com.obs.services.model.PutObjectResult;

import java.io.File;
import java.io.FileOutputStream;
import java.io.IOException;
import java.io.InputStream;
import java.security.NoSuchAlgorithmException;
import java.security.PrivateKey;
import java.security.PublicKey;
import java.security.spec.InvalidKeySpecException;

public class CtrRSACipherGeneratorDemo001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
        // Obtain an endpoint using environment variables or import it in other ways.
        //String endPoint = System.getenv("ENDPOINT");
        CtrRSACipherGenerator ctrRSACipherGenerator = null;
        try {
            String examplePrivateKeyPath = "yourRSAPrivateKeyPath";
            String examplePublicKeyPath = "yourRSAPublicKeyPath";
            ObsConfiguration config = new ObsConfiguration();
            PrivateKey privateKeyObj = CtrRSACipherGenerator.importPKCS8PrivateKey(examplePrivateKeyPath);
            PublicKey publicKeyObj = CtrRSACipherGenerator.importPublicKey(examplePublicKeyPath);
            ctrRSACipherGenerator =
                new CtrRSACipherGenerator(
                    "example_master_key_info", true, config.getSecureRandom(), privateKeyObj,
                    publicKeyObj);
        } catch (IllegalArgumentException | IOException | NoSuchAlgorithmException |
        InvalidKeySpecException e) {
            e.printStackTrace();
        }
        assert ctrRSACipherGenerator != null;
        // Create an ObsClient instance.
        try (CryptoObsClient cryptoObsClient = new CryptoObsClient(ak, sk, securityToken, endPoint,
        ctrRSACipherGenerator)) {
            String exampleBucketName = "example-bucket";
            String exampleObjectKey = "exampleObjectKey";
            String examplePlainTextFilePath = "examplePlainTextFilePath";
            String exampleDecryptedFilePath = "exampleDecryptedFilePath";
            PutObjectResult putObjectResult =
                cryptoObsClient.putObject(exampleBucketName, exampleObjectKey, new
        File(examplePlainTextFilePath));
            System.out.println("HTTP Code: " + putObjectResult.getStatusCode());
            System.out.println("Etag: " + putObjectResult.getEtag());
            // The object is successfully encrypted on the client and then uploaded.
            System.out.println("CtrRSACipherGeneratorDemo001 putObject successfully");
        }
```

```
GetObjectRequest getObjectRequest = new GetObjectRequest(exampleBucketName,
exampleObjectKey);
ObsObject obsObject = cryptoObsClient.getObject(getObjectRequest);
InputStream input = obsObject.getObjectContent();
byte[] b = new byte[1024];
FileOutputStream fileOutputStream = new FileOutputStream(exampleDecryptedFilePath);
int len;
while ((len = input.read(b)) != -1) {
    fileOutputStream.write(b, 0, len);
}
fileOutputStream.close();
input.close();

System.out.println("HTTP Code: " + obsObject.getMetadata().getStatusCode());
// The object is successfully decrypted on the client and then downloaded.
System.out.println("CtrRSACipherGeneratorDemo001 getObject successfully");

// Check whether the file remains the same before encryption and after decryption.
byte[] plainTextFileSha256 = CTCipherGenerator.getFileSha256Bytes(examplePlainTextFilePath);
byte[] decryptedFileSha256 = CTCipherGenerator.getFileSha256Bytes(exampleDecryptedFilePath);
String plainTextFileSha256Base64Encoded =
CTCRCipherGenerator.getBase64Info(plainTextFileSha256);
String decryptedFileSha256Base64Encoded =
CTCRCipherGenerator.getBase64Info(decryptedFileSha256);
System.out.println("plainTextFileSha256 base64 encoded: " + plainTextFileSha256Base64Encoded);
System.out.println("decryptedFileSha256 base64 encoded: " + decryptedFileSha256Base64Encoded);
System.out.println(
    "plainTextFileSha256 equals decryptedFileSha256 ? "
    + decryptedFileSha256Base64Encoded.equals(plainTextFileSha256Base64Encoded));
System.out.println("CtrRSACipherGeneratorDemo001 successfully");
} catch (ObsException e) {
    System.out.println("CtrRSACipherGeneratorDemo001 failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code: " + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message: " + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
} catch (Exception e) {
    System.out.println("CtrRSACipherGeneratorDemo001 putObject failed");
    // Print other error details.
    e.printStackTrace();
}
}
```

CTCRCipherGenerator:

```
import com.obs.services.crypto.CTCipherGenerator;
import com.obs.services.crypto.CryptoObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.GetObjectRequest;
import com.obs.services.model.ObsObject;
import com.obs.services.model.PutObjectResult;

import java.io.File;
import java.io.FileOutputStream;
import java.io.InputStream;
import java.security.NoSuchAlgorithmException;
import java.security.SecureRandom;

public class CTCipherGeneratorDemo001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
```

```
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
// Obtain an AK/SK pair and a security token using environment variables or import them in other
ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is located.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");
CTRCipherGenerator ctrCipherGenerator = null;
try {
    SecureRandom secureRandom = SecureRandom.getInstanceStrong();
    // Note that in Linux, using SecureRandom.getInstanceStrong() may block the thread in case of
insufficient system entropy. In this case, you are advised to either use entropy addition methods or set
SecureRandom in another way.
    byte[] exampleMasterKey = new byte[CTRCipherGenerator.CRYPTO_KEY_BYTES_LEN];
    secureRandom.nextBytes(exampleMasterKey);
    // Replace exampleMasterKey with the one you actually use. Make sure the key is 32 bytes and
keep it properly.
    ctrCipherGenerator =
        new CTRCipherGenerator("example_master_key_info", exampleMasterKey, true,
secureRandom);
} catch (IllegalArgumentException | NoSuchAlgorithmException e) {
    e.printStackTrace();
}
assert ctrCipherGenerator != null;
// Create an ObsClient instance.
try (CryptoObsClient cryptoObsClient =
    new CryptoObsClient(ak, sk, securityToken, endPoint, ctrCipherGenerator)) {
    String exampleBucketName = "example-bucket";
    String exampleObjectKey = "exampleObjectKey";
    String examplePlainTextFilePath = "examplePlainTextFilePath";
    String exampleDecryptedFilePath = "exampleDecryptedFilePath";
    PutObjectResult putObjectResult =
        cryptoObsClient.putObject(exampleBucketName, exampleObjectKey, new
File(examplePlainTextFilePath));
    System.out.println("HTTP Code: " + putObjectResult.getStatusCode());
    System.out.println("Etag: " + putObjectResult.getEtag());
    // The object is successfully encrypted on the client and then uploaded.
    System.out.println("CTRCipherGeneratorDemo001 putObject successfully");

    GetObjectRequest getObjectRequest = new GetObjectRequest(exampleBucketName,
exampleObjectKey);
    ObsObject obsObject = cryptoObsClient.getObject(getObjectRequest);
    InputStream input = obsObject.getObjectContent();
    byte[] b = new byte[1024];
    FileOutputStream fileOutputStream = new FileOutputStream(exampleDecryptedFilePath);
    int len;
    while ((len = input.read(b)) != -1) {
        fileOutputStream.write(b, 0, len);
    }
    fileOutputStream.close();
    input.close();

    System.out.println("HTTP Code: " + obsObject.getMetadata().getStatusCode());
    // The object is successfully decrypted on the client and then downloaded.
    System.out.println("CTRCipherGeneratorDemo001 getObject successfully");

    // Check whether the file remains the same before encryption and after decryption.
    byte[] plainTextFileSha256 = CTRCipherGenerator.getFileSha256Bytes(examplePlainTextFilePath);
    byte[] decryptedFileSha256 = CTRCipherGenerator.getFileSha256Bytes(exampleDecryptedFilePath);
    String plainTextFileSha256Base64Encoded =
CTRCipherGenerator.getBase64Info(plainTextFileSha256);
    String decryptedFileSha256Base64Encoded =
CTRCipherGenerator.getBase64Info(decryptedFileSha256);
    System.out.println("plainTextFileSha256 base64 encoded: " + plainTextFileSha256Base64Encoded);
    System.out.println("decryptedFileSha256 base64 encoded: " + decryptedFileSha256Base64Encoded);
```

```
        System.out.println(
            "plainTextFileSha256 equals decryptedFileSha256 ? "
            + decryptedFileSha256Base64Encoded.equals(plainTextFileSha256Base64Encoded));
        System.out.println("CTRCipherGeneratorDemo001 successfully");
    } catch (ObsException e) {
        System.out.println("CTRCipherGeneratorDemo001 failed");
        // Request failed. Print the HTTP status code.
        System.out.println("HTTP Code: " + e.getResponseCode());
        // Request failed. Print the server-side error code.
        System.out.println("Error Code:" + e.getErrorCode());
        // Request failed. Print the error details.
        System.out.println("Error Message: " + e.getErrorMessage());
        // Request failed. Print the request ID.
        System.out.println("Request ID:" + e.getErrorRequestId());
        System.out.println("Host ID:" + e.getErrorHostId());
    } catch (Exception e) {
        System.out.println("CTRCipherGeneratorDemo001 putObject failed");
        // Print other error details.
        e.printStackTrace();
    }
}
```

18 Fault Locating (SDK for Java)

18.1 Methods (SDK for Java)

If problems occur when using the OBS Java SDK, you can perform the following steps to analyze and locate the problems.

- Step 1** Make sure that the latest version of OBS Java SDK is used. You can submit a service ticket to obtain the latest version of OBS Java SDK and download it.
- Step 2** Make sure that the logging function of OBS Java SDK is enabled. For details about how to enable the function, see the [Log Analysis](#) section. The recommended log level is WARN.
- Step 3** Make sure that the program code of the OBS Java SDK complies with [Using an OBS Client](#). All call exceptions of ObsClient APIs are processed as required. The following is a code example of uploading an object:

```
ObsClient obsClient = null;
try
{
    String endPoint = "https://your-endpoint";
    // Hard-coded or plaintext AK/SK are risky. For security purposes, encrypt your AK/SK and store them in
    the configuration file or environment variables. In this example, the AK/SK are stored in environment
    variables for identity authentication. Before running this example, configure environment variables
    ACCESS_KEY_ID and SECRET_ACCESS_KEY_ID.
    // Obtain an AK/SK pair on the management console.
    String ak = System.getenv("ACCESS_KEY_ID");
    String sk = System.getenv("SECRET_ACCESS_KEY_ID");

    obsClient = new ObsClient(ak, sk, endPoint);
    HeaderResponse response = obsClient.putObject("bucketname", "objectname", new
    ByteArrayInputStream("Hello OBS".getBytes()));
    // (Optional) If a call is successful, record the HTTP status code and request ID returned by the server.
    System.out.println(response.getStatusCode());
    System.out.println(response.getRequestId());
}
catch (ObsException e)
{
    // Recommended: When an exception occurs, record the HTTP status code, server-side error code, and
    request ID returned by the server.
    System.out.println("HTTP Code: " + e.getResponseCode());
    System.out.println("Error Code:" + e.getErrorCode());
    System.out.println("Request ID:" + e.getErrorRequestId());
    // Recommended: When an exception occurs, record the stack information.
    e.printStackTrace(System.out);
}
```

 NOTE

You can click [here](#) to view the details about **ObsException**.

- Step 4** If an exception occurs when an ObsClient API is called, obtain the **HTTP status code** and **OBS server-side error code** from **ObsException** or log file, and compare them to locate the exception cause.
- Step 5** If the exception cause cannot be found in step 4, obtain the request ID returned by the OBS server from **ObsException** or log file and contact the OBS server O&M team to locate the cause.
- Step 6** If the request ID is unable to be obtained, collect the stack information of **ObsException** and contact the OBS client O&M team to locate the cause.

----End

18.2 Resolving Dependency Missing or Conflicts (SDK for Java)

Dependency missing and dependency conflict are commonly encountered in Java development or during SDK integration. If error message "ClassNotFoundException" or "NoClassDefFoundError" is reported during application compilation and running, check whether this is a dependency-triggered problem. If yes, perform the following steps to locate and rectify the fault.

Dependency Missing

The following table lists the third-party components (including their versions) that the latest SDK depends on.

Dependency	Version	Description
okhttp	4.12.0	Used to send HTTP requests.
okio	3.8.0	Component of OkHttp
jackson-core	2.15.4	Used to construct and parse JSON files.
jackson-databind	2.15.4	Component of jackson-core
jackson-annotations	2.15.4	Component of jackson-core

Dependency Conflict

If your project has multiple versions of OBS Java SDK packages or third-party dependencies, dependency conflicts may occur. If there are SDKs of earlier versions, delete them and use the latest version. If there are multiple versions of third-party dependencies, replace the conflict ones with the versions required by the SDK.

18.3 Other Notable Issues (SDK for Java)

SignatureDoesNotMatch

HTTP Code: 403
Error Code: SignatureDoesNotMatch

Possible causes are as follows:

1. The SK input into ObsClient initialization is incorrect. Solution: Make sure that the SK is correct.
2. This problem is caused by a bug in the OBS Java SDK of an earlier version. Solution: Upgrade the SDK to the latest version.
3. OBS Java SDK 2.1.x versions are incompatible with the dependent library Apache HttpClient. Solution: Use httpcore-4.4.4 and httpclient-4.5.3.

MethodNotAllowed

HTTP Code: 405
Error Code: MethodNotAllowed

This error occurs because a feature on which the ObsClient API depends has not been rolled out on the requested OBS server. Contact the OBS O&M team for further confirmation.

BucketAlreadyOwnedByYou

HTTP Code: 409
Error Code: BucketAlreadyOwnedByYou

In OBS, a bucket name must be globally unique. Solution: If this error occurs when the **ObsClient.createBucket** is called, check whether the bucket exists. You can use either of the following methods to check whether a bucket exists:

Method 1 (recommended): Call **ObsClient.listBuckets** to query the list of all buckets that you own and check whether the bucket exists.

Method 2: Call **ObsClient.headBucket** to check whether the bucket exists.

NOTE

ObsClient.headBucket can query only buckets in the current region, while **ObsClient.listBuckets** can query buckets in all regions.

BucketAlreadyExists

HTTP Code: 409
Error Code: BucketAlreadyExists

In OBS, a bucket name must be globally unique. Solution: If this error occurs when **ObsClient.createBucket** is called, it indicates that the bucket has been created by another user. Use a different bucket name and try again.

Connection Timeout

HTTP Code: 408
Caused by: java.net.ConnectException: Connection timed out: connect

```
at java.net.DualStackPlainSocketImpl.waitForConnect(Native Method)
at java.net.DualStackPlainSocketImpl.socketConnect(DualStackPlainSocketImpl.java:85)
```

Possible causes are as follows:

1. The endpoint input during ObsClient initialization is incorrect. Solution: Make sure that the endpoint is correct.
2. The network between the OBS client and OBS server is abnormal. Solution: Check the health status of the network.
3. The OBS domain name resolved by DNS is inaccessible. Solution: Contact the OBS O&M team.

Read/Write Timeout

```
HTTP Code: 408
Error Code:RequestTimeOut
Caused by: java.net.SocketTimeoutException: timeout
at okio.Okio$4.newTimeoutException(Okio.java:232)
at okio.AsyncTimeout.exit(AsyncTimeout.java:285)
at okio.AsyncTimeout$2.read(AsyncTimeout.java:241)
```

Possible causes are as follows:

1. The network latency between the OBS client and OBS server is too long. Solution: Check the health status of the network.
2. The network between the OBS client and OBS server is abnormal. Solution: Check the health status of the network.

Abnormal Returned Value -1

```
HTTP Code: -1
```

Possible causes are as follows:

1. The OBS Java SDK of an earlier version is used and a connection timeout or read/write timeout occurs. Solution: See the solutions for [connection timeout](#) and [read/write timeout](#).
2. This is a bug in earlier versions of OBS Java SDK. Solution: Submit a service ticket to obtain the latest version of Java SDK and download it.
3. The server returns an exception. Solution: Obtain the request ID returned by OBS server from the log and contact the OBS O&M team.

Unable to Obtain Error Codes from ObsException

Possible causes are as follows:

1. An error is reported when **ObsClient.getBucketMetadata** or **ObsClient.getObjectMetadata** is called. In this scenario, the server does not return an error code because the request method used in the background is HEAD. Solution: Call **ObsException.getResponseCode** to obtain the HTTP status code to analyze the possible cause. For example, 403 indicates that the user does not have the access permission, and 404 indicates that the bucket or object does not exist. If the cause cannot be located, obtain the request ID returned by the OBS server from **ObsException** and contact the OBS O&M team.
2. The endpoint passed during ObsClient initialization cannot correspond to a valid IP address of the OBS server after DNS resolution. Solution: Check

whether the endpoint is correct. If the endpoint is correct, contact the OBS O&M team.

UnknownHostException

```
Caused by: java.net.UnknownHostException: bucketname.unknowndomain.com
    at java.net.Inet6AddressImpl.lookupAllHostAddr(Native Method)
    at java.net.InetAddress$1.lookupAllHostAddr(InetAddress.java:901)
    at java.net.InetAddress.getAddressesFromNameService(InetAddress.java:1293)
```

Possible causes are as follows:

1. The endpoint input during ObsClient initialization is incorrect. Solution: Make sure that the endpoint is correct.
2. DNS cannot resolve the OBS domain name. Solution: Contact the OBS O&M team.

NullPointerException

```
Exception in thread "main" java.lang.NullPointerException
    at com.obs.services.internal.RestStorageService.isCname(RestStorageService.java:1213)
    at com.obs.services.ObsClient.doActionWithResult(ObsClient.java:2805)
```

Possible causes are as follows:

1. **ObsClient.close** is called to close ObsClient and then another ObsClient API is called. Solution: Call **ObsClient.close** to release resources only before exiting the application.
2. This is a bug in earlier versions of OBS Java SDK. Solution: Submit a service ticket to obtain the latest version of Java SDK and download it.

Connection Leakage

A connection to xxx was leaked. Did you forget to close a response body?

This error occurs when **ObsClient.getObject** is not properly closed after it is called to obtain the data stream of the object to be downloaded. To fix this error, call the **ObsObject.getObjectContent.close** method in the **finally** statement block to close the connection.

Problem in SDK Version Upgrade

The third-party dependency of the SDK of an earlier version (2.1.x) is not completely compatible with that of the new SDK (3.x). If a program startup error occurs after the earlier version is upgraded to the new version, refer to [Resolving Dependency Missing or Conflicts \(SDK for Java\)](#) to rectify the fault. If the fault persists, contact the OBS O&M team.

OkHttp Error After an SDK Upgrade

```
Exception in thread "main" java.lang.NoSuchMethodError: 'okhttp3.RequestBody
okhttp3.RequestBody.create(java.lang.String, okhttp3.MediaType)'
    at com.obs.services.internal.RestConnectionService.createRequestBuilder(RestConnectionService.java:157)
    at com.obs.services.internal.RestConnectionService.setupConnection(RestConnectionService.java:148)
    at com.obs.services.internal.RestConnectionService.setupConnection(RestConnectionService.java:124)
    at com.obs.services.internal.RestStorageService.performRequest(RestStorageService.java:395)
    at com.obs.services.internal.RestStorageService.performRequest(RestStorageService.java:388)
```

This error occurs because OkHttp of an earlier version was used after the SDK upgrade. To resolve this issue, upgrade OkHttp to a required version by referring to [Resolving Dependency Missing or Conflicts \(SDK for Java\)](#).

StackOverflowError After an SDK Upgrade

```
Caused by: java.lang.StackOverflowError
  at sun.misc.URLClassPath.getResource(URLClassPath.java:211) ~[?:1.8.0_91]
  at java.net.URLClassLoader$1.run(URLClassLoader.java:365) ~[?:1.8.0_91]
  at java.net.URLClassLoader$1.run(URLClassLoader.java:362) ~[?:1.8.0_91]
  at java.security.AccessController.doPrivileged(Native Method) ~[?:1.8.0_91]
  at java.net.URLClassLoader.findClass(URLClassLoader.java:361) ~[?:1.8.0_91]
  at java.lang.ClassLoader.loadClass(ClassLoader.java:424) ~[?:1.8.0_91]
  at java.lang.ClassLoader.loadClass(ClassLoader.java:357) ~[?:1.8.0_91]
  at org.apache.catalina.loader.WebappClassLoaderBase.loadClass(WebappClassLoaderBase.java:1806)
```

Check the JVM parameter **xss** and set it to **1 MB**. **xss** indicates the memory size allocated to each thread started by the JVM. By default, the value of **xss** is 256 KB for JDK 1.4 and 1 MB for JDK 1.5 or later.

SSL peer shut down incorrectly

```
javax.net.ssl.SSLException: SSL peer shut down incorrectly
  at sun.security.ssl.InputRecord.readV3Record(InputRecord.java:596)
  at sun.security.ssl.InputRecord.read(InputRecord.java:532)
  at sun.security.ssl.SSLSocketImpl.readRecord(SSLSocketImpl.java:975)
  at sun.security.ssl.SSLSocketImpl.readDataRecord(SSLSocketImpl.java:933)
  at sun.security.ssl.AppInputStream.read(AppInputStream.java:105)
  at obs.shaded.okio.Okio$2.read(Okio.java:140)
  at obs.shaded.okio.AsyncTimeout$2.read(AsyncTimeout.java:237)
  at obs.shaded.okio.RealBufferedSource.read(RealBufferedSource.java:51)
  at obs.shaded.okhttp3.internal.http1.Http1ExchangeCodec
$AbstractSource.read(Http1ExchangeCodec.java:389)
  at obs.shaded.okhttp3.internal.http1.Http1ExchangeCodec
$FixedLengthSource.read(Http1ExchangeCodec.java:427)
  at obs.shaded.okhttp3.internal.connection.Exchange$ResponseBodySource.read(Exchange.java:286)
  at obs.shaded.okio.RealBufferedSource$1.read(RealBufferedSource.java:447)
  at java.util.zip.InflaterInputStream.fill(InflaterInputStream.java:238)
  at java.util.zip.InflaterInputStream.read(InflaterInputStream.java:158)
  at java.util.zip.GZIPInputStream.read(GZIPInputStream.java:117)
  at sun.nio.cs.StreamDecoder.readBytes(StreamDecoder.java:284)
  at sun.nio.cs.StreamDecoder.implRead(StreamDecoder.java:326)
  at sun.nio.cs.StreamDecoder.read(StreamDecoder.java:178)
  at java.io.InputStreamReader.read(InputStreamReader.java:184)
  at java.io.BufferedReader.fill(BufferedReader.java:161)
  at java.io.BufferedReader.readLine(BufferedReader.java:324)
  at java.io.BufferedReader.readLine(BufferedReader.java:389)
```

Do not read file streams by line during a download on the client. For details, see the demo for [a streaming download](#).

Others

For details, see [FAQs](#).

19 Troubleshooting (SDK for Java)

19.1 HTTP Status Codes (SDK for Java)

The OBS server complies with the HTTP standard. After an API is called, the OBS server returns a standard HTTP status code. The following tables list the categories of HTTP status codes and the common HTTP status codes in OBS.

- Categories of HTTP status codes

Category	Description
1XX	Informational response. A request is received by the server and the server requires the requester to continue the operation. This category is usually invisible to the client.
2XX	Success. The operation is received and processed successfully.
3XX	Redirection. Further operations to complete the request are required. This category is usually invisible to the client.
4XX	Client errors. The request contains a syntax error, or the request cannot be implemented.
5XX	Server errors. There is an error when the server is processing a request.

- Common HTTP status codes in OBS and their meanings

HTTP Status Code	Description	Possible Cause
400 Bad Request	The request parameter is incorrect.	- Invalid request parameter. - The consistency check fails after the client request carries MD5. - An invalid parameter is transferred when the SDK is used. - An invalid bucket name is used.
403 Forbidden	The access is denied.	- The signature carried in the request header does not match with the signature calculated by the OBS server. Generally, the error is caused by incorrect AK/SK. - The account does not have the permission to access the requested resource. - The account is in arrears. - The bucket space is insufficient when a quota is set for the bucket. - Invalid AK - The time difference between the client and the server is too large (the time on the machine where the client is located is not in sync with the time on the NTP server).
404 Not Found	The requested resource does not exist.	- The bucket does not exist. - The object does not exist. - The bucket policy configuration does not exist. For example, the bucket CORS configuration or bucket policy configuration does not exist. - The multipart upload does not exist.
405 Method Not Allowed	The request method is not supported.	The request method or feature is not supported in the region where the bucket is located.

HTTP Status Code	Description	Possible Cause
408 Request Timeout	Request timed out.	The Socket connection between the server and client times out.
409 Conflict	Request conflicts occur.	• Buckets of the same name are created in different regions. • The bucket to delete is not empty.
500 Internal Server Error	An internal error occurs on the server side.	An internal error occurs on the server side.
503 Service Unavailable	The service is unavailable.	The server is inaccessible temporarily.

19.2 OBS Server-side Error Codes (SDK for Java)

If the OBS server encounters an error when processing a request, a response containing the error code and error description is returned. The following table lists details about each error code and HTTP status code.

HTTP Status Code	Error Code	Error Message	Solution
301 Moved Permanently	PermanentRedirect	The requested bucket can be accessed only through the specified address. Send subsequent requests to the address.	Send the request to the returned redirection address.
301 Moved Permanently	WebsiteRedirect	The website request lacks bucketName .	Put the bucket name in the request and try again.
307 Moved Temporarily	TemporaryRedirect	Temporary redirection. If the DNS is updated, the request is redirected to the bucket.	The system automatically redirects the request or sends the request to the redirection address.

HTTP Status Code	Error Code	Error Message	Solution
400 Bad Request	BadDigest	The specified value of Content-MD5 does not match the value received by OBS.	Check whether the MD5 value carried in the header is the same as that calculated by the message body.
400 Bad Request	BadDomainName	Invalid domain name.	Use a valid domain name.
400 Bad Request	BadRequest	Invalid request parameter.	Modify the parameter according to the error details returned in the message body.
400 Bad Request	CustomDomainAlreadyExist	The configured domain already exists.	It has been configured and does not need to be configured again.
400 Bad Request	CustomDomainNotExist	The domain to be deleted does not exist.	The domain is not configured or has been deleted. You do not need to delete it.
400 Bad Request	EntityTooLarge	The size of the object uploaded using the POST method exceeds the upper limit.	Modify the conditions specified in the policy when posting the object or reduce the object size.
400 Bad Request	EntityTooSmall	The size of the object uploaded using the POST method does not reach the lower limit.	Modify the conditions specified in the policy when posting the object or increase the object size.

HTTP Status Code	Error Code	Error Message	Solution
400 Bad Request	IllegalLocation-ConstraintException	A request without Location is sent for creating a bucket in a non-default region.	Send the bucket creation request to the default region, or send the request with the Location of the non-default region.
400 Bad Request	IncompleteBody	No complete request body is received due to network or other problems.	Upload the object again.
400 Bad Request	IncorrectNumberOfFilesInPostRequest	Each POST request must contain one file to be uploaded.	Carry a file to be uploaded.
400 Bad Request	InvalidArgument	Invalid parameter.	Modify the parameter according to the error details in the message body.
400 Bad Request	InvalidBucket	The bucket to be accessed does not exist.	Try another bucket name.
400 Bad Request	InvalidBucketName	The bucket name specified in the request is invalid, which may have exceeded the maximum length, or contain special characters that are not allowed.	Try another bucket name.
400 Bad Request	InvalidLocation-Constraint	The specified Location in the bucket creation request is invalid or does not exist.	Correct the Location in the bucket creation request.

HTTP Status Code	Error Code	Error Message	Solution
400 Bad Request	InvalidPart	One or more specified parts are not found. The parts may not be uploaded or the specified entity tags (ETags) do not match the parts' ETags.	Specify the correct parts and entity tags.
400 Bad Request	InvalidPartOrder	Parts are not listed in ascending order by part number.	Sort the parts in ascending order and assemble them again.
400 Bad Request	InvalidPolicyDocument	The content of the form does not meet the conditions specified in the policy document.	Modify the policy in the constructed form according to the error details in the message body and try again.
400 Bad Request	InvalidRedirectLocation	Invalid redirect location.	Specify the correct IP address.
400 Bad Request	InvalidRequest	Invalid request.	Modify the parameter according to the error details returned in the message body.
400 Bad Request	InvalidRequestBody	The request body is invalid. The request requires a message body but no message body is uploaded.	Upload the message body in the correct format.
400 Bad Request	InvalidTargetBucketForLogging	The delivery group has no ACL permission for the target bucket.	Configure the target bucket ACL and try again.
400 Bad Request	KeyTooLongError	The provided key is too long.	Use a shorter key.

HTTP Status Code	Error Code	Error Message	Solution
400 Bad Request	MalformedACLError	The XML file you provided was not well-formed or did not validate against our format requirements.	Use the correct XML format to retry.
400 Bad Request	MalformedError	The XML format in the request is incorrect.	Use the correct XML format to retry.
400 Bad Request	MalformedLoggingStatus	The XML format of Logging is incorrect.	Use the correct XML format to retry.
400 Bad Request	MalformedPolicy	The bucket policy failed the check.	Modify the bucket policy according to the error details returned in the message body.
400 Bad Request	MalformedQuotaError	The Quota XML format is incorrect.	Use the correct XML format to retry.
400 Bad Request	MalformedXML	An XML file of a configuration item is in incorrect format.	Use the correct XML format to retry.
400 Bad Request	MaxMessageLengthExceeded	Copying an object does not require a message body in the request.	Remove the message body and retry.
400 Bad Request	MetadataTooLarge	The size of the metadata header has exceeded the upper limit.	Reduce the size of the metadata header.
400 Bad Request	MissingRegion	No region contained in the request and no default region defined in the system.	Carry the region information in the request.
400 Bad Request	MissingRequestBodyError	An empty XML file is sent as a request.	Provide the correct XML file.

HTTP Status Code	Error Code	Error Message	Solution
400 Bad Request	MissingRequired-Header	A required header is missing in the request.	Provide the required header.
400 Bad Request	MissingSecurity-Header	A required header is missing in the request.	Provide the required header.
400 Bad Request	TooManyBuckets	You have attempted to create more buckets than allowed.	Delete some buckets and try again.
400 Bad Request	TooManyCustomDomains	Too many user accounts are configured.	Delete some user accounts and try again.
400 Bad Request	TooManyWrongSignatures	The request is rejected due to high-frequency errors.	Replace AK and try again.
400 Bad Request	UnexpectedContent	The request requires a message body which is not carried by the client, or the request does not require a message body but the client carries the message body.	Try again according to the instruction.
400 Bad Request	AZRedundancyTypeNotSupported	The current region does not support buckets with this AZ redundancy.	Choose a type of AZ redundancy supported in the region.
400 Bad Request	UserKeyMustBeSpecified	This operation is only available to special users.	Contact the technical support.
403 Forbidden	AccessDenied	Access denied, because the request does not carry a date header or the header format is incorrect.	Provide a correct date header in the request.

HTTP Status Code	Error Code	Error Message	Solution
403 Forbidden	AccessForbidden	Insufficient permission. No CORS rule is configured for the bucket or the CORS rule does not match.	Modify the CORS configuration of the bucket or send the matched OPTIONS request based on the CORS configuration of the bucket.
403 Forbidden	AllAccessDisabled	You have no permission to perform the operation. The bucket name is forbidden.	Change the bucket name.
403 Forbidden	DeregisterUserId	The user has been deregistered.	Top up or re-register.
403 Forbidden	InArrearOrInsufficientBalance	The subscriber owes fees or the account balance is insufficient, and the subscriber does not have the permission to perform an operation.	Top up the account.
403 Forbidden	InsufficientStorageSpace	Insufficient storage space.	If the quota is exceeded, increase quota or delete some objects.
403 Forbidden	InvalidAccessKeyId	The access key ID provided by the customer does not exist in the system.	Provide a correct access key ID.
403 Forbidden	RequestTimeTooSkewed	The request time and the server's time differ a lot.	Check whether the difference between the client time and the current time is too large.

HTTP Status Code	Error Code	Error Message	Solution
403 Forbidden	SignatureDoesNotMatch	The provided signature in the request does not match the signature calculated by OBS.	Check your secret access key and signature calculation method.
403 Forbidden	Unauthorized	You have not been authenticated in real name.	Authenticate your real name and try again.
404 Not Found	NoSuchBucket	The specified bucket does not exist.	Create a bucket and perform the operation again.
404 Not Found	NoSuchBucketPolicy	No bucket policy exists.	Configure a bucket policy.
404 Not Found	NoSuchCORSConfiguration	No CORS configuration exists.	Configure CORS first.
404 Not Found	NoSuchCustomDomain	The requested user domain does not exist.	Set a user domain first.
404 Not Found	NoSuchKey	The specified key does not exist.	Upload the object first.
404 Not Found	NoSuchLifecycleConfiguration	The requested lifecycle rule does not exist.	Configure a lifecycle rule first.
404 Not Found	NoSuchUpload	The specified multipart upload does not exist. The upload ID does not exist or the multipart upload job has been aborted or completed.	Use the existing part or reinitialize the part.
404 Not Found	NoSuchVersion	The specified version ID does not match any existing version.	Use a correct version ID.

HTTP Status Code	Error Code	Error Message	Solution
404 Not Found	NoSuchWebsiteConfiguration	The requested website does not exist.	Configure the website first.
405 Method Not Allowed	MethodNotAllowed	The specified method is not allowed against the requested resource. The message "Specified method is not supported." is returned.	The method is not allowed.
408 Request Timeout	RequestTimeout	No read or write operation has been performed within the timeout period of the socket connection between the user and the server.	Check the network and try again, or contact technical support.
409 Conflict	BucketAlreadyExists	The requested bucket name already exists. The bucket namespace is shared by all users of OBS. Select another name and retry.	Try another bucket name.
409 Conflict	BucketAlreadyOwnedByYou	Your previous request for creating the named bucket succeeded and you already own it.	You do not need to create the bucket again.
409 Conflict	BucketNotEmpty	The bucket that you tried to delete is not empty.	Delete the objects in the bucket and then delete the bucket.

HTTP Status Code	Error Code	Error Message	Solution
409 Conflict	FsObjectConflict	File upload or creation failed.	Check the file creation rule. For example, check whether you are overwriting files when overwrite is not allowed, or whether you are uploading a file to a file (taking this file as a directory) under POSIX semantics.
409 Conflict	OperationAborted	A conflicting operation is being performed on this resource. Retry later.	Try again later.
409 Conflict	ServiceNotSupported	The request method is not supported by the server.	Not supported by the server. Contact technical support.
411 Length Required	MissingContentLength	The HTTP header Content-Length is not provided.	Provide the Content-Length header.
412 Precondition Failed	PreconditionFailed	At least one of the specified preconditions is not met.	Modify according to the condition prompt in the returned message body.
416 Client Requested Range Not Satisfiable	InvalidRange	The requested range cannot be obtained.	Retry with the correct range.
500 Internal Server Error	InternalServerError	An internal error occurs. Retry later.	Contact the technical support.
501 Not Implemented	ServiceNotImplemented	The request method is not implemented by the server.	Not supported currently. Contact the technical support.
503 Service Unavailable	ServiceUnavailable	The server is overloaded or has internal errors.	Try again later or contact the technical support.

HTTP Status Code	Error Code	Error Message	Solution
503 Service Unavailable	SlowDown	Too frequent requests. Reduce your request frequency.	Reduce your request frequency.

19.3 SDK Exceptions (SDK for Java)

SDK custom exceptions (**ObsException**), thrown by **ObsClient**, are inherited from class **java.lang.RuntimeException**. Exceptions are usually OBS server errors, including **OBS error codes** and error information. This facilitates users to locate problems and troubleshoot faults.

ObsException contains the following error information:

- **ObsException.getResponseCode()**: HTTP status code
- **ObsException.getErrorCode()**: Error code returned by the OBS server
- **ObsException.getErrorMessage()**: Error description returned by the OBS server
- **ObsException.getErrorRequestId()**: Request ID returned by the OBS server
- **ObsException.getErrorHostId()**: Requested server ID
- **ObsException.getResponseHeaders()**: HTTP response headers
- **ObsException.printStackTrace()**: Prints the detailed stack trace of an exception.

19.4 SDK Common Response Headers (SDK for Java)

After you successfully call an API in an instance of **ObsClient**, an instance of the **HeaderResponse** class (or of its sub-class) will be returned.

It contains information about HTTP/HTTPS response headers.

Sample code for processing public response headers:

```
// Hard-coded or plaintext AK/SK are risky. For security purposes, encrypt your AK/SK and store them in the
// configuration file or environment variables. In this example, the AK/SK are stored in environment variables
// for identity authentication. Before running this example, configure environment variables ACCESS_KEY_ID
// and SECRET_ACCESS_KEY_ID.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// Create an ObsClient instance.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
HeaderResponse response = obsClient.createBucket("bucketname");

// Obtain the request ID from the common response headers.
System.out.println("\t" + response.getRequestId());

obsClient.close();
```

19.5 Log Analysis (SDK for Java)

How To Enable Logging

1. Save the **log4j2.xml** file obtained from the OBS Java SDK package to the **classpath** root directory.
2. Call **Log4j2Configurator.setLogConfig** to specify the save path of **log4j2.xml** directly.

NOTE

You can obtain the default log configuration file **log4j2.xml** from the OBS Java SDK package, and then modify to customize the file.

Log Path

The log path of OBS Java SDK is specified in **log4j2.xml**. Logs are saved in the path represented by system variable **user.dir** of JDK by default. In general, there are three logs files as follows:

File Name	Description
OBS-SDK.interface_north.log	Northbound log file, which saves the logs about the communication between OBS Java SDK and third-party applications of users.
OBS-SDK.interface_south.log	Southbound log file, which saves the logs about the communication between OBS Java SDK and the OBS server.
OBS-SDK.access.log	Run log file of the OBS server.

Log Format

The SDK log format is: *Log time|Thread number|Log level|Log content*. The following are example logs:

```
#Southbound logs
2017-08-21 17:40:07 133|main|INFO |HttpClient cost 157 ms to apply http request
2017-08-21 17:40:07 133|main|INFO |Received expected response code: true
2017-08-21 17:40:07 133|main|INFO |expected code(s): [200, 204].

#Northbound logs
2017-08-21 17:40:06 820|main|INFO |Storage|1|HTTP+XML|ObsClient|2017-08-21 17:40:05|2017-08-21 17:40:06||0|
2017-08-21 17:40:07 136|main|INFO |Storage|1|HTTP+XML|setObjectAcl|2017-08-21 17:40:06|2017-08-21 17:40:07||0|
2017-08-21 17:40:07 137|main|INFO |ObsClient [setObjectAcl] cost 312 ms
```

Log Level

When current logs cannot be used to troubleshoot system faults, you can change the log level to obtain more information. You can obtain the most information in **TRACE** logs and the least information in **ERROR** logs.

Log level description:

- **OFF**: Close level. If this level is set, logging will be disabled.
- **TRACE**: Trace level. If this level is set, all log information will be printed. This level is not recommended.
- **DEBUG**: Debugging level. If this level is set, information about logs of the **INFO** level and above, HTTP/HTTPS request and response headers, and **StringToSign** information calculated by authentication algorithm will be printed.
- **INFO**: Information level. If this level is set, information about logs of the **WARN** level and above, time consumed for each HTTP/HTTPS request, and time consumed for calling the ObsClient API will be printed.
- **WARN**: Warning level. If this level is set, information about logs of the **ERROR** level and above, as well as information about some critical events (for example, the number of retry attempts exceeds the upper limit) will be printed.
- **ERROR**: Error level. If this level is set, only error information will be printed.

How to Set

The following sample code shows how to set different levels for the southbound logs, northbound logs, and OBS server run logs. (For details about log configuration, see configuration file **log4j2.xml**.)

```
<!-- north log -->
<Logger name="com.obs.services.AbstractClient" level="INFO" additivity="false">
  <AppenderRef ref="NorthInterfaceLogAppender" />
</Logger>

<!-- south log -->
<Logger name="com.obs.services.internal.RestStorageService" level="WARN" additivity="false">
  <AppenderRef ref="SouthInterfaceLogAppender" />
</Logger>

<!-- access log -->
<Logger name="com.obs.log.AccessLogger" level="ERROR" additivity="false">
  <AppenderRef ref="AccessLogAppender" />
</Logger>
```

20 FAQs (SDK for Java)

20.1 How Can I Set an Object to Be Accessible to Anonymous Users? (Java SDK)

To do this, perform the following steps:

- Step 1** Set the ACL to public read for the object by referring to [Configuring an Object ACL \(SDK for Java\)](#).
- Step 2** Obtain the URL of the object by referring to [How Do I Obtain an Object URL? \(Java SDK\)](#) and provide it to anonymous users.
- Step 3** An anonymous user can access the object by entering the URL on a browser.

----End

20.2 What Is the Retry Mechanism of SDK? (Java SDK)

SDK uses the **maxErrorRetry** parameter configured in [Creating and Configuring an OBS Client \(SDK for Java\)](#) to retry. The default value for retry times is 3. 0 to 5 is recommended.

If the network connection is abnormal or the server returns the 5XX error when an ObsClient API is called, the SDK performs an exponential backoff retry.

NOTICE

- For **ObsClient.putObject**, when the data source is an InputStream other than FileInputStream, the SDK does not retry when an I/O exception occurs because the data stream cannot be re-read. The upper-layer application needs to retry.
 - When **ObsClient.getObject** is successfully called and **ObsObject** is returned, the SDK does not retry when an I/O exception occurs during data reading from **ObsObject.getObjectContent** because this situation is beyond the scope of the processing logic of the SDK. The upper-layer application needs to retry.
-

20.3 How Do I Obtain the Static Website Access Address of a Bucket? (Java SDK)

After a bucket is configured to work in static website hosting mode, you can use the following method to combine the static website access address of the bucket.

`https://bucket name.static website hosting domain name`

20.4 How Do I Obtain an Object URL? (Java SDK)

If the uploaded object is set to be read by anonymous users, anonymous users can download the object through the object URL directly. Methods to obtain the object URL are as follows:

Method 1: Query by calling the API. After an object is uploaded using the ObsClient API, **PutObjectResult** is returned. You can call **getObjectUrl** to obtain the URL of the uploaded object. The sample code is as follows:

```
// Hard-coded or plaintext AK/SK are risky. For security purposes, encrypt your AK/SK and store them in the
// configuration file or environment variables. In this example, the AK/SK are stored in environment variables
// for identity authentication. Before running this example, configure environment variables ACCESS_KEY_ID
// and SECRET_ACCESS_KEY_ID.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// Create an ObsClient instance.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Call putObject to upload the object and obtain the return result.
PutObjectResult result = obsClient.putObject("bucketname", "objectname", new File("localfile"));
// Read the URL of the uploaded object.
System.out.println("\t" + result.getObjectUrl());
```

Method 2: Assemble the URL in the format of **https://*Bucket name.Domain name/Directory level/Object name***.

NOTE

- If the object resides in the root directory of a bucket, its URL does not contain a directory level.

20.5 How to Improve the Speed of Uploading Large Files over the Public Network? (Java SDK)

If a file exceeds 100 MB, you are advised to upload the file using multipart upload.

Multipart upload refers to splitting an object into multiple parts and uploading them separately. Each part is a contiguous portion of the object's data. You can upload parts in any sequence. A part can be reloaded after an upload failure, without affecting other parts. Uploading multiple parts of an object using multiple threads concurrently can greatly improve the transmission efficiency.

For code examples, see [Multipart Upload \(SDK for Java\)](#).

20.6 How Can I Perform a Multipart Upload? (Java SDK)

In a multipart upload, you can specify a part of the file to be uploaded by performing the following steps:

- Step 1** Initialize an instance of `ObsClient` based on the AK, SK, and endpoint.
- Step 2** Specify the bucket name and object name to initialize `InitiateMultipartUploadRequest`. Call `InitiateMultipartUploadRequest.setMetadata` to specify the metadata of the object to be uploaded. Then, call `ObsClient.initiateMultipartUpload` to initiate a multipart upload task. A globally unique identifier (upload ID) for this task will be returned.
- Step 3** Specify the bucket name and object name to initialize `UploadPartRequest`. Call `UploadPartRequest.setUploadId` to specify the upload ID to which the part to be uploaded corresponds. Call `setPartNumber` to specify the part number of the part. Call `setFile` to specify the large file to which the part belongs. Call `setPartSize` to specify the part size. Then, call `ObsClient.uploadPart` to upload the part. The ETag value of the uploaded part is returned.
- Step 4** After all parts are uploaded, specify the bucket name, object name, `uploadId`, and `partETags` to initialize a `CompleteMultipartUploadRequest` request. Then, call `ObsClient.completeMultipartUpload` to assemble parts.

----End

For details, see [Multipart Upload \(SDK for Java\)](#).

20.7 How Can I Perform a Download in Multipart Mode? (Java SDK)

In a multipart download, you can specify the range of data to be downloaded. The procedure is as follows:

- Step 1** You need to initialize an instance of `ObsClient` by using AK, SK, and endpoint.
- Step 2** Specify the bucket name and object name to initialize `GetObjectRequest`. Call `GetObjectRequest.setRangeStart` and `GetObjectRequest.setRangeEnd` to set the start and end points of the object data to be downloaded.
- Step 3** Call `ObsClient.getObject` to send the `GetObjectRequest` request in step 2 to download the data in multipart mode.

----End

For details, see [Downloading an Object - Range-Based \(SDK for Java\)](#).

20.8 How Do I Confirm that an Upload is Successful If I Upload an Object to Overwrite an Existing Object with the Same Name in a Bucket? (Java SDK)

After the upload is complete, you can call `ObsClient.getObjectMetadata` to obtain the size and last modification time of the newly uploaded object and compare them with those of the overwritten object.

If the sizes are the same and the last modification time of the new object is later than that of the overwritten object, the upload succeeded. Otherwise, the upload failed.

For details about how to call `ObsClient.getObjectMetadata`, see [Obtaining Object Metadata \(SDK for Java\)](#).

20.9 Does the SDK Support Uploading, Downloading, or Copying Objects in a Batch? (Java SDK)

No.

Currently, the SDK does not provide such APIs. You need to encapsulate the service codes for uploading, downloading, or copying objects in a batch by yourself. The procedure is as follows:

- Step 1** List all objects to be uploaded, downloaded, or copied. For details about how to list objects to be downloaded, see [Listing Objects \(SDK for Java\)](#).
- Step 2** Call the API for uploading, downloading, or copying a single object for the listed objects.

----End

Sample code:

```
// Hard-coded or plaintext AK/SK are risky. For security purposes, encrypt your AK/SK and store them in the
// configuration file or environment variables. In this example, the AK/SK are stored in environment variables
// for identity authentication. Before running this example, configure environment variables ACCESS_KEY_ID
// and SECRET_ACCESS_KEY_ID.
// Obtain an AK/SK pair on the management console.
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
final String bucketName = "bucketname";
// Define the prefix of objects in a bucket.
final String objectPre = "object/";
// Folder to be uploaded
final String localDirPath = "localDirPath";
final List<File> list = new ArrayList<>();
// Scan all objects in the folder.
static void listFiles(File file){
    File[] fs = file.listFiles();
    assert fs != null;
    if (fs.length < 1){
        // If an empty folder needs to be uploaded, add it to the list.
        list.add(file);
    }else{
```

```
        for (File f:fs){
            if (f.isDirectory()){
                listFiles(f);
            }
            if (f.isFile()){
                // Add objects to be uploaded to the list.
                list.add(f);
            }
        }
    }
}

// Traverse the folder to be uploaded and obtain all objects to be uploaded.
File file = new File(localDirPath);
listFiles(file);

// Create an instance of ObsClient.
final ObsClient obsClient = new ObsClient(ak, sk, endPoint);

// Initialize the thread pool.
ExecutorService executorService = Executors.newFixedThreadPool(20);

// Concurrently upload parts.
for (File f:list){
    executorService.execute(() -> {
        if (f.isDirectory()){
            // For empty folders, create empty folder objects in the bucket.
            String remoteObjectKey = objectPre + f.getPath().substring(localDirPath.length() + 1) + "/";
            obsClient.putObject(bucketName, remoteObjectKey, new ByteArrayInputStream(new byte[0]));
        }else{
            String remoteObjectKey = objectPre + f.getPath().substring(localDirPath.length() + 1);
            obsClient.putObject(bucketName, remoteObjectKey, new File(f.getPath()));
        }
    });
}

// Wait until the upload is complete.
executorService.shutdown();
while (!executorService.isTerminated())
{
    try
    {
        executorService.awaitTermination(5, TimeUnit.SECONDS);
    }
    catch (InterruptedException e)
    {
        e.printStackTrace();
    }
}

// Close ObsClient.
try {
    obsClient.close();
} catch (IOException e) {
    e.printStackTrace();
}
```

NOTE

You can use multiple threads to concurrently upload, download, and copy data to improve efficiency.

20.10 What Is Content-Type (MIME)? (Java SDK)

Multipurpose Internet Mail Extensions (MIME) type is a standard way of describing a data type. The MIME type is passed in the Content-Type header.

 NOTE

If you do not specify Content-Type when uploading an object, the SDK determines the object type based on the suffix of the specified object name and automatically assigns a value to Content-Type.

Content-Type

Content-Type indicates the type of data to send or receive and determine the form and encoding method browsers will use to display data (mainly custom client files or media files). If no content type is specified, the type will be generated based on the file name extension. If there is no extension, the content type is **application/octet-stream** by default.

Common Content Types

The commonly used content types are given in the table below.

File Name Extension	Content-Type	File Name Extension	Content-Type
.* (binary stream, unknown file type)	application/octet-stream	.7z	application/x-7z-compressed
.001	application/x-001	.301	application/x-301
.323	text/h323	.906	application/x-906
.907	drawing/907	.a11	application/x-a11
.acp	audio/x-mei-aac	.ai	application/postscript
.aif	audio/aiff	.aifc	audio/aiff
.aiff	audio/aiff	.anv	application/x-anv
.asa	text/asa	.asf	video/x-ms-asf
.asp	text/asp	.asx	video/x-ms-asf
.au	audio/basic	.avi	video/avi
.awf	application/vnd.adobe.workflow	.biz	text/xml
.bmp	application/x-bmp	.bot	application/x-bot
.c4t	application/x-c4t	.c90	application/x-c90
.cal	application/x-cals	.cat	application/vnd.ms-pki.seccat
.cdf	application/x-netcdf	.cdr	application/x-cdr

File Name Extension	Content-Type	File Name Extension	Content-Type
.cel	application/x-cel	.cer	application/x-x509-ca-cert
.cg4	application/x-g4	.cgm	application/x-cgm
.cit	application/x-cit	.class	java/*
.cml	text/xml	.cmp	application/x-cmp
.cmx	application/x-cmx	.cot	application/x-cot
.crl	application/pkix-crl	.crt	application/x-x509-ca-cert
.csi	application/x-csi	.css	text/css
.cut	application/x-cut	.dbf	application/x-dbf
.dbm	application/x-dbm	.dbx	application/x-dbx
.dcd	text/xml	.dcx	application/x-dcx
.der	application/x-x509-ca-cert	.dgn	application/x-dgn
.dib	application/x-dib	.dll	application/x-msdownload
.doc	application/msword	.dot	application/msword
.drw	application/x-drw	.dtd	text/xml
.dwf	Model/vnd.dwf	.dwf	application/x-dwf
.dwg	application/x-dwg	.dxb	application/x-dxb
.dxf	application/x-dxf	.edn	application/vnd.adobe.edn
.emf	application/x-emf	.eml	message/rfc822
.ent	text/xml	.epi	application/x-epi
.eps	application/x-ps	.eps	application/postscript
.etd	application/x-ebx	.exe	application/x-msdownload
.fax	image/fax	.fdf	application/vnd.fdf
.fif	application/fractals	.fo	text/xml

File Name Extension	Content-Type	File Name Extension	Content-Type
.frm	application/x-frm	.g4	application/x-g4
.gbr	application/x-gbr	.	application/x-
.gif	image/gif	.gl2	application/x-gl2
.gp4	application/x-gp4	.hgl	application/x-hgl
.hmr	application/x-hmr	.hpg	application/x-hpgl
.hpl	application/x-hpl	.hqx	application/mac-binhex40
.hrf	application/x-hrf	.hta	application/hta
.htc	text/x-component	.htm	text/html
.html	text/html	.htt	text/webviewhtml
.htx	text/html	.icb	application/x-icb
.ico	image/x-icon	.ico	application/x-ico
.iff	application/x-iff	.ig4	application/x-g4
.igs	application/x-igs	.iii	application/x-iphone
.img	application/x-img	.ins	application/x-internet-signup
.isp	application/x-internet-signup	.IVF	video/x-ivf
.java	java/*	.jfif	image/jpeg
.jpe	image/jpeg	.jpe	application/x-jpe
.jpeg	image/jpeg	.jpg	image/jpeg
.jpg	application/x-jpg	.js	application/x-javascript
.jsp	text/html	.la1	audio/x-liquid-file
.lar	application/x-laplayer-reg	.latex	application/x-latex
.lavs	audio/x-liquid-secure	.lbm	application/x-lbm
.lmsff	audio/x-la-lms	.ls	application/x-javascript
.ltr	application/x-ltr	.m1v	video/x-mpeg
.m2v	video/x-mpeg	.m3u	audio/mpegurl

File Name Extension	Content-Type	File Name Extension	Content-Type
.m4e	video/mpeg4	.mac	application/x-mac
.man	application/x-troff-man	.math	text/xml
.mdb	application/msaccess	.mdb	application/x-mdb
.mfp	application/x-shockwave-flash	.mht	message/rfc822
.mhtml	message/rfc822	.mi	application/x-mi
.mid	audio/mid	.midi	audio/mid
.mil	application/x-mil	.mml	text/xml
.mnd	audio/x-musicnet-download	.mns	audio/x-musicnet-stream
.mocha	application/x-javascript	.movie	video/x-sgi-movie
.mp1	audio/mp1	.mp2	audio/mp2
.mp2v	video/mpeg	.mp3	audio/mp3
.mp4	video/mp4	.mpa	video/x-mpg
.mpd	application/vnd.ms-project	.mpe	video/x-mpeg
.mpeg	video/mpg	.mpg	video/mpg
.mpga	audio/rn-mpeg	.mpp	application/vnd.ms-project
.mps	video/x-mpeg	.mpt	application/vnd.ms-project
.mpv	video/mpg	.mpv2	video/mpeg
.mpw	application/vnd.ms-project	.mpx	application/vnd.ms-project
.mtx	text/xml	.mxxp	application/x-mmxp
.net	image/pnetvue	.nrf	application/x-nrf
.nws	message/rfc822	.odc	text/x-ms-odc
.out	application/x-out	.p10	application/pkcs10

File Name Extension	Content-Type	File Name Extension	Content-Type
.p12	application/x-pkcs12	.p7b	application/x-pkcs7-certificates
.p7c	application/pkcs7-mime	.p7m	application/pkcs7-mime
.p7r	application/x-pkcs7-certreqresp	.p7s	application/pkcs7-signature
.pc5	application/x-pc5	.pci	application/x-pci
.pcl	application/x-pcl	.pcx	application/x-pcx
.pdf	application/pdf	.pdf	application/pdf
.pdx	application/vnd.adobe.pdx	.pfx	application/x-pkcs12
.pgl	application/x-pgl	.pic	application/x-pic
.pko	application/vnd.ms-pki.pko	.pl	application/x-perl
.plg	text/html	.pls	audio/scpls
.plt	application/x-plt	.png	image/png
.png	application/x-png	.pot	application/vnd.ms-powerpoint
.ppa	application/vnd.ms-powerpoint	.ppm	application/x-ppm
.pps	application/vnd.ms-powerpoint	.ppt	application/vnd.ms-powerpoint
.ppt	application/x-ppt	.pr	application/x-pr
.prf	application/pics-rules	.prn	application/x-prn
.prt	application/x-prt	.ps	application/x-ps
.ps	application/postscript	.ptn	application/x-ptn
.pwz	application/vnd.ms-powerpoint	.r3t	text/vnd.rn-realtex3d
.ra	audio/vnd.rn-realaudio	.ram	audio/x-pn-realaudio

File Name Extension	Content-Type	File Name Extension	Content-Type
.ras	application/x-ras	.rat	application/rat-file
.rdf	text/xml	.rec	application/vnd.rn-recording
.red	application/x-red	.rgb	application/x-rgb
.rjs	application/vnd.rn-realsystem-rjs	.rjt	application/vnd.rn-realsystem-rjt
.rlc	application/x-rlc	.rle	application/x-rle
.rm	application/vnd.rn-realmedia	.rmf	application/vnd.adobe.rmf
.rmi	audio/mid	.rmj	application/vnd.rn-realsystem-rmj
.rmm	audio/x-pn-realaudio	.rmp	application/vnd.rn-rn_music_package
.rms	application/vnd.rn-realmedia-secure	.rmvb	application/vnd.rn-realmedia-vbr
.rmx	application/vnd.rn-realsystem-rmx	.rnx	application/vnd.rn-realplayer
.rp	image/vnd.rn-realpix	.rpm	audio/x-pn-realaudio-plugin
.rsml	application/vnd.rn-rsml	.rt	text/vnd.rn-realtex
.rtf	application/msword	.rtf	application/x-rtf
.rv	video/vnd.rn-realvideo	.sam	application/x-sam
.sat	application/x-sat	.sdp	application/sdp
.sdw	application/x-sdw	.sit	application/x-stuffit
.slb	application/x-slb	.sld	application/x-sld
.slk	drawing/x-slk	.smi	application/smil
.smil	application/smil	.smk	application/x-smk

File Name Extension	Content-Type	File Name Extension	Content-Type
.snd	audio/basic	.sol	text/plain
.sor	text/plain	.spc	application/x-pkcs7-certificates
.spl	application/futuresplash	.spp	text/xml
.ssm	application/streamingmedia	.sst	application/vnd.ms-pki.certstore
.stl	application/vnd.ms-pki.stl	.stm	text/html
.sty	application/x-sty	.svg	text/xml
.swf	application/x-shockwave-flash	.tdf	application/x-tdf
.tg4	application/x-tg4	.tga	application/x-tga
.tif	image/tiff	.tiff	image/tiff
.ttf	font/otf	.tld	text/xml
.top	drawing/x-top	.torrent	application/x-bittorrent
.tsd	text/xml	.txt	text/plain
.uin	application/x-icq	.uls	text/iuls
.vcf	text/x-vcard	.vda	application/x-vda
.vdx	application/vnd.visio	.vml	text/xml
.vpg	application/x-vpeg005	.vsd	application/vnd.visio
.vsd	application/x-vsd	.vss	application/vnd.visio
.vst	application/vnd.visio	.vst	application/x-vst
.vsw	application/vnd.visio	.vsx	application/vnd.visio
.vtx	application/vnd.visio	.vxml	text/xml
.wav	audio/wav	.wax	audio/x-ms-wax
.wb1	application/x-wb1	.wb2	application/x-wb2

File Name Extension	Content-Type	File Name Extension	Content-Type
.wb3	application/x-wb3	.wbmp	image/ vnd.wap.wbmp
.wiz	application/ msword	.wk3	application/x-wk3
.wk4	application/x-wk4	.wkq	application/x-wkq
.wks	application/x-wks	.wm	video/x-ms-wm
.wma	audio/x-ms-wma	.wmd	application/x-ms- wmd
.wmf	application/x-wmf	.wml	text/vnd.wap.wml
.wmv	video/x-ms-wmv	.wmx	video/x-ms-wmx
.wmz	application/x-ms- wmz	.wp6	application/x-wp6
.wpd	application/x-wpd	.wpg	application/x-wpg
.wpl	application/ vnd.ms-wpl	.wq1	application/x-wq1
.wr1	application/x-wr1	.wri	application/x-wri
.wrk	application/x-wrk	.ws	application/x-ws
.ws2	application/x-ws	.wsc	text/scriptlet
.wsdl	text/xml	.wvx	video/x-ms-wvx
.xdp	application/ vnd.adobe.xdp	.xdr	text/xml
.xfd	application/ vnd.adobe.xfd	.xdf	application/ vnd.adobe.xdf
.xhtml	text/html	.xlsx	application/ vnd.openxmlform ats- officedocument.sp readsheetml.sheet
.xls	application/x-xls	.xlw	application/x-xlw
.xml	text/xml	.xpl	audio/scpls
.xq	text/xml	.xql	text/xml
.xquery	text/xml	.xsd	text/xml
.xsl	text/xml	.xslt	text/xml
.xwd	application/x-xwd	.x_b	application/x-x_b

File Name Extension	Content-Type	File Name Extension	Content-Type
.sis	application/vnd.symbian.install	.sisx	application/vnd.symbian.install
.x_t	application/x-x_t	.ipa	application/vnd.iphone
.apk	application/vnd.android.package-archive	.xap	application/x-silverlight-app
.zip	application/zip	.rar	application/x-rar-compressed

20.11 How Do I Get My Account ID and User ID?

Obtaining the Account ID and User ID

When calling APIs, you may need to specify the account ID (**DomainID**) and user ID (**UserID**) in some requests. You need to obtain them from the console in advance. To obtain them from the console, do as follows:

Step 1 Log in to the management console.

Step 2 Hover the mouse pointer over the username and choose **My Credentials** from the drop-down list.

On the **My Credentials** page, view the account ID and user ID.

----End

20.12 How Do I Limit the Rate During Object Upload? (SDK for Java)

This example limits the upload rate by specifying the **x-obs-traffic-limit** header. The SDK transparently transmits the header for the server to verify.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        // hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        // advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        // ways.
```

```
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is located.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use a permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use a temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Upload a file.
    // localfile indicates the path of the local file to be uploaded, in which the file name must be
specified.
    PutObjectRequest request = new PutObjectRequest();
    request.setBucketName("examplebucket");
    request.setObjectKey("objectkey");
    request.setFile(new File("localfile"));
    HashMap<String, String> userHeaders = new HashMap<>();
    // Specify x-obs-traffic-limit as the rate limit, in bit/s. 819200 (100 KB/s) is used in this example.
    userHeaders.put("x-obs-traffic-limit", "819200");
    request.setUserHeaders(userHeaders);
    PutObjectResult putObjectResult = obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

20.13 How Do I Limit the Rate During Object Download? (SDK for Java)

This example limits the download rate by specifying the **x-obs-traffic-limit** header. The SDK transparently transmits the header for the server to verify.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.ObsObject;
import java.io.ByteArrayOutputStream;
import java.io.InputStream;
public class GetObject001 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
advised not to use hard coding, which may result in information leakage.
```

```
// Obtain an AK/SK pair and a security token using environment variables or import them in other ways.
// String securityToken = System.getenv("SECURITY_TOKEN");
// Enter the endpoint corresponding to the region where the bucket is located.
String endPoint = "https://your-endpoint";
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use a permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk, endPoint);
// Use a temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Download an object.
    GetObjectRequest request = new GetObjectRequest("examplebucket", "objectname");
    HashMap<String, String> userHeaders = new HashMap<>();
    //Specify x-obs-traffic-limit as the rate limit, in bit/s. 819200 (100 KB/s) is used in this example.
    userHeaders.put("x-obs-traffic-limit", "819200");
    request.setUserHeaders(userHeaders);
    ObsObject obsObject = obsClient.getObject(request);
} catch (ObsException e) {
    System.out.println("getObjectContent failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessages());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("getObjectContent failed");
    // Print other error information.
    e.printStackTrace();
}
}
```

20.14 How Do I Configure the referer Header? (SDK for Java)

This example uploads **localfile** to **examplebucket** as **objectkey** and sets the **referer** header to **https://example.com***.

```
import com.obs.services.ObsClient;
import com.obs.services.exception.ObsException;
import com.obs.services.model.PutObjectRequest;
import java.io.File;
public class PutObject004 {
    public static void main(String[] args) {
        // Obtain an AK/SK pair using environment variables or import the AK/SK pair in other ways. Using
        hard coding may result in leakage.
        // Obtain an AK/SK pair on the management console.
        String ak = System.getenv("ACCESS_KEY_ID");
        String sk = System.getenv("SECRET_ACCESS_KEY_ID");
        // (Optional) If you are using a temporary AK/SK pair and a security token to access OBS, you are
        advised not to use hard coding, which may result in information leakage.
        // Obtain an AK/SK pair and a security token using environment variables or import them in other
        ways.
        // String securityToken = System.getenv("SECURITY_TOKEN");
        // Enter the endpoint corresponding to the region where the bucket is located.
        String endPoint = "https://your-endpoint";
```

```
// Obtain an endpoint using environment variables or import it in other ways.
//String endPoint = System.getenv("ENDPOINT");

// Create an ObsClient instance.
// Use a permanent AK/SK pair to initialize the client.
ObsClient obsClient = new ObsClient(ak, sk,endPoint);
// Use a temporary AK/SK pair and security token to initialize the client.
// ObsClient obsClient = new ObsClient(ak, sk, securityToken, endPoint);

try {
    // Upload a file.
    // localfile indicates the path of the local file to be uploaded, in which the file name must be
specified.
    PutObjectRequest request = new PutObjectRequest();
    request.setBucketName("examplebucket");
    request.setObjectKey("objectkey");
    request.setFile(new File("localfile"));
    HashMap<String, String> userHeaders = new HashMap<>();
    // Specify the referer header when uploading an object.
    userHeaders.put("referer", "https://example.com*");
    request.setUserHeaders(userHeaders);
    PutObjectResult putObjectResult = obsClient.putObject(request);
    System.out.println("putObject successfully");
} catch (ObsException e) {
    System.out.println("putObject failed");
    // Request failed. Print the HTTP status code.
    System.out.println("HTTP Code:" + e.getResponseCode());
    // Request failed. Print the server-side error code.
    System.out.println("Error Code:" + e.getErrorCode());
    // Request failed. Print the error details.
    System.out.println("Error Message:" + e.getErrorMessage());
    // Request failed. Print the request ID.
    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
    e.printStackTrace();
} catch (Exception e) {
    System.out.println("putObject failed");
    // Print other error information.
    e.printStackTrace();
}
}
```