对象存储服务

BrowserJS SDK API 参考

文档版本 01

发布日期 2024-09-12

华为技术有限公司

商标声明


和其他华为商标均为华为技术有限公司的商标。
本文档提及的其他所有商标或注册商标，由各自的所有人拥有。

注意

您购买的产品、服务或特性等应受华为公司商业合同和条款的约束，本文档中描述的全部或部分产品、服务或特性可能不在您的购买或使用范围之内。除非合同另有约定，华为公司对本文档内容不做任何明示或暗示的声明或保证。

由于产品版本升级或其他原因，本文档内容会不定期进行更新。除非另有约定，本文档仅作为使用指导，本文档中的所有陈述、信息和建议不构成任何明示或暗示的担保。

目 录

1 简介

本文档提供了对象存储服务OBS（Object Storage Service） BrowserJS SDK所有接口的描述、方法定义及参数说明等内容。

如果您想了解OBS BrowserJS SDK的端到端使用方法（如安装、初始化、开发、常见问题），以及各接口的具体使用场景，各场景的代码样例等更丰富的信息，请参考**《对象存储服务BrowserJS SDK开发指南》**。

2 初始化

2.1 OBS 客户端初始化

功能说明

OBS客户端（ObsClient）是访问OBS服务的BrowserJS客户端，它为调用者提供一系列与OBS服务进行交互的接口，用于管理、操作桶（Bucket）和对象（Object）等OBS服务上的资源。

方法定义

```
1.构造函数形式：ObsClient(parameter)
2.工厂方法形式：ObsClient.factory(parameter)
```

参数描述

字段名	类型	约束	说明
access_key_id	String	可选	访问密钥中的AK。
secret_access_key	String	可选	访问密钥中的SK。
server	String	必选	连接OBS的服务地址。可包含协议类型、域名、端口号。示例：https://your-endpoint:443。 您可以从这里查看OBS当前开通的服务地址。
timeout	Number	可选	HTTP/HTTPS请求的总超时时间（单位：秒）。默认为300秒。如网络状况不佳或者上传文件较大，建议增大timeout的值。
is_cname	Boolean	可选	是否通过自定义域名访问OBS服务。默认为false。
useRawXhr	Boolean	可选	是否使用原生XHR发送Ajax请求。默认为false。

代码样例

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    timeout : 300,
    // 这里以华北-北京四为例，其他地区请按实际情况填写
    server: 'https://obs.cn-north-4.myhuaweicloud.com'
});
```

2.2 日志初始化

功能说明

通过开启SDK日志功能，可将接口调用过程中产生的日志信息记录到日志文件，用于后续的数据分析或问题定位。

方法定义

```
ObsClient.initLog(parameter)
```

参数描述

字段名	类型	约束	说明
level	String	必选	日志级别，支持debug、info、warn、error四个值。

代码样例

```
obsClient.initLog({
      level:'info'
});
```

2.3 请求对象

功能说明

调用OBS客户端的相关接口均需要传入请求对象（Object类型）作为输入。对于桶操作接口，请求对象中固定包含Bucket字段用于指定桶名；对于对象操作接口，请求对象中固定包含Bucket字段和Key字段分别用于指定桶名与对象名。

参数描述

字段名	类型	约束	说明
Bucket	String	必选	指定桶名。
Key	String	对于对象操作接口必选	指定对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
其他参数	请参见“桶相关接口”章节和“对象相关接口”章节的详细描述。		

2.4 SDK 公共结果对象

功能说明

调用OBS客户端的相关接口完成后，如果未发生异常，则均会返回公共结果对象。

参数描述

字段名		类型	说明
CommonMsg		Object	接口调用完成后的公共信息，包含HTTP状态码，操作失败的错误码等。
	Status	Number	HTTP状态码，小于300表明操作成功；反之，表明操作失败。
	Code	String	OBS服务端错误码，当Status小于300时为空。
	Message	String	OBS服务端错误描述，当Status小于300时为空。
	HostId	String	请求的服务端ID，当Status小于300时为空。
	RequestId	String	OBS服务端返回的请求ID。
	Id2	String	OBS服务端返回的请求ID2。
	Indicator	String	OBS服务端返回的详细错误码，当Status小于300时为空。
InterfaceResult		Object	操作成功后的结果数据，当Status大于300时为空。

字段名		类型	说明
	RequestId	String	OBS服务端返回的请求ID。
	Id2	String	OBS服务端返回的请求ID2。
	其他字段	请参见“桶相关接口”章节和“对象相关接口”章节的详细描述。	

2.5 回调函数返回结果

功能说明

OBS客户端可通过回调函数的形式返回结果，回调函数依次包含异常信息和**SDK公共结果对象**两个参数。如果回调函数中异常信息参数不为空，则表明接口调用异常；反之，则表明接口调用完成，此时应从**SDK公共结果对象**中获取HTTP状态码，判断操作是否成功。

代码样例

```
obsClient.putObject({
    Bucket : 'bucketname',
    Key : 'objectkey',
    Body : 'Hello OBS'
}, function(err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            if(result.InterfaceResult){
                console.log('Operation Succeed');
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
            console.log('HostId-->' + result.CommonMsg.HostId);
            console.log('RequestId-->' + result.CommonMsg.RequestId);
        }
    }
});
```

2.6 Promise 对象返回结果

功能说明

OBS客户端可通过Promise对象返回结果，如果通过Promise对象的catch方法没有捕获到异常，则表明接口调用完成，此时应从**SDK公共结果对象**中获取HTTP状态码，判断操作是否成功。

代码样例

```
obsClient.putObject({
    Bucket : 'bucketname',
    Key : 'objectkey',
    Body : 'Hello OBS'
```

```
}).then(function(result) {
    if(result.CommonMsg.Status < 300){
        if(result.InterfaceResult){
            console.log('Operation Succeed');
        }
    }else{
        console.log('Code-->' + result.CommonMsg.Code);
        console.log('Message-->' + result.CommonMsg.Message);
        console.log('HostId-->' + result.CommonMsg.HostId);
        console.log('RequestId-->' + result.CommonMsg.RequestId);
    }
}).catch(function(err) {
    console.error('Error-->' + err);
});
```


3 预定义常量

3.1 权限类型

访问方式	类型	说明
ObsClient.enums.PermissionRead	String	如果有桶的读权限，则可以获取该桶内对象列表、桶内多段任务、桶的元数据、桶的多版本。 如果有对象的读权限，则可以获取该对象内容和元数据。
ObsClient.enums.PermissionWrite	String	如果有桶的写权限，则可以上传、覆盖和删除该桶内任何对象和段。 此权限在对象上不适用。
ObsClient.enums.PermissionReadAcp	String	如果有读ACP的权限，则可以获取对应的桶或对象的权限控制列表（ACL）。 桶或对象的所有者永远拥有读对应桶或对象ACP的权限。
ObsClient.enums.PermissionWriteAcp	String	如果有写ACP的权限，则可以更新对应桶或对象的权限控制列表（ACL）。 桶或对象的所有者永远拥有写对应桶或对象的ACP的权限。 拥有了写ACP的权限，由于可以更改权限控制策略，实际上意味着拥有了完全访问的权限。

访问方式	类型	说明
ObsClient.enums.PermissionFullControl	String	如果有桶的完全控制权限意味着拥有PermissionRead、PermissionWrite、PermissionReadAcp和PermissionWriteAcp的权限。 如果有对象的完全控制权限意味着拥有PermissionRead、PermissionReadAcp和PermissionWriteAcp的权限。

3.2 可被授权的类型

访问方式	类型	说明
ObsClient.enums.GranteeGroup	String	授权给用户组。
ObsClient.enums.GranteeUser	String	授权给单个用户。

3.3 可被授权用户组

访问方式	类型	说明
ObsClient.enums.GroupAllUsers	String	所有用户。
ObsClient.enums.GroupAuthenticatedUsers	String	授权用户，已废弃。
ObsClient.enums.GroupLogDelivery	String	日志投递组，已废弃。

3.4 预定义访问策略

访问方式	类型	说明
ObsClient.enums.AclPrivate	String	私有读写。
ObsClient.enums.AclPublicRead	String	公共读。
ObsClient.enums.AclPublicReadWrite	String	公共读写。
ObsClient.enums.AclPublicReadDelivered	String	桶公共读，桶内对象公共读。

访问方式	类型	说明
ObsClient.enums.AclPublicReadWriteDelivered	String	桶公共读写，桶内对象公共读写。

3.5 存储类型

访问方式	类型	说明
ObsClient.enums.StorageClassStandard	String	标准存储。
ObsClient.enums.StorageClassWarm	String	低频访问存储。
ObsClient.enums.StorageClassCold	String	归档存储。

3.6 恢复选项

访问方式	类型	说明
ObsClient.enums.RestoreTierExpedited	String	快速恢复，恢复耗时1~5分钟。
ObsClient.enums.RestoreTierStandard	String	标准恢复，恢复耗时3~5小时。

3.7 元数据复制策略

访问方式	类型	说明
ObsClient.enums.CopyMetadata	String	复制元数据。
ObsClient.enums.ReplaceMetadata	String	替换元数据。

4 桶相关接口

4.1 使用前需知

由于浏览器跨域问题，调用桶相关接口前，需要首先配置桶的CORS，如下：

- 设置允许的请求来源为：*
- 设置允许的HTTP方法为：PUT, GET, POST, DELETE, HEAD
- 设置允许请求中可携带的头域为：*
- 设置允许响应中可返回的附加头域为：
  - ETag
  - x-obs-request-id
  - x-obs-id-2
  - x-reserved-indicator
  - x-obs-api
  - x-obs-bucket-location
  - x-obs-version
  - x-obs-storage-class
  - x-default-storage-class

说明

ObsClient的桶相关操作接口函数均支持首字母大小写，如ObsClient.getBucketLocation和ObsClient.GetBucketLocation是相同的函数。

4.2 判断桶是否存在

功能说明

判断桶是否存在，如果返回的结果中HTTP状态码为200则表明桶存在，如果返回404则表明桶不存在。

方法定义

```
ObsClient.headBucket
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.headBucket({
    Bucket : 'bucketname'
}, function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('Bucket exists');
        }else if(result.CommonMsg.Status === 404){
            console.log('Bucket does not exist');
        }
    }
});
```

4.3 删除桶

功能说明

删除桶，待删除的桶必须为空（不包含对象、历史版本对象或分段上传碎片）。

方法定义

```
ObsClient.deleteBucket
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。

字段名	类型	约束	说明
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.deleteBucket({
    Bucket : 'bucketname'
}, function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.4 列举桶内对象

功能说明

列举桶内对象，默认返回最大1000个对象。

方法定义

```
ObsClient.listObjects
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

字段名	类型	约束	说明
Prefix	String	可选	限定返回的对象名必须带有Prefix前缀。
Marker	String	可选	列举对象的起始位置，返回的对象列表将是对象名按照字典序排序后该参数以后的所有对象。
MaxKeys	Number	可选	列举对象的最大数目，取值范围为1~1000，当超出范围时，按照默认的1000进行处理。
Delimiter	String	可选	用于对对象名进行分组的字符。对于对象名中包含Delimiter的对象，其对象名（如果请求中指定了Prefix，则此处的对象名需要去掉Prefix）中从首字符至第一个Delimiter之间的字符串将作为一个分组并作为CommonPrefix返回。

返回结果（InterfaceResult）

字段名		类型	说明
RequestId		String	OBS服务端返回的请求ID。
Location		String	桶的区域位置。
Bucket		String	桶名。
Delimiter		String	用于对对象名进行分组的字符，与请求中的该参数对应。
IsTruncated		String	表明本次请求是否返回了全部结果，“true”表示没有返回全部结果；“false”表示已返回了全部结果。
Prefix		String	对象名的前缀，与请求中的该参数对应。
Marker		String	列举对象的起始位置，与请求中的该参数对应。
NextMarker		String	下次列举对象时请求的起始位置。
MaxKeys		String	列举对象的最大数目，与请求中的该参数对应。
Contents		Array	对象列表。
	ETag	String	对象的MD5值（当对象是服务端加密的对象时，etag值不是对象的MD5值）。
	Size	String	对象的字节数。

字段名			类型	说明
	Key		String	对象名。
	LastModified		String	对象最近一次被修改的时间。
	Owner		Object	对象的所有者。
		ID	String	对象所有者的DomainId。
	StorageClass		String	对象的存储类型。
	Type		String	对象是否可被追加上传。
CommonPrefixes			Array	当请求中设置了Delimiter分组字符时，返回按Delimiter分组后的对象名称前缀列表。
	Prefix		String	按Delimiter分组后的对象名称前缀。

代码样例

```
obsClient.listObjects({
    Bucket : 'bucketname',
    Prefix : 'prefix',
    MaxKeys : 100
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            for(var j=0;j<result.InterfaceResult.Contents.length;j++){
                console.log('Contents[' + j +  ']:');
                console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                console.log('LastModified-->' + result.InterfaceResult.Contents[j]['LastModified']);
                console.log('ETag-->' + result.InterfaceResult.Contents[j]['ETag']);
                console.log('Size-->' + result.InterfaceResult.Contents[j]['Size']);
                console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']['ID']);
                console.log('StorageClass-->' + result.InterfaceResult.Contents[j]['StorageClass']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.5 列举桶内多版本对象

功能说明

列举桶内多版本对象，默认返回最大1000个多版本对象。

方法定义

```
ObsClient.listVersions
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Prefix	String	可选	限定返回的对象名必须带有Prefix前缀。
KeyMarker	String	可选	列举多版本对象的起始位置，返回的对象列表将是对象名按照字典序排序后该参数以后的所有对象。
MaxKeys	Number	可选	列举多版本对象的最大数目，取值范围为1~1000，当超出范围时，按照默认的1000进行处理。
Delimiter	String	可选	用于对对象名进行分组的字符。对于对象名中包含Delimiter的对象，其对象名（如果请求中指定了Prefix，则此处的对象名需要去掉Prefix）中从首字符至第一个Delimiter之间的字符串将作为一个分组并作为CommonPrefix返回。
VersionIdMarker	String	可选	与KeyMarker配合使用，返回的对象列表将是对象名和版本号按照字典序排序后该参数以后的所有对象。 如果VersionIdMarker不是KeyMarker的一个版本号，则该参数无效。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Location	String	桶的区域位置。
Bucket	String	桶名。
Delimiter	String	用于对对象名进行分组的字符，与请求中的该参数对应。
Prefix	String	对象名的前缀，与请求中的该参数对应。
IsTruncated	String	表明本次请求是否返回了全部结果， “true”表示没有返回全部结果； “false”表示已返回了全部结果。

字段名			类型	说明
KeyMarker			String	列举多版本对象的起始位置，与请求中的该参数对应。
VersionIdMarker			String	表示列举多版本对象的起始位置（VersionId标识），与请求中的该参数对应。
NextKeyMarker			String	下次列举多版本对象请求的起始位置。
NextVersionIdMarker			String	下次列举多版本对象请求的起始位置（VersionId标识），与NextKeyMarker配合使用。
MaxKeys			String	列举多版本对象的最大数目，与请求中的该参数对应。
Versions			Array	多版本对象列表。
	ETag		String	对象的MD5值。
	Size		String	对象的字节数。
	Key		String	对象名。
	VersionId		String	对象的版本号。
	IsLatest		String	标识对象是否是最新的版本，true代表是最新的版本。
	LastModified		String	对象最近一次被修改的时间。
	Owner		Object	对象的所有者。
		ID	String	对象所有者的DomainId。
	StorageClass		String	对象的存储类型。
	Type		String	对象是否可被追加上传。
DeleteMarkers			Array	删除标记列表。
	Owner		Object	对象的所有者。
		ID	String	对象所有者的DomainId。
	Key		String	对象名。
	VersionId		String	对象的版本号。
	IsLatest		String	标识对象是否是最新的版本，true代表是最新的版本。
	LastModified		String	对象最近一次被修改的时间。
CommonPrefixes			Array	当请求中设置了Delimiter分组字符时，返回按Delimiter分组后的对象名称前缀列表。

字段名		类型	说明
	Prefix	String	按Delimiter分组后的对象名称前缀。

代码样例

```
obsClient.listVersions({
    Bucket : 'bucketname',
    Prefix : 'prefix',
    MaxKeys : 100
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Versions:');
            for(var j=0;j<result.InterfaceResult.Versions;j++){
                console.log('Version[' + j +  ']:');
                console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                console.log('IsLatest-->' + result.InterfaceResult.Versions[j]['IsLatest']);
                console.log('LastModified-->' + result.InterfaceResult.Versions[j]['LastModified']);
                console.log('ETag-->' + result.InterfaceResult.Versions[j]['ETag']);
                console.log('Size-->' + result.InterfaceResult.Versions[j]['Size']);
                console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                console.log('StorageClass-->' + result.InterfaceResult.Versions[j]['StorageClass']);
            }
            console.log('DeleteMarkers:');
            for(var i=0;i<result.InterfaceResult.DeleteMarkers.length;i++){
                console.log('DeleteMarker[' + i +  ']:');
                console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                console.log('IsLatest-->' + result.InterfaceResult.DeleteMarkers[i]['IsLatest']);
                console.log('LastModified-->' + result.InterfaceResult.DeleteMarkers[i]['LastModified']);
                console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.6 列举桶内分段上传任务

功能说明

列举指定桶中所有的初始化后还未合并或还未取消的分段上传任务。

方法定义

```
ObsClient.listMultipartUploads
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。

字段名	类型	约束	说明
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Delimiter	String	可选	用于对分段上传任务中的对象名进行分组的字符。对于对象名中包含Delimiter的任务，其对象名（如果请求中指定了Prefix，则此处的对象名需要去掉Prefix）中从首字符至第一个Delimiter之间的字符串将作为一个分组并作为CommonPrefix返回。
Prefix	String	可选	限定返回的分段上传任务中的对象名必须带有Prefix前缀。
MaxUploads	Number	可选	列举分段上传任务的最大数目，取值范围为1~1000，当超出范围时，按照默认的1000进行处理。
KeyMarker	String	可选	表示列举时返回指定的KeyMarker之后的分段上传任务。
UploadIdMarker	String	可选	只有与KeyMarker参数一起使用时才有意义，用于指定返回结果的起始位置，即列举时返回指定KeyMarker的UploadIdMarker之后的分段上传任务。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Bucket	String	桶名。
KeyMarker	String	列举分段上传任务的起始位置，与请求中的该参数对应。
UploadIdMarker	String	列举分段上传任务的起始位置（UploadId标识），与请求中的该参数对应。
NextKeyMarker	String	下次列举分段上传任务请求的起始位置。
NextUploadIdMarker	String	下次列举分段上传任务请求的起始位置（UploadId标识），与NextKeyMarker配合使用。
Delimiter	String	用于对分段上传任务中的对象名进行分组的字符，与请求中的该参数对应。

字段名			类型	说明
Prefix			String	分段上传任务中的对象名前缀，与请求中的该参数对应。
MaxUploads			String	列举分段上传任务的最大数目，与请求中的该参数对应。
IsTruncated			String	表明本次请求是否返回了全部结果， “true”表示没有返回全部结果； “false”表示已返回了全部结果。
Uploads			Array	分段上传任务列表。
	Key		String	分段上传任务的对象名。
	UploadId		String	分段上传任务的ID。
	Initiator		Object	分段上传任务的创建者。
		ID	String	创建者的DomainId。
	Owner		Object	和Initiator相同，代表分段上传任务的创建者。
		ID	String	创建者的DomainId。
	Initiated		String	分段上传任务的初始化时间。
	StorageClass		String	分段上传对象的存储类型。
CommonPrefixes			Array	当请求中设置了Delimiter分组字符时，返回按Delimiter分组后的对象名称前缀列表。
	Prefix		String	按Delimiter分组后的对象名称前缀。

代码样例

```
obsClient.listMultipartUploads ({
    Bucket : 'bucketname',
    Prefix : 'prefix',
    MaxUploads : 100
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Bucket-->' + result.InterfaceResult.Bucket);
            for(var i=0;i<result.InterfaceResult.Uploads.length;i++){
                console.log('Uploads[' + i + ']');
                console.log('UploadId-->' + result.InterfaceResult.Uploads[i]['UploadId']);
                console.log('Key-->' + result.InterfaceResult.Uploads[i]['Key']);
                console.log('Initiated-->' + result.InterfaceResult.Uploads[i]['Initiated']);
                console.log('StorageClass-->' + result.InterfaceResult.Uploads[i]['StorageClass']);
                console.log('Owner[ID]-->' + result.InterfaceResult.Uploads[i]['Owner']['ID']);
                console.log('Initiator[ID]-->' + result.InterfaceResult.Uploads[i]['Initiator']['ID']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
```

```
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.7 获取桶元数据

功能说明

对桶发送HEAD请求，获取桶的元数据信息。

方法定义

```
ObsClient.getBucketMetadata
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
StorageClass	String	桶的存储类型，当桶存储类型是标准存储时，该值为空。
Location	String	桶区域位置。
ObsVersion	String	OBS服务端版本。

代码样例

```
obsClient.getBucketMetadata({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('StorageClass-->' + result.InterfaceResult.StorageClass);
            console.log('Location-->' + result.InterfaceResult.Location);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
```

```
    }
});
```

4.8 获取桶区域位置

功能说明

获取桶所在的区域位置。

方法定义

```
ObsClient.getBucketLocation
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Location	String	桶所在的区域位置。

代码样例

```
obsClient.getBucketLocation({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Location-->' + result.InterfaceResult.Location);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.9 获取桶存量信息

功能说明

获取桶的存量信息，包含桶的空间大小以及对象个数。

方法定义

```
ObsClient.getBucketStorageInfo
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Size	String	桶的空间大小。
ObjectNumber	String	桶内对象个数。
StandardSize	String	返回标准存储类型存量大小
StandardObjectNumber	String	返回标准存储类型对象个数
WarmSize	String	obs请求时返回低频存储类型存量大小
WarmObjectNumber	String	obs请求时返回低频存储类型对象个数
ColdSize	String	obs请求时返回归档存储类型存量大小
ColdObjectNumber	String	obs请求时返回归档存储类型对象个数
DeepArchiveSize	String	返回深度归档存储类型存量大小
DeepArchiveObjectNumber	String	返回深度归档存储类型对象个数
HighPerformanceSize	String	返回高性能存储类型存量大小
HighPerformanceObjectNumber	String	返回高性能存储类型对象个数

字段名	类型	说明
Standard_IASize	String	s3请求时返回低频存储类型存量大小
Standard_IAObjectNumber	String	s3请求时返回低频存储类型对象个数
GlacierSize	String	s3请求时返回归档存储类型存量大小
GlacierObjectNumber	String	s3请求时返回归档存储类型对象个数

代码样例

```
obsClient.getBucketStorageInfo({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Size-->' + result.InterfaceResult.Size);
            console.log('ObjectNumber-->' + result.InterfaceResult.ObjectNumber);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.10 设置桶配额

功能说明

设置桶的配额值，单位为字节，支持的最大值为2^{63}-1，配额值设为0表示桶的配额没有上限。

方法定义

```
ObsClient.setBucketQuota
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
StorageQuota	Number	必选	桶的配额值，非负整数。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketQuota ({
    Bucket : 'bucketname',
    StorageQuota : 1024 * 1024 * 1024
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.11 获取桶配额

功能说明

获取桶的配额值，0代表配额没有上限。

方法定义

```
ObsClient.getBucketQuota
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
StorageQuota	String	桶的配额值。

代码样例

```
obsClient.getBucketQuota ({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('StorageQuota-->' + result.InterfaceResult.StorageQuota);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.12 设置桶存储类型

功能说明

设置桶的存储类型，桶中对象的存储类型默认将与桶的存储类型保持一致。

方法定义

```
ObsClient.setBucketStoragePolicy
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
StorageClass	String	必选	桶的**存储类型**。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketStoragePolicy({
    Bucket : 'bucketname',
    StorageClass: obsClient.enums.StorageClassWarm
},function (err, result) {
    if(err){
```

```
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.13 获取桶存储类型

功能说明

获取桶的存储类型。

方法定义

```
ObsClient.getBucketStoragePolicy
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
StorageClass	String	桶的存储类型。

代码样例

```
obsClient.getBucketStoragePolicy ({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('StorageClass-->' + result.InterfaceResult.StorageClass);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
```

```
        }
    }
});
```

4.14 设置桶 ACL

功能说明

设置桶的访问权限。

方法定义

```
ObsClient.setBucketAcl
```

请求参数

字段名			类型	约束	说明
Bucket			String	必选	桶名。
RequestDate			String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
ACL			String	可选	**预定义访问策略**。
Owner			Object	可选	桶的所有者。
	ID		String	必选	桶所有者的DomainId。
Grants			Array	可选	被授权用户权限信息列表。
	Grantee		Object	必选	被授权用户。
		Type	String	必选	被授权的**用户类型**。
		ID	String	如果Type为“CanonicalUser”则必选，否则必须为空	被授权用户的DomainId。
		URI	String	如果Type为“Group”则必选，否则必须为空	被授权的**用户组**。

字段名		类型	约束	说明
	Permission	String	必选	被授予的**权限**。
	Delivered	Boolean	可选	桶内对象ACL是否继承桶的ACL。

说明

- Owner和Grants必须配套使用，且与ACL互斥。当设置了这两个字段时，不能设置ACL；反之，当设置了ACL时，不能设置Owner和Grants。
- Owner、Grants与ACL不能全为空。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketAcl ({
	Bucket : 'bucketname',
	Owner:{ID:'ownerid'},
	Grants:[
		{Grantee:{Type:'CanonicalUser',ID:'userid'},Permission:obsClient.enums.PermissionRead},
		{Grantee:{Type:'CanonicalUser',ID:'userid'},Permission:obsClient.enums.PermissionWrite},
		{Grantee:{Type:'Group', URI: obsClient.enums.GroupLogDelivery},Permission:
obsClient.enums.PermissionWrite},
		{Grantee:{Type:'Group', URI: obsClient.enums.GroupLogDelivery},Permission:
obsClient.enums.PermissionWriteAcp}
	]
},function (err, result) {
	if(err){
		console.error('Error-->' + err);
	}else{
		if(result.CommonMsg.Status < 300){
			console.log('RequestId-->' + result.InterfaceResult.RequestId);
		}else{
			console.log('Code-->' + result.CommonMsg.Code);
			console.log('Message-->' + result.CommonMsg.Message);
		}
	}
});
```

4.15 获取桶 ACL

功能说明

获取桶的访问权限。

方法定义

```
ObsClient.getBucketAcl
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名			类型	说明
RequestId			String	OBS服务端返回的请求ID。
Owner			Object	桶的所有者。
	ID		String	桶所有者的DomainId。
	Name		String	桶所有者的名字。
Grants			Array	被授权用户权限信息列表。
	Grantee		Object	被授权用户。
		Name	String	被授权用户的名字，当用户类型是Group时为空。
		ID	String	被授权用户的DomainId，当用户类型是Group时为空。
		URI	String	被授权的用户组，当用户类型是CanonicalUser时为空。
	Permission		String	被授予的权限。
	Delivered		String	桶内对象ACL是否继承桶的ACL。

代码样例

```
obsClient.getBucketAcl ({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Owner[ID]-->' + result.InterfaceResult.Owner.ID);
            console.log('Grants:');
            for(var i=0;i<result.InterfaceResult.Grants.length;i++){
                console.log('Grant[' + i + ']:');
                console.log('Grantee[ID]-->' + result.InterfaceResult.Grants[i]['Grantee']['ID']);
                console.log('Grantee[URI]-->' + result.InterfaceResult.Grants[i]['Grantee']['URI']);
                console.log('Permission-->' + result.InterfaceResult.Grants[i]['Permission']);
            }
```

```
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.16 设置桶日志管理配置

功能说明

设置桶的访问日志配置。

方法定义

```
ObsClient.setBucketLogging
```

请求参数

字段名				类型	约束	说明
Bucket				String	必选	桶名。
RequestDate				String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Agency				String	如果是设置桶日志配置则必选	委托名。
LoggingEnabled				Object	可选	日志配置信息。
	TargetBucket			String	必选	生成日志的目标桶。
	TargetPrefix			String	必选	在目标桶中生成日志对象的对象名前缀。
	TargetGrants			Array	可选	被授权用户权限信息列表。
		Grantee		Object	可选	被授权用户。
			Type	String	必选	被授权的**用户类型**。
			ID	String	如果Type为“CanonicalUser”则必选，否则必须为空	被授权用户的DomainId。

字段名				类型	约束	说明
			URI	String	如果Type为“Group”则必选，否则必须为空	被授权的**用户组**。
		Permission		String	可选	被授予的**权限**。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketLogging ({
    Bucket : 'bucketname',
    LoggingEnabled:{
        TargetBucket:'targetbucketname',
        TargetPrefix:'prefix',
        TargetGrants:[
            {Grantee: {Type:'CanonicalUser',ID:'userid'},Permission: obsClient.enums.PermissionRead},
            {Grantee: {Type:'Group',URI: obsClient.enums.GroupAllUsers},Permission:
obsClient.enums.PermissionRead}
        ]
    }
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.17 获取桶日志管理配置

功能说明

获取桶的访问日志配置。

方法定义

```
ObsClient.getBucketLogging
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名				类型	说明
RequestId				String	OBS服务端返回的请求ID。
LoggingEnabled				Object	日志配置信息。
	TargetBucket			String	生成日志的目标桶。
	TargetPrefix			String	在目标桶中生成日志对象的对象名前缀。
	TargetGrants			Array	被授权用户权限信息列表。
		Grantee		Object	被授权用户。
			ID	String	被授权用户的DomainId，当用户类型是Group时为空。
			URI	String	被授权的用户组，当用户类型是CanonicalUser时为空。
		Permission		String	被授予的权限。

代码样例

```
obsClient.getBucketLogging ({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            if(result.InterfaceResult.LoggingEnabled){
                console.log('TargetBucket-->' + result.InterfaceResult.LoggingEnabled.TargetBucket);
                console.log('TargetPrefix-->' + result.InterfaceResult.LoggingEnabled.TargetPrefix);
                for(var i=0;i<result.InterfaceResult.LoggingEnabled.TargetGrants.length;i++){
                    console.log('Grant[' + i + ']:');
                    console.log('Grantee[ID]-->' + result.InterfaceResult.LoggingEnabled.TargetGrants[i]
['Grantee']['ID']);
                    console.log('Grantee[URI]-->' + result.InterfaceResult.LoggingEnabled.TargetGrants[i]
['Grantee']['URI']);
                    console.log('Permission-->' + result.InterfaceResult.LoggingEnabled.TargetGrants[i]
['Permission']);
                }
```

```
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.18 设置桶策略

功能说明

配置桶的策略，如果桶已经存在一个策略，那么当前请求中的策略将完全覆盖桶中现存的策略。

方法定义

```
ObsClient.setBucketPolicy
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Policy	String	必选	策略信息，JSON格式的字符串。具体格式请参考**Policy格式**。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketPolicy ({
    Bucket : 'bucketname',
    Policy: 'your policy',
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
```

```
    }
});
```

须知

Policy中的Resource字段包含的桶名必须和当前设置桶策略的桶名一致。

4.19 获取桶策略

功能说明

获取桶的策略。

方法定义

```
ObsClient.getBucketPolicy
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Policy	String	策略信息，JSON格式的字符串。

代码样例

```
obsClient.getBucketPolicy ({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Policy-->' + result.InterfaceResult.Policy);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
```

```
    }
});
```

4.20 删除桶策略

功能说明

删除桶的策略。

方法定义

```
ObsClient.deleteBucketPolicy
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.deleteBucketPolicy ({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.21 设置桶的生命周期配置

功能说明

配置桶的生命周期规则，实现定时转换桶中对象的存储类型，以及定时删除桶中对象的功能。

方法定义

```
ObsClient.setBucketLifecycle
```

请求参数

字段名			类型	约束	说明
Bucket			String	必选	桶名。
RequestDate			String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Rules			Array	必选	桶生命周期规则列表。
	Transitions		Array	可选	对象转换策略列表。
		StorageClass	String	必选	对象转换后的**存储类型**。 **说明** 不支持“标准存储”类型。
		Date	String	如果没有设置Days则必选	表示对象转换的日期。该值必须兼容ISO8601格式（例如：2018-01-01T00:00:00Z），而且必须是UTC午夜0点。
		Days	Number	如果没有设置Date则必选	表示在对象创建时间后第几天时转换，正整数。
	Expiration		Object	可选	对象过期时间配置。

字段名			类型	约束	说明
		Date	String	如果没有设置Days则必选	表示对象过期的日期。该值必须兼容ISO8601格式（例如：2018-01-01T00:00:00Z），而且必须是UTC午夜0点。
		Days	Number	如果没有设置Date则必选	表示在对象创建时间后第几天时过期，正整数。
	ID		String	可选	规则ID，由不超过255个字符的字符串组成。
	Prefix		String	必选	对象名前缀，用以标识哪些对象可以匹配到当前这条规则。可为空字符串，代表匹配桶内所有对象。
	Status		String	必选	标识当前这条规则是否启用，支持的值： • Enabled • Disabled
	NoncurrentVersionTransitions		Array	可选	历史版本对象转换策略列表。
		StorageClass	String	必选	历史版本对象转换后的**存储类型**。
		NoncurrentDays	Number	必选	表示对象成为历史版本后第几天时转换，正整数。
	NoncurrentVersionExpiration		Object	可选	历史版本对象过期时间配置。
		NoncurrentDays	Number	必选	表示对象成为历史版本后第几天时过期，正整数。

说明

Transition、Expiration、NoncurrentVersionTransitions、NoncurrentVersionExpiration不能全为空。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketLifecycle({
    Bucket : 'bucketname',
    Rules:[
        {ID:'rule1',Prefix:'prefix1',Status:'Enabled',Expiration:{Days: 60}, NoncurrentVersionExpiration:
{NoncurrentDays : 60}},
        {ID:'rule2',Prefix:'prefix2',Status:'Enabled',Expiration:{Date: '2018-12-31T00:00:00Z'}}
    ]
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.22 获取桶的生命周期配置

功能说明

获取桶的生命周期规则。

方法定义

```
ObsClient.getBucketLifecycle
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名			类型	说明
RequestId			String	OBS服务端返回的请求ID。
Rules			Array	桶生命周期规则列表。
	Transitions		Array	对象转换策略列表。
		StorageClass	String	对象转换后的存储类型。
		Date	String	表示对象转换的日期。
		Days	String	表示在对象创建时间后第几天时转换。
	Expiration		Object	对象过期时间配置。
		Date	String	表示对象过期的日期。
		Days	String	表示在对象创建时间后第几天时过期。
	ID		String	规则ID。
	Prefix		String	对象名前缀，用以标识哪些对象可以匹配到当前这条规则。
	Status		String	标识当前这条规则是否启用。
	NoncurrentVersionTransitions		Array	历史版本对象转换策略列表。
		StorageClass	String	历史版本对象转换后的存储类型。
		NoncurrentDays	String	表示对象成为历史版本后第几天时转换。
	NoncurrentVersionExpiration		Object	历史版本对象过期时间配置。
		NoncurrentDays	String	表示对象成为历史版本后第几天时过期。

代码样例

```
obsClient.getBucketLifecycle({
    Bucket : 'bucketname'
},function (err, result) {
```



```
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            for(var i=0;i<result.InterfaceResult.Rules.length;i++){
                console.log('Rule[' + i + ']:');
                console.log('ID-->' + result.InterfaceResult.Rules[i]['ID']);
                console.log('Prefix-->' + result.InterfaceResult.Rules[i]['Prefix']);
                console.log('Status-->' + result.InterfaceResult.Rules[i]['Status']);
                console.log('Expiration[Date]-->' + result.InterfaceResult.Rules[i]['Expiration']['Date']);
                console.log('Expiration[Days]-->' + result.InterfaceResult.Rules[i]['Expiration']['Days']);
                console.log('NoncurrentVersionExpiration[NoncurrentDays]-->' +
result.InterfaceResult.Rules[i]['NoncurrentVersionExpiration']['NoncurrentDays']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.23 删除桶的生命周期配置

功能说明

删除桶所有的生命周期规则。

方法定义

```
ObsClient.deleteBucketLifecycle
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.deleteBucketLifecycle({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
```

```
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.24 设置桶的 Website 配置

功能说明

设置桶的Website配置。

方法定义

```
ObsClient.setBucketWebsite
```

请求参数

字段名		类型	约束	说明
Bucket		String	必选	桶名。
RequestDate		String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
RedirectAllRequestsTo		Object	可选	所有请求重定向规则。
	HostName	String	必选	重定向请求时使用的站点名。
	Protocol	String	可选	重定向请求时使用的协议，支持的值： ● http（默认） ● https
ErrorDocument		Object	可选	错误页面配置。
	Key	String	可选	指定当4XX错误出现时返回的页面。
IndexDocument		Object	可选	默认页面配置。

字段名			类型	约束	说明
	Suffix		String	必选	该字段被追加在对文件夹的请求的末尾（例如：配置的是“index.html”，请求的是“samplebucket/images/”，返回的数据将是“samplebucket”桶内名为“images/index.html”的对象的内容）。该字段不能为空或者包含“/”字符。
RoutingRules			Array	可选	重定向规则列表。
	Condition		Object	可选	重定向规则的匹配条件。
		HttpErrorCodeReturnedEquals	String	可选	重定向规则生效需要匹配的HTTP错误码。
		KeyPrefixEquals	String	可选	重定向规则生效需要匹配的对象名前缀。
	Redirect		Object	必选	重定向请求时的具体信息。
		Protocol	String	可选	重定向请求时使用的协议，支持的值： • http • https
		HostName	String	可选	重定向请求时使用的站点名。
		ReplaceKeyPrefixWith	String	可选	重定向请求时使用的对象名前缀。
		ReplaceKeyWith	String	可选	重定向请求时使用的对象名。不可与ReplaceKeyPrefixWith同时存在。
		HttpRedirectCode	String	可选	重定向请求时响应中的HTTP状态码。

说明

- ErrorDocument、IndexDocument和RoutingRules必须配套使用，且与RedirectAllRequestsTo互斥。当设置了这三个字段时，不能设置RedirectAllRequestsTo；反之，当设置了RedirectAllRequestsTo时，不能设置ErrorDocument、IndexDocument和RoutingRules。
- 当ErrorDocument、IndexDocument和RoutingRules三个字段一起使用时，RoutingRules可为空。
- ErrorDocument、IndexDocument、RoutingRules与RedirectAllRequestsTo不能全为空。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketWebsite({
        Bucket : 'bucketname',
//      RedirectAllRequestsTo : {HostName : 'www.example.com', Protocol : 'https'}
        IndexDocument:{Suffix:'index.html'},
        ErrorDocument:{Key:'error.html'},
        RoutingRules:[
            {Condition:{HttpErrorCodeReturnedEquals:'404'},Redirect:
{Protocol:'http',ReplaceKeyWith:'NotFound.html'}},
            {Condition:{HttpErrorCodeReturnedEquals:'404'},Redirect:
{Protocol:'https',ReplaceKeyWith:'test.html'}}
        ]
},function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            if(result.CommonMsg.Status < 300){
                console.log('RequestId-->' + result.InterfaceResult.RequestId);
            }else{
                console.log('Code-->' + result.CommonMsg.Code);
                console.log('Message-->' + result.CommonMsg.Message);
            }
        }
});
```

4.25 获取桶的 Website 配置

功能说明

获取桶的Website配置。

方法定义

```
ObsClient.getBucketWebsite
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名			类型	说明
RequestId			String	OBS服务端返回的请求ID。
RedirectAllRequestsTo			Object	所有请求重定向的规则。
	HostName		String	重定向时使用的站点名。
	Protocol		String	重定向时使用的协议。
ErrorDocument			Object	错误页面配置。
	Key		String	指定当4XX错误出现时返回的页面。
IndexDocument			Object	默认页面配置。
	Suffix		String	该字段被追加在对文件夹的请求的末尾（例如：配置的是“index.html”，请求的是“samplebucket/images/”，返回的数据将是“samplebucket”桶内名为“images/index.html”的对象的内容）。该字段不能为空或者包含“/”字符。
RoutingRules			Array	重定向规则列表。
	Condition		Object	重定向规则的匹配条件。
		HttpErrorCodeReturnedEquals	String	重定向规则生效需要匹配的HTTP错误码。
		KeyPrefixEquals	String	重定向规则生效需要匹配的对象名前缀。
	Redirect		Object	重定向请求时的具体信息。
		Protocol	String	重定向请求时使用的协议。
		HostName	String	重定向请求时使用的站点名。
		ReplaceKeyPrefixWith	String	重定向请求时使用的对象名前缀。

字段名			类型	说明
		ReplaceKeyWith	String	重定向请求时使用的对象名。不可与ReplaceKeyPrefixWith同时存在。
		HttpRedirectCode	String	重定向请求时响应中的HTTP状态码。

代码样例

```
obsClient.getBucketWebsite({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('RedirectAllRequestsTo:');
            console.log('HostName-->' + result.InterfaceResult.RedirectAllRequestsTo['HostName']);
            console.log('Protocol-->' + result.InterfaceResult.RedirectAllRequestsTo['Protocol']);
            console.log('IndexDocument[Suffix]-->' + result.InterfaceResult.IndexDocument['Suffix']);
            console.log('ErrorDocument[Key]-->' + result.InterfaceResult.ErrorDocument['Key']);
            console.log('RoutingRules:');
            for(var i=0;i<result.InterfaceResult.RoutingRules.length;i++){
                console.log('RoutingRule[' + i + ']:');
                var RoutingRule = result.InterfaceResult.RoutingRules[i];
                console.log('Condition[HttpErrorCodeReturnedEquals]-->' + RoutingRule['Condition']
['HttpErrorCodeReturnedEquals']);
                console.log('Condition[KeyPrefixEquals]-->' + RoutingRule['Condition']['KeyPrefixEquals']);
                console.log('Redirect[HostName]-->' + RoutingRule['Redirect']['HostName']);
                console.log('Redirect[HttpRedirectCode]-->' + RoutingRule['Redirect']
['HttpRedirectCode']);
                console.log('Redirect[Protocol]-->' + RoutingRule['Redirect']['Protocol']);
                console.log('Redirect[ReplaceKeyPrefixWith]-->' + RoutingRule['Redirect']
['ReplaceKeyPrefixWith']);
                console.log('Redirect[ReplaceKeyWith]-->' + RoutingRule['Redirect']['ReplaceKeyWith']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.26 删除桶的 Website 配置

功能说明

删除指定桶的Website配置。

方法定义

```
ObsClient.deleteBucketWebsite
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.deleteBucketWebsite({
    Bucket : 'bucketname'
},function (err, result){
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.27 设置桶的多版本状态

功能说明

设置桶的多版本状态。

方法定义

```
ObsClient.setBucketVersioning
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。

字段名	类型	约束	说明
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionStatus	String	必选	桶的多版本状态，支持的值： • Enabled • Suspended

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketVersioning({
    Bucket : 'bucketname',
    VersionStatus : 'Enabled'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.28 获取桶的多版本状态

功能说明

获取桶的多版本状态。

方法定义

```
ObsClient.getBucketVersioning
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。

字段名	类型	约束	说明
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
VersionStatus	String	桶的多版本状态。

代码样例

```
obsClient.getBucketVersioning({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('VersionStatus-->' + result.InterfaceResult.VersionStatus);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.29 获取桶的 CORS 配置

功能说明

获取指定桶的跨域资源共享规则。

方法定义

```
ObsClient.getBucketCors
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名		类型	说明
RequestId		String	OBS服务端返回的请求ID。
CorsRules		Array	桶的CORS规则列表。
	ID	String	CORS规则ID。
	AllowedMethod	Array of Strings	CORS规则允许的HTTP方法。
	AllowedOrigin	Array of Strings	CORS规则允许的请求来源（表示域名的字符串）。
	AllowedHeader	Array of Strings	CORS规则允许请求中可携带的头域。
	MaxAgeSeconds	String	CORS规则允许客户端可以对跨域请求返回结果的缓存时间，以秒为单位。
	ExposeHeader	Array of Strings	CORS规则允许响应中可返回的附加头域。

代码样例

```
obsClient.getBucketCors({
    Bucket : 'bucketname'
}, function(err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            for(var k=0;k<result.InterfaceResult.CorsRules.length;k++){
                console.log('CorsRule[',k,']');
                console.log('CorsRule[ID]-->' + result.InterfaceResult.CorsRules[k]['ID']);
                console.log('CorsRule[MaxAgeSeconds]-->' + result.InterfaceResult.CorsRules[k]
['MaxAgeSeconds']);
                for (var i=0;i<result.InterfaceResult.CorsRules[k]['AllowedMethod'].length;i++){
                    console.log('CorsRule[AllowedMethod][' , i ,']-->'+result.InterfaceResult.CorsRules[k]
['AllowedMethod'][i]);
                }
                for(var i=0;i< result.InterfaceResult.CorsRules[k]['AllowedOrigin'].length;i++){
                    console.log('CorsRule[AllowedOrigin][',i ,']-->'+result.InterfaceResult.CorsRules[k]
['AllowedOrigin'][i]);
                }
                for(var i=0;i<result.InterfaceResult.CorsRules[k]['AllowedHeader'].length;i++){
                    console.log('CorsRule[AllowedHeader][',i,']-->'+result.InterfaceResult.CorsRules[k]
['AllowedHeader'][i]);
                }
                for(var i=0;i<result.InterfaceResult.CorsRules[k]['ExposeHeader'].length;i++){
                    console.log('CorsRule[ExposeHeader][',i ,']-->'+result.InterfaceResult.CorsRules[k]
['ExposeHeader'][i]);
                }
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
```

```
        }
    }
});
```

4.30 删除桶的 CORS 配置

功能说明

删除指定桶的跨域资源共享规则。

方法定义

```
ObsClient.deleteBucketCors
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.deleteBucketCors({
    Bucket : 'bucketname'
}, function (err, result){
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.31 设置桶标签

功能说明

设置桶的标签。

方法定义

```
ObsClient.setBucketTagging
```

请求参数

字段名		类型	约束	说明
Bucket		String	必选	桶名。
RequestDate		String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Tags		Array	必选	桶标签列表。
	Key	String	必选	标签的名字，最大36个字符。不能为空，不能包含非打印字ASCII（0-31）、“=”、“*”、“<”、“>”、“\”。同一个桶，Tag中的key不能重复。
	Value	String	必选	标签的值，最大43个字符。可以为空，不能包含非打印字ASCII（0-31）、“=”、“*”、“<”、“>”、“\”。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setBucketTagging({
    Bucket : 'bucketname',
    Tags :[{Key:'tag1', Value : 'value1'}, {Key:'tag2', Value : 'value2'}]
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.32 获取桶标签

功能说明

获取指定桶的标签。

方法定义

```
ObsClient.getBucketTagging
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名		类型	说明
RequestId		String	OBS服务端返回的请求ID。
Tags		Array	桶标签列表。
	Key	String	标签的名字，最大36个字符。
	Value	String	标签的值，最大43个字符。

代码样例

```
obsClient.getBucketTagging({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            for(var i=0;i<result.InterfaceResult.Tags.length;i++){
                var tag = result.InterfaceResult.Tags[i];
                console.log('Tag-->' + tag.Key + ':' + tag.Value);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

4.33 删除桶标签

功能

删除指定桶的标签。

方法定义

```
ObsClient.deleteBucketTagging
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.deleteBucketTagging({
    Bucket : 'bucketname'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5 对象相关接口

5.1 使用前需知

由于浏览器跨域问题，调用对象相关接口前，需要首先配置其所在桶的CORS，如下：

- 设置允许的请求来源为：*
- 设置允许的HTTP方法为：PUT, GET, POST, DELETE, HEAD
- 设置允许请求中可携带的头域为：*
- 设置允许响应中可返回的附加头域为：
  - ETag
  - Content-Type
  - Content-Length
  - Cache-Control
  - Content-Disposition
  - Content-Encoding
  - Content-Language
  - Expires
  - x-obs-request-id
  - x-obs-id-2
  - x-reserved-indicator
  - x-obs-api
  - x-obs-version-id
  - x-obs-copy-source-version-id
  - x-obs-storage-class
  - x-obs-delete-marker
  - x-obs-expiration
  - x-obs-website-redirect-location
  - x-obs-restore
  - x-obs-version

- x-obs-object-type
- x-obs-next-append-position

说明

ObsClient的对象相关操作接口函数均支持首字母大小写，如ObsClient.putObject和ObsClient.PutObject是相同的函数。

5.2 上传对象

功能说明

上传单个对象到指定桶中。

方法定义

```
ObsClient.putObject
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
ACL	String	可选	创建对象时可指定的**预定义访问策略**。
StorageClass	String	可选	创建对象时可指定的对象的**存储类型**。
Body	String	可选	待上传对象的内容。
SourceFile	File 或 Blob	可选	待上传的文件（浏览器必须支持FileReader）。
ProgressCallback	Function	可选	获取上传进度的回调函数。 **说明** 该回调函数依次包含三个参数：已上传的字节数、总字节数、已使用的时间（单位：秒）。
Offset	Number	可选	当设置了SourceFile时有效，代表源文件中某一分段的起始偏移大小，默认值为0， 单位为字节。
Metadata	Object	可选	待上传对象的自定义元数据。

字段名	类型	约束	说明
WebsiteRedirectLocation	String	可选	当桶设置了Website配置，该参数指明对象的重定向地址。
Expires	Number	可选	待上传对象的生命周期，单位：天。
SuccessActionRedirect	String	可选	上传对象成功后的重定向的地址。
ContentType	String	可选	待上传对象的MIME类型。
ContentLength	Number	可选	当设置了SourceFile时有效，代表待上传对象数据的长度。
ContentMD5	String	可选	待上传对象数据的MD5值（经过Base64编码），提供给OBS服务端，校验数据完整性。
SseKms	String	可选	以SSE-KMS方式加密对象，支持的值： ● kms
SseKmsKey	String	可选	SSE-KMS方式下加密的主密钥，可为空。
SseC	String	可选	以SSE-C方式加密对象，支持的值： ● AES256
SseCKey	String	可选	SSE-C方式下加密的密钥，由AES256算法得到。

说明

- Body与SourceFile不能同时使用。
- 当Body与SourceFile都为空时，上传对象的大小为0字节。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	对象的ETag值。
VersionId	String	对象的版本号。
StorageClass	String	对象的存储类型，当对象存储类型是标准存储时，该值为空。
SseKms	String	SSE-KMS方式的算法。

字段名	类型	说明
SseKmsKey	String	SSE-KMS加密方式下使用的KMS主密钥的ID值。
SseC	String	SSE-C方式的算法。
SseCKeyMd5	String	SSE-C方式下加密使用密钥的MD5值，该值用于验证密钥传输过程中是否出错。

代码样例

```
obsClient.putObject({
    Bucket : 'bucketname',
    Key : 'objectkey',
    Metadata:{meta1:'value1', meta2:'value2'},
    Body : 'Hello OBS',
    ContentType: 'text/plain'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('ETag-->' + result.InterfaceResult.ETag);
            console.log('VersionId-->' + result.InterfaceResult.VersionId);
            console.log('StorageClass-->' + result.InterfaceResult.StorageClass);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.3 追加上传

功能说明

对同一个对象追加数据内容。

方法定义

```
ObsClient.appendObject
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。

字段名	类型	约束	说明
Position	Number	必选	追加上传的位置，第一次追加上传时必须为0。
ACL	String	可选	创建对象时可指定的**预定义访问策略**。
StorageClass	String	可选	创建对象时可指定的对象的**存储类型**。
Body	String	可选	待上传对象的内容。
SourceFile	File 或 Blob	可选	待上传的文件（浏览器必须支持FileReader）。
ProgressCallback	Function	可选	获取上传进度的回调函数。 **说明** 该回调函数依次包含三个参数：已上传的字节数、总字节数、已使用的时间（单位：秒）。
Offset	Number	可选	当设置了SourceFile时有效，代表源文件中某一分段的起始偏移大小，默认值为0， 单位为字节。
Metadata	Object	可选	待上传对象的自定义元数据。
WebsiteRedirectLocation	String	可选	当桶设置了Website配置，该参数指明对象的重定向地址。
Expires	Number	可选	待上传对象的生命周期，单位：天。
SuccessActionRedirect	String	可选	上传对象成功后的重定向的地址。
ContentType	String	可选	待上传对象的MIME类型。
ContentLength	Number	可选	当设置了SourceFile时有效，代表待上传对象数据的长度。
ContentMD5	String	可选	待上传对象数据的MD5值（经过Base64编码），提供给OBS服务端，校验数据完整性。
SseKms	String	可选	以SSE-KMS方式加密对象，支持的值： • kms
SseKmsKey	String	可选	SSE-KMS方式下加密的主密钥，可为空。
SseC	String	可选	以SSE-C方式加密对象，支持的值： • AES256

字段名	类型	约束	说明
SseCKey	Buffer	可选	SSE-C方式下加密的密钥，由AES256算法得到。

说明

- Body与SourceFile不能同时使用。
- 当Body与SourceFile都为空时，上传对象的大小为0字节。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	对象的ETag值。
NextPosition	String	下次追加上传的位置。
StorageClass	String	对象的存储类型，当对象存储类型是标准存储时，该值为空。

代码样例

```
obsClient.appendObject({
        Bucket : 'bucketname',
        Key : 'objectkey',
//      SourceFile : 'localfile',
        Body : 'Hello OBS',
        Position: 0
}, function (err, result) {
        if(err){
                console.error('Error-->' + err);
        }else{
                if(result.CommonMsg.Status < 300){
                        console.log('RequestId-->' + result.InterfaceResult.RequestId);
                        console.log('ETag-->' + result.InterfaceResult.ETag);
                        console.log('NextPosition-->' + result.InterfaceResult.NextPosition);
                }else{
                        console.log('Code-->' + result.CommonMsg.Code);
                        console.log('Message-->' + result.CommonMsg.Message);
                }
        }
});
```

5.4 下载对象

功能说明

下载指定桶中的对象。

方法定义

```
ObsClient.getObject
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionId	String	可选	对象的版本号。
IfMatch	String	可选	如果对象的ETag值与该参数值相同，则返回对象内容，否则返回异常码。
IfModifiedSince	String	可选	如果对象的修改时间晚于该参数值指定的时间，则返回对象内容，否则返回异常码。该参数值必须符合http://www.ietf.org/rfc/rfc2616.txt规定的HTTP时间格式。
IfNoneMatch	String	可选	如果对象的ETag值与该参数值不相同，则返回对象内容，否则返回异常码。
IfUnmodifiedSince	String	可选	如果对象的修改时间早于该参数值指定的时间，则返回对象内容，否则返回异常码。该参数值必须符合http://www.ietf.org/rfc/rfc2616.txt规定的HTTP时间格式。
Range	String	可选	指定下载的范围，取值区间：[0，对象长度-1]，格式：bytes=x-y。如果Range的最大长度超出对象长度-1，仍旧取对象长度-1。
Origin	String	可选	预请求指定的跨域请求Origin（通常为域名）。
RequestHeader	String	可选	跨域请求可以使用的HTTP头域。
ResponseCacheControl	String	可选	重写响应中的Cache-Control头。
ResponseContentDisposition	String	可选	重写响应中的Content-Disposition头。
ResponseContentEncoding	String	可选	重写响应中的Content-Encoding头。

字段名	类型	约束	说明
ResponseContentLanguage	String	可选	重写响应中的Content-Language头。
ResponseContentType	String	可选	重写响应中的Content-Type头。
ResponseExpires	String	可选	重写响应中的Expires头。
ImageProcess	String	可选	图片处理参数。
SaveByType	String	可选	下载对象的方式，支持的值： ● text ● arraybuffer ● blob ● file
ProgressCallback	Function	可选	获取下载进度的回调函数。 **说明** 该回调函数依次包含三个参数：已下载的字节数、总字节数、已使用的时间（单位：秒）。
SseC	String	可选	以SSE-C方式解密对象，支持的值： ● AES256
SseCKey	String	可选	SSE-C方式下解密的密钥，由AES256算法算出。

说明

- 如果包含IfUnmodifiedSince并且不符合或者包含IfMatch并且不符合，抛出异常中HTTP状态码为：412 precondition failed。
- 如果包含IfModifiedSince并且不符合或者包含IfNoneMatch并且不符合，抛出异常中HTTP状态码为：304 Not Modified。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
DeleteMarker	String	标识删除的对象是否是删除标记。
LastModified	String	对象的最近一次修改时间。
ContentLength	String	对象数据的长度。
CacheControl	String	响应中的Cache-Control头。
ContentDisposition	String	响应中的Content-Disposition头。

字段名	类型	说明
ContentEncoding	String	响应中的Content-Encoding头
ContentLanguage	String	响应中的Content-Language头
ContentType	String	对象的MIME类型。
Expires	String	响应中的Expires头。
ETag	String	对象的ETag值。
VersionId	String	对象的版本号。
WebsiteRedirectLocation	String	当桶设置了Website配置，该参数指明对象的重定向地址。
StorageClass	String	对象的存储类型，当对象存储类型是标准存储时，该值为空。
Restore	String	归档存储类型对象的恢复状态。
Expiration	String	对象的详细过期信息。
Content	String 或 ArrayBuffer 或 Blob 或 Object	对象的内容。当SavaByType未设置或者为text时该值为String对象；当SavaByType为arraybuffer时该值为ArrayBuffer对象；当SaveByType为blob时该值为Blob对象；当SaveByType为file时该值为包含下载该对象的URL（有效期3600秒）的Object对象。
Metadata	Object	对象自定义元数据。需要在桶的CORS配置中增加允许响应中可返回的附加头域。例如，新增x-amz-meta-property1以获取自定义元数据property1。

代码样例

```
obsClient.getObject({
    Bucket : 'bucketname',
    Key : 'objectkey',
    Range: 'bytes=0-10',
    SaveByType: 'file'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('ETag-->' + result.InterfaceResult.ETag);
            console.log('VersionId-->' + result.InterfaceResult.VersionId);
            console.log('ContentLength-->' + result.InterfaceResult.ContentLength);
            console.log('DeleteMarker-->' + result.InterfaceResult.DeleteMarker);
            console.log('LastModified-->' + result.InterfaceResult.LastModified);
            console.log('StorageClass-->' + result.InterfaceResult.StorageClass);
```

```
            console.log('Content-->' + result.InterfaceResult.Content);
            console.log('Metadata-->' + JSON.stringify(result.InterfaceResult.Metadata));
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.5 复制对象

功能说明

为指定桶中的对象创建一个副本。

方法定义

```
ObsClient.copyObject
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	目标桶名。
Key	String	必选	目标对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
ACL	String	可选	创建对象时可指定的**预定义访问策略**。
StorageClass	String	可选	创建对象时可指定的对象的**存储类型**。
CopySource	String	必选	指定源桶、源对象和源对象版本号（可为空）的参数，格式：源桶名/源对象名?versionId=源对象版本号。
CopySourceIfMatch	String	可选	如果源对象的ETag值与该参数值相同，则进行复制，否则返回异常码。
CopySourceIfModifiedSince	String	可选	如果源对象的修改时间晚于该参数值指定的时间，则进行复制，否则返回异常码。该参数值必须符合http://www.ietf.org/rfc/rfc2616.txt规定的HTTP时间格式。
CopySourceIfNoneMatch	String	可选	如果源对象的ETag值与该参数值不相同，则进行复制，否则返回异常码。

字段名	类型	约束	说明
CopySourceIfUnmodifiedSince	String	可选	如果源对象的修改时间早于该参数值指定的时间，则进行复制，否则返回异常码。该参数值必须符合http://www.ietf.org/rfc/rfc2616.txt规定的HTTP时间格式。
CacheControl	String	可选	复制时重写响应中的Cache-Control头。
ContentDisposition	String	可选	复制时重写响应中的Content-Disposition头。
ContentEncoding	String	可选	复制时重写响应中的Content-Encoding头。
ContentLanguage	String	可选	复制时重写响应中的Content-Language头。
ContentType	String	可选	复制时重写响应中的Content-Type头。
Expires	String	可选	复制时重写响应中的Expires头。
MetadataDirective	String	可选	**复制策略**。
Metadata	Object	可选	目标对象的自定义元数据。
WebsiteRedirectLocation	String	可选	当桶设置了Website配置，该参数指明对象的重定向地址。
SseKms	String	可选	以SSE-KMS方式加密目标对象，支持的值： ● kms
SseKmsKey	String	可选	SSE-KMS方式下加密目标对象的主密钥，可为空。
SseC	String	可选	以SSE-C方式加密目标对象，支持的值： ● AES256
SseCKey	String	可选	SSE-C方式下加密目标对象的密钥，由AES256算法算出。
CopySourceSseC	String	可选	以SSE-C方式解密源对象，支持的值： ● AES256
CopySourceSseCKey	String	可选	SSE-C方式下解密源对象的密钥，由AES256算法算出。

说明

- 如果包含CopySourceIfUnmodifiedSince并且不符合，或者包含CopySourceIfMatch并且不符合，或者包含CopySourceIfModifiedSince并且不符合，或者包含CopySourceIfNoneMatch并且不符合，抛出异常中HTTP状态码为：412 precondition failed。
- CopySourceIfModifiedSince和CopySourceIfNoneMatch可以一起使用；CopySourceIfUnmodifiedSince和CopySourceIfMatch可以一起使用。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	目标对象的ETag值。
LastModified	String	目标对象的最近一次修改时间。
VersionId	String	目标对象的版本号，如果目标桶未开启多版本状态则该值为空。
CopySourceVersionId	String	源对象的版本号，如果源桶未开启多版本状态则该值为空。

代码样例

```
obsClient.copyObject({
    Bucket:'bucketname',
    Key:'objectkey',
    CopySource:'srcbucketname/srcobjectkey',
    Metadata:{meta1:'value1'}
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('ETag-->' + result.InterfaceResult.ETag);
            console.log('VersionId-->' + result.InterfaceResult.VersionId);
            console.log('CopySourceVersionId-->' + result.InterfaceResult.CopySourceVersionId);
            console.log('LastModified-->' + result.InterfaceResult.LastModified);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.6 删除对象

功能说明

删除指定桶中的对象。

方法定义

```
ObsClient.deleteObject
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionId	String	可选	待删除对象的版本号。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
DeleteMarker	String	标识删除的对象是否是删除标记。
VersionId	String	待删除对象的版本号。

代码样例

```
obsClient.deleteObject({
    Bucket:'bucketname',
    Key:'objectkey'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('VersionId-->' + result.InterfaceResult.VersionId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.7 批量删除对象

功能说明

批量删除指定桶中的多个对象。

方法定义

```
ObsClient.deleteObjects
```

请求参数

字段名		类型	约束	说明
Bucket		String	必选	桶名。
RequestDate		String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
Objects		Array	必选	待删除的对象列表。
	Key	String	必选	待删除对象的名称。
	VersionId	String	可选	待删除对象的版本号。
Quiet		Boolean	可选	批量删除对象的响应方式，false表示详细模式，返回删除成功和删除失败的所有结果；true表示简单模式，只返回删除过程中出错的结果。

返回结果（InterfaceResult）

字段名		类型	说明
RequestId		String	OBS服务端返回的请求ID。
Deleteds		Array	删除成功的对象列表。
	Key	String	删除成功的对象名。
	VersionId	String	删除成功的对象版本号。
	DeleteMarker	String	标识删除的对象是否是删除标记。
	DeleteMarkerVersionId	String	删除标记的版本号。
Errors		Array	删除失败的对象列表。
	Key	String	删除失败的对象名。
	VersionId	String	删除失败的对象版本号。
	Code	String	删除失败的错误码。
	Message	String	删除失败的错误消息。

代码样例

```
obsClient.deleteObjects({
    Bucket:'bucketname',
    Quiet:false,
    Objects:[{Key:'objectkey'},{Key:'objectkey2'}]
```

```
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Deleteds:');
            for(var i=0;i<result.InterfaceResult.Deleteds.length;i++){
                console.log('Deleted[' + i + ']:');
                console.log('Key-->'+result.InterfaceResult.Deleteds[i]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.Deleteds[i]['VersionId']);
                console.log('DeleteMarker-->' + result.InterfaceResult.Deleteds[i]['DeleteMarker']);
                console.log('DeleteMarkerVersionId-->' + result.InterfaceResult.Deleteds[i]
['DeleteMarkerVersionId']);
            }
            console.log('Errors:');
            for(var i=0;i<result.InterfaceResult.Errors.length;i++){
                console.log('Error[' + i + ']:');
                console.log('Key-->' + result.InterfaceResult.Errors[i]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.Errors[i]['VersionId']);
                console.log('Code-->' + result.InterfaceResult.Errors[i]['Code']);
                console.log('Message-->' + result.InterfaceResult.Errors[i]['Message']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.8 获取对象元数据

功能说明

对指定桶中的对象发送HEAD请求，获取对象的元数据信息。

方法定义

```
ObsClient.getObjectMetadata
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionId	String	可选	对象的版本号。
SseC	String	可选	以SSE-C方式解密对象，支持的值： ● AES256
SseCKey	String	可选	SSE-C方式下解密的密钥，由AES256算法算出。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
LastModified	String	对象的最近一次修改时间。
ContentLength	String	对象数据的长度。
ContentType	String	对象的MIME类型。
ETag	String	对象的ETag值。
VersionId	String	对象的版本号。
WebsiteRedirectLocation	String	当桶设置了Website配置，该参数指明对象的重定向地址。
StorageClass	String	对象的存储类型，当对象存储类型是标准存储时，该值为空。
Restore	String	归档存储类型对象的恢复状态。
Expiration	String	对象的详细过期信息。
Metadata	Object	对象自定义元数据。需要在桶的CORS配置中增加允许响应中可返回的附加头域。例如，新增x-obs-meta-property1以获取自定义元数据property1。
ObjectType	String	对象是否可被追加上传。
NextPosition	String	下次追加上传的位置。

代码样例

```
obsClient.getObjectMetadata({
    Bucket:'bucketname',
    Key : 'objectkey'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('ETag-->' + result.InterfaceResult.ETag);
            console.log('VersionId-->' + result.InterfaceResult.VersionId);
            console.log('ContentLength-->' + result.InterfaceResult.ContentLength);
            console.log('Expiration-->' + result.InterfaceResult.Expiration);
            console.log('LastModified-->' + result.InterfaceResult.LastModified);
            console.log('StorageClass-->' + result.InterfaceResult.StorageClass);
        }else{
            console.log('Code-->' + result.CommonMsg.Status);
        }
    }
});
```

5.9 设置对象 ACL

功能说明

设置指定桶中对象的访问权限。

方法定义

```
ObsClient.setObjectAcl
```

请求参数

字段名			类型	约束	说明
Bucket			String	必选	桶名。
Key			String	必选	对象名。
RequestDate			String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionId			String	可选	对象的版本号。
ACL			String	可选	**预定义访问策略**。
Owner			Object	可选	对象的所有者。
	ID		String	必选	对象所有者的DomainId。
Delivered			Boolean	可选	桶的ACL是否向桶内对象传递。
Grants			Array	可选	被授权用户权限信息列表。
	Grantee		Object	必选	被授权用户。
		Type	String	必选	被授权的**用户类型**。
		ID	String	如果Type为“CanonicalUser”则必选，否则必须为空	被授权用户的DomainId。
		URI	String	如果Type为“Group”则必选，否则必须为空	被授权的**用户组**。

字段名		类型	约束	说明
	Permission	String	必选	被授予的权限。

说明

- Owner和Grants必须配套使用，且与ACL互斥。当设置了这两个字段时，不能设置ACL；反之，当设置了ACL时，不能设置Owner和Grants。
- Owner、Grants与ACL不能全为空。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.setObjectAcl({
        Bucket:'bucketname',
        Key : 'objectkey',
        Owner:{ID:'ownerid'},
        Grants:[
                {Grantee:{Type:'CanonicalUser',ID:'userid'},Permission:obsClient.enums.PermissionRead},
                {Grantee:{Type:'CanonicalUser',ID:'userid'},Permission:obsClient.enums.PermissionWrite},
                {Grantee:
{Type:'Group',URI:obsClient.enums.GroupAllUsers},Permission:obsClient.enums.PermissionWrite},
                {Grantee:
{Type:'Group',URI:obsClient.enums.GroupAllUsers},Permission:obsClient.enums.PermissionRead}
        ]
},function (err, result){
        if(err){
                console.error('Error-->' + err);
        }else{
                if(result.CommonMsg.Status < 300){
                        console.log('RequestId-->' + result.InterfaceResult.RequestId);
                }else{
                        console.log('Code-->' + result.CommonMsg.Code);
                        console.log('Message-->' + result.CommonMsg.Message);
                }
        }
});
```

5.10 获取对象 ACL

功能说明

获取指定桶中对象的访问权限。

方法定义

```
ObsClient.getObjectAcl
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionId	String	可选	对象的版本号。

返回结果（InterfaceResult）

字段名			类型	说明
RequestId			String	OBS服务端返回的请求ID。
VersionId			String	对象的版本号。
Owner			Object	对象的所有者。
	ID		String	对象所有者的DomainId。
	Name		String	对象所有者的名字。
Delivered			String	桶的ACL是否向桶内对象传递。
Grants			Array	被授权用户权限信息列表。
	Grantee		Object	被授权用户。
		Name	String	被授权用户的名字，当用户类型是Group时为空。
		ID	String	被授权用户的DomainId，当用户类型是Group时为空。
		URI	String	被授权的用户组，当用户类型是CanonicalUser时为空。
	Permission		String	被授予的权限。

代码样例

```
obsClient.getObjectAcl({
    Bucket:'bucketname',
    Key : 'objectkey'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
```

```
                console.log('Owner[ID]-->' + result.InterfaceResult.Owner.ID);
                for(var i=0;i<result.InterfaceResult.Grants.length;i++){
                    console.log('Grant[' + i + ']:');
                    console.log('Grantee[ID]-->' + result.InterfaceResult.Grants[i]['Grantee']['ID']);
                    console.log('Grantee[URI]-->' + result.InterfaceResult.Grants[i]['Grantee']['URI']);
                    console.log('Permission-->' + result.InterfaceResult.Grants[i]['Permission']);
                }
            }else{
                console.log('Code-->' + result.CommonMsg.Code);
                console.log('Message-->' + result.CommonMsg.Message);
            }
        }
});
```

5.11 初始化分传段任务

功能说明

在指定桶中初始化分段上传任务。

方法定义

```
ObsClient.initiateMultipartUpload
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
ACL	String	可选	创建对象时可指定的**预定义访问策略**。
StorageClass	String	可选	创建对象时可指定的对象的**存储类型**。
Metadata	Object	可选	对象的自定义元数据信息。
WebsiteRedirectLocation	String	可选	当桶设置了Website配置，该参数指明对象的重定向地址。
Expires	Number	可选	对象的生命周期，单位：天。
ContentType	String	可选	对象的MIME类型。
SseKms	String	可选	以SSE-KMS方式加密对象，支持的值： • kms
SseKmsKey	String	可选	SSE-KMS方式下加密的主密钥，可为空。

字段名	类型	约束	说明
SseC	String	可选	以SSE-C方式加密对象，支持的值： ● AES256
SseCKey	String	可选	SSE-C方式下加密的密钥，由AES256算法得到。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Bucket	String	分段上传任务的桶名。
Key	String	分段上传任务的对象名。
UploadId	String	分段上传任务的ID。

代码样例

```
obsClient.initiateMultipartUpload({
    Bucket:'bucketname',
    Key : 'objectkey',
    ContentType : 'text/plain'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Bucket-->' + result.InterfaceResult.Bucket);
            console.log('Key-->' + result.InterfaceResult.Key);
            console.log('UploadId-->' + result.InterfaceResult.UploadId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.12 上传段

功能说明

初始化分段上传任务后，通过分段上传任务的ID，上传段到指定桶中。除了最后一段以外，其他段的大小范围是100KB~5GB；最后段大小范围是0~5GB。上传的段号也有范围限制，其范围是1~10000。

方法定义

```
ObsClient.uploadPart
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
PartNumber	Number	必选	段号，取值范围：1~10000。
UploadId	String	必选	分段上传任务的ID。
ContentMD5	String	可选	待上传对象数据的MD5值（经过Base64编码），提供给OBS服务端，校验数据完整性。
Body	String	可选	待上传对象的内容。
SourceFile	File 或 Blob	可选	待上传段的源文件（浏览器必须支持FileReader）。
ProgressCallback	Function	可选	获取上传进度的回调函数。 **说明** 该回调函数依次包含三个参数：已上传的字节数、总字节数、已使用的时间（单位：秒）。
Offset	Number	可选	当设置了SourceFile时有效，代表源文件中某一分段的起始偏移大小，默认值为0， 单位为字节。
PartSize	Number	可选	当设置了SourceFile时有效，代表源文件中某一分段的大小，默认值为文件大小减去Offset的剩下字节数，单位为字节。除最后一段的大小范围是0~5GB外，其他段的大小范围是100KB~5GB。
SseC	String	可选	以SSE-C方式加密段，支持的值： AES256
SseCKey	String	可选	SSE-C方式下加密的密钥，由AES256算法得到。

说明

- Body与SourceFile不能同时使用。
- 当Body与SourceFile都为空时，上传对象的大小为0字节。
- Offset、PartSize和SourceFile配套使用，用于指定上传源文件中的某一分段数据。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	当前上传段的ETag值。

代码样例

```
obsClient.uploadPart({
    Bucket:'bucketname',
    Key : 'objectkey',
    UploadId : 'uploadid',
    PartNumber : 1,
    Body : 'Hello OBS'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('ETag-->' + result.InterfaceResult.ETag);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.13 复制段

功能说明

初始化分段上传任务后，通过分段上传任务的ID，复制段到指定桶中。

方法定义

```
ObsClient.copyPart
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。

字段名	类型	约束	说明
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
PartNumber	Number	必选	段号，取值范围：[1，10000]。
UploadId	String	必选	分段上传任务的ID。
CopySource	String	必选	指定源桶、源对象和源对象版本号（可为空）的参数，格式：源桶名/源对象名?versionId=源对象版本号。
CopySourceRange	String	可选	指定复制源对象的范围，取值区间：[0，源对象长度-1]，格式：bytes=x-y。如果CopySourceRange的最大长度超出源对象长度-1，仍旧取源对象长度-1。
SseC	String	可选	以SSE-C方式加密目标段，支持的值： ● AES256
SseCKey	String	可选	SSE-C方式下加密目标段的密钥，由AES256算法算出。
CopySourceSseC	String	可选	以SSE-C方式解密源对象，支持的值： ● AES256
CopySourceSseCKey	String	可选	SSE-C方式下解密源对象的密钥，由AES256算法算出。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	目标段的ETag值。
LastModified	String	目标段的最近一次修改时间。

代码样例

```
obsClient.copyPart({
    Bucket:'bucketname',
    Key : 'objectkey',
    PartNumber : 1,
    UploadId : 'uploadid',
    CopySource : 'sourcebucketname/sourceobjectkey',
```

```
    CopySourceRange : 'bytes=0-10'
},function (err, result){
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('LastModified-->' + result.InterfaceResult.LastModified);
            console.log('ETag-->' + result.InterfaceResult.ETag);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.14 列举已上传的段

功能说明

通过分段上传任务的ID，列举指定桶中已上传的段。

方法定义

```
ObsClient.listParts
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
UploadId	String	必选	分段上传任务的ID。
MaxParts	Number	可选	列举已上传段的返回结果最大段数目，即分页时每一页中段数目。
PartNumberMarker	Number	可选	列举已上传段的起始位置，只有PartNumber大于该参数的段会被列出。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
Bucket	String	桶名。
Key	String	对象名。

字段名		类型	说明
UploadId		String	分段上传任务的ID。
PartNumberMarker		String	列举已上传段的起始位置，与请求中的该参数对应。
NextPartNumberMarker		String	下次列举已上传段请求的起始位置。
MaxParts		String	列举已上传段的返回结果最大段数目，与请求中的该参数对应。
IsTruncated		String	表明本次请求是否返回了全部结果，“true”表示没有返回全部结果；“false”表示已返回了全部结果。
Parts		Array	已上传段列表。
	PartNumber	String	段号。
	LastModified	String	段的最后上传时间。
	ETag	String	段的ETag值。
	Size	String	段的大小。
Initiator		Object	分段上传任务的创建者。
	ID	String	创建者的DomainId。
Owner		Object	和Initiator相同，代表分段上传任务的创建者。
	ID	String	创建者的DomainId。
StorageClass		String	待分段上传对象的存储类型。

代码样例

```
obsClient.listParts({
    Bucket:'bucketname',
    Key : 'objectkey',
    UploadId : 'uploadid',
    MaxParts : 10
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Bucket-->' + result.InterfaceResult.Bucket);
            console.log('Key-->' + result.InterfaceResult.Key);
            console.log('UploadId-->' + result.InterfaceResult.UploadId);
            console.log('PartNumberMarker-->' + result.InterfaceResult.PartNumberMarker);
            console.log('NextPartNumberMarker-->' + result.InterfaceResult.NextPartNumberMarker);
            console.log('MaxParts-->' + result.InterfaceResult.MaxParts);
            console.log('IsTruncated-->' + result.InterfaceResult.IsTruncated);
            console.log('StorageClass-->' + result.InterfaceResult.StorageClass);
            console.log('Initiator[ID]-->' + result.InterfaceResult.Initiator['ID']);
            console.log('Owner[ID]-->' + result.InterfaceResult.Owner['ID']);
```

```
            for(var i=0;i<result.InterfaceResult.Parts.length;i++){
                console.log('Part['+i+']:');
                console.log('PartNumber-->' + result.InterfaceResult.Parts[i]['PartNumber']);
                console.log('LastModified-->' + result.InterfaceResult.Parts[i]['LastModified']);
                console.log('ETag-->' + result.InterfaceResult.Parts[i]['ETag']);
                console.log('Size-->' + result.InterfaceResult.Parts[i]['Size']);
            }
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.15 合并段

功能说明

通过分段上传任务的ID，合并指定桶中已上传的段。

方法定义

```
ObsClient.completeMultipartUpload
```

请求参数

字段名		类型	约束	说明
Bucket		String	必选	桶名。
Key		String	必选	对象名。
RequestDate		String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
UploadId		String	必选	分段上传任务的ID。
Parts		Array	必选	合并的段列表。
	PartNumber	String	必选	段号。
	ETag	String	必选	段的ETag值。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	合并段后根据各个段的ETag值计算出的结果。
Bucket	String	合并段所在的桶。

字段名	类型	说明
Key	String	合并段后得到的对象名。
Location	String	合并段后得到的对象的url。
VersionId	String	合并段后得到的对象版本号。

代码样例

```
obsClient.completeMultipartUpload({
    Bucket:'bucketname',
    Key : 'objectkey',
    UploadId : 'uploadid',
    Parts : [{PartNumber : 1, ETag : 'etag1'}, {PartNumber : 2, ETag : 'etag2'}]
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('Bucket-->' + result.InterfaceResult.Bucket);
            console.log('Key-->' + result.InterfaceResult.Key);
            console.log('Location-->' + result.InterfaceResult.Location);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.16 取消分段上传任务

功能说明

通过分段上传任务的ID，取消指定桶中的分段上传任务。

方法定义

```
ObsClient.abortMultipartUpload
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
UploadId	String	必选	分段上传任务的ID。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。

代码样例

```
obsClient.abortMultipartUpload({
    Bucket:'bucketname',
    Key : 'objectkey',
    UploadId : 'uploadid'
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

5.17 恢复归档存储对象

功能说明

恢复指定桶中的归档存储对象。

方法定义

```
ObsClient.restoreObject
```

请求参数

字段名	类型	约束	说明
Bucket	String	必选	桶名。
Key	String	必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
VersionId	String	可选	待恢复归档存储对象的版本号。
Days	Number	必选	恢复对象的保存时间（单位：天），取值范围：[1, 30]。
Tier	String	可选	**恢复选项**。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
RestoreStatus	String	恢复对象的状态，AVAILABLE代表可以下载，INPROGRESS代表正在恢复。

代码样例

```
obsClient.restoreObject({
    Bucket:'bucketname',
    Key : 'objectkey',
    Days : 1,
    Tier : obsClient.enums.RestoreTierExpedited
},function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        if(result.CommonMsg.Status < 300){
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
            console.log('RestoreStatus-->' + result.InterfaceResult.RestoreStatus);
        }else{
            console.log('Code-->' + result.CommonMsg.Code);
            console.log('Message-->' + result.CommonMsg.Message);
        }
    }
});
```

6 其他接口

6.1 生成带授权信息的 URL

功能说明

通过访问密钥、请求方法类型、请求参数等信息生成一个在Query参数中携带鉴权信息的URL，以对OBS服务进行特定操作。

方法定义

```
ObsClient.createSignedUrlSync
```

请求参数

表 6-1 参数列表

字段名	类型	约束	说明
Method	String	必选	HTTP方法类型，支持的值： • GET • POST • PUT • DELETE • HEAD
Bucket	String	可选	桶名。
Key	String	可选	对象名。
SpecialParam	String	可选	特殊操作符，代表要访问的子资源，详情参见**SubResourceType**。
Expires	Number	可选	带授权信息的URL的过期时间（单位：秒），默认值：300。

字段名	类型	约束	说明
Headers	Object	可选	请求中携带的头域。
QueryParams	Object	可选	请求中携带的查询参数。

表 6-2 SubResourceType

常量值	适用接口
storagePolicy	设置/获取桶存储类型。
quota	设置/获取桶配额。
storageinfo	获取桶存量信息。
location	获取桶区域位置。
acl	设置/获取桶ACL、设置/获取对象ACL。
policy	设置/获取/删除桶策略。
cors	设置/获取/删除桶CORS配置。
versioning	设置/获取桶多版本状态。
website	设置/获取/删除桶Website配置。
logging	设置/获取桶日志管理配置。
lifecycle	设置/获取/删除桶生命周期配置。
notification	设置/获取桶时间通知配置。
tagging	设置/获取/删除桶标签。
delete	批量删除对象。
versions	列举桶内多版本对象。
uploads	列举桶内分段上传任务、初始化分段上传任务。
restore	恢复归档或深度归档存储对象。

返回结果

表 6-3 返回结果

字段名	类型	说明
SignedUrl	String	带授权信息的URL。
ActualSignedRequestHeaders	Object	通过带授权信息的URL发起请求时实际应携带的头域。

代码样例

```
// 生成上传对象的带授权信息的URL
var putObjectResult = obsClient.createSignedUrlSync({Method : 'PUT', Bucket : 'bucketname', Key :
'objectkey', Headers : {'Content-Type' : 'text/plain'}});
console.log('SignedUrl-->' + putObjectResult['SignedUrl']);
console.log('ActualSignedRequestHeaders-->' +
JSON.stringify(putObjectResult['ActualSignedRequestHeaders']));

// 生成设置对象ACL的带授权信息的URL
var setObjectAclResult = obsClient.createSignedUrlSync({Method : 'PUT', Bucket : 'bucketname', Key :
'objectkey', SpecialParam: 'acl', Headers: {'x-amz-acl' : 'public-read'}});
console.log('SignedUrl-->' + setObjectAclResult['SignedUrl']);
console.log('ActualSignedRequestHeaders-->' +
JSON.stringify(setObjectAclResult['ActualSignedRequestHeaders']));

// 生成下载对象的带授权信息的URL
var getObjectResult = obsClient.createSignedUrlSync({Method : 'GET', Bucket : 'bucketname', Key :
'objectkey'});
console.log('SignedUrl-->' + getObjectResult['SignedUrl']);
console.log('ActualSignedRequestHeaders-->' +
JSON.stringify(getObjectResult['ActualSignedRequestHeaders']));

// 生成删除对象的带授权信息的URL
var deleteObjectResult = obsClient.createSignedUrlSync({Method : 'DELETE', Bucket : 'bucketname', Key :
'objectkey'});
console.log('SignedUrl-->' + deleteObjectResult['SignedUrl']);
console.log('ActualSignedRequestHeaders-->' +
JSON.stringify(deleteObjectResult['ActualSignedRequestHeaders']));
```

6.2 生成带授权信息的表单上传参数

功能说明

生成用于鉴权的请求参数，以进行基于浏览器的POST表单上传。

说明

使用SDK生成用于鉴权的请求参数包括两个：

- Policy，对应表单中policy字段。
- Signature，对应表单中的signature字段。

方法定义

```
ObsClient.createPostSignatureSync
```

请求参数

字段名	类型	约束	说明
Bucket	String	可选	桶名。
Key	String	可选	对象名，对应表单中的key字段。
Expires	Number	可选	表单上传鉴权参数的过期时间（单位：秒），默认值300。

字段名	类型	约束	说明
FormParams	Object	可选	除key、policy、signature外，表单上传时的其他参数，支持的值： • acl • cache-control • content-type • content-disposition • content-encoding • expires

返回结果

字段名	类型	说明
OriginPolicy	String	Policy未经过base64之前的值，仅用于校验。
Policy	String	表单中的policy。
Signature	String	表单中的signature。

代码样例

```
var formParams = {acl: 'public-read', 'content-type': 'text/plain'};
var res = obsClient.createPostSignatureSync({Bucket: 'bucketname', Key: 'objectkey', Expires:3600,
FormParams: formParams});

console.log('Policy-->' + res['Policy']);
console.log('Signature-->' + res['Signature']);
```

6.3 断点续传上传

功能说明

对分段上传的封装和加强，支持反馈上传进度、反馈上传事件、任务暂停和任务续传。

方法定义

```
ObsClient.uploadFile
```

请求参数

字段名	类型	约束	说明
Bucket	String	如未设置UploadCheckpoint则必选	桶名。
Key	String	如未设置UploadCheckpoint则必选	对象名。
RequestDate	String 或 Date	可选	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
SourceFile	File 或 Blob	如未设置UploadCheckpoint则必选	待上传的源文件（浏览器必须支持FileReader）。
UploadCheckpoint	Object	可选	断点续传记录对象，可通过ResumeCallback获取到。
PartSize	Number	可选	分段大小，单位字节，取值范围是100KB~5GB，默认为9MB。
TaskNum	Number	可选	分段上传时的最大并发数，默认为1。
ProgressCallback	Function	可选	获取上传进度的回调函数。 **说明** 该回调函数依次包含三个参数：已上传的字节数、总字节数、已使用的时间（单位：秒）。
EventCallback	Function	可选	获取上传事件的回调函数。 **说明** • 该回调函数依次包含三个参数：事件类型，事件参数，事件结果； • 事件类型可能的值：uploadPartSucceed、uploadPartFailed、uploadPartAborted、initiateMultipartUploadSucceed、initiateMultipartUploadFailed、completeMultipartUploadSucceed、completeMultipartUploadFailed、completeMultipartUploadAborted

字段名	类型	约束	说明
ResumeCallback	Function	可选	获取断点续传控制参数的回调函数。 **说明** • 该回调函数依次包含两个参数：取消断点续传上传任务控制参数，断点续传记录对象； • 可以调用取消断点续传上传任务控制参数的cancel方法来暂停断点续传上传任务； • 断点续传记录对象中包含sourceFile字段代表待上传的文件，如果浏览器重启后该字段需要调用者重新进行设置。
ACL	String	可选	创建对象时可指定的**预定义访问策略**。
StorageClass	String	可选	创建对象时可指定的对象的**存储类型**。
Metadata	Object	可选	对象的自定义元数据信息。
WebsiteRedirectLocation	String	可选	当桶设置了Website配置，该参数指明对象的重定向地址。
Expires	Number	可选	对象的生命周期，单位：天。
ContentType	String	可选	对象的MIME类型。
SseKms	String	可选	以SSE-KMS方式加密对象，支持的值：kms
SseKmsKey	String	可选	SSE-KMS方式下加密的主密钥，可为空。
SseC	String	可选	以SSE-C方式加密对象，支持的值：AES256
SseCKey	String	可选	SSE-C方式下加密的密钥，由AES256算法得到。

返回结果（InterfaceResult）

字段名	类型	说明
RequestId	String	OBS服务端返回的请求ID。
ETag	String	合并段后根据各个段的ETag值计算出的结果。
Bucket	String	合并段所在的桶。
Key	String	合并段后得到的对象名。

字段名	类型	说明
Location	String	合并段后得到的对象的url。
VersionId	String	合并段后得到的对象版本号。

代码样例

```
var cp;
var hook;
obsClient.uploadFile({
        Bucket : 'bucketname',
        Key : 'objectkey',
        SourceFile : document.getElementById('input-file').files[0],
        PartSize : 9 * 1024 * 1024,
        ProgressCallback : function(transferredAmount, totalAmount, totalSeconds){
            console.log(transferredAmount * 1.0 / totalSeconds / 1024);
            console.log(transferredAmount * 100.0 / totalAmount);
        },
        ResumeCallback : function(resumeHook, uploadCheckpoint){
            hook = resumeHook;
            cp = uploadCheckpoint;
        }
}, function(err, result){
    if(err){
        console.error('Error-->' + err);
    }else{
            if(result.CommonMsg.Status < 300){
                    console.log('RequestId-->' + result.InterfaceResult.RequestId);
                    console.log('Bucket-->' + result.InterfaceResult.Bucket);
                    console.log('Key-->' + result.InterfaceResult.Key);
                    console.log('Location-->' + result.InterfaceResult.Location);
            }else{
                    console.log('Code-->' + result.CommonMsg.Code);
                    console.log('Message-->' + result.CommonMsg.Message);
            }
    }
});
```