

对象存储服务

# PHP SDK API 参考

文档版本

01

发布日期

2020-03-31



版权所有 © 华为技术有限公司 2023。保留一切权利。

非经本公司书面许可，任何单位和个人不得擅自摘抄、复制本文档内容的部分或全部，并不得以任何形式传播。

## 商标声明



HUAWEI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

<b>1 简介</b>	<b>1</b>
<b>2 初始化</b>	<b>2</b>
2.1 ObsClient 初始化	2
2.2 日志初始化	3
2.3 请求数组	4
2.4 SDK 公共结果对象	5
2.5 SDK 自定义异常	5
2.6 异步调用	6
<b>3 预定义常量</b>	<b>7</b>
3.1 权限类型	7
3.2 可被授权的用户组	8
3.3 可被授权用户类型	8
3.4 预定义访问策略	8
3.5 存储类型	9
3.6 取回选项	9
3.7 元数据复制策略	9
<b>4 桶相关接口</b>	<b>10</b>
4.1 创建桶	10
4.2 获取桶列表	11
4.3 判断桶是否存在	12
4.4 删除桶	13
4.5 列举桶内对象	14
4.6 列举桶内多版本对象	16
4.7 列举桶内分段上传任务	19
4.8 获取桶元数据	21
4.9 获取桶区域位置	23
4.10 获取桶存量信息	24
4.11 设置桶配额	24
4.12 获取桶配额	25
4.13 设置桶存储类型	26
4.14 获取桶存储类型	27
4.15 设置桶 ACL	28

4.16 获取桶 ACL.....	29
4.17 设置桶日志管理配置.....	31
4.18 获取桶日志管理配置.....	33
4.19 设置桶策略.....	34
4.20 获取桶策略.....	35
4.21 删除桶策略.....	36
4.22 设置桶的生命周期配置.....	37
4.23 获取桶的生命周期配置.....	40
4.24 删除桶的生命周期配置.....	41
4.25 设置桶的 Website 配置.....	42
4.26 获取桶的 Website 配置.....	44
4.27 删除桶的 Website 配置.....	46
4.28 设置桶的多版本状态.....	47
4.29 获取桶的多版本状态.....	48
4.30 设置桶的 CORS 配置.....	49
4.31 获取桶的 CORS 配置.....	51
4.32 删除桶的 CORS 配置.....	52
4.33 设置桶标签.....	53
4.34 获取桶标签.....	54
4.35 删除桶标签.....	55
<b>5 对象相关接口.....</b>	<b>57</b>
5.1 上传对象.....	57
5.2 下载对象.....	59
5.3 复制对象.....	63
5.4 删除对象.....	65
5.5 批量删除对象.....	66
5.6 获取对象元数据.....	68
5.7 设置对象 ACL.....	70
5.8 获取对象 ACL.....	72
5.9 初始化分传段任务.....	73
5.10 上传段.....	75
5.11 复制段.....	77
5.12 列举已上传的段.....	78
5.13 合并段.....	80
5.14 取消分段上传任务.....	82
5.15 取回归档存储对象.....	82
<b>6 其他接口.....</b>	<b>84</b>
6.1 生成带授权信息的 URL.....	84
6.2 生成带授权信息的表单上传参数.....	87
<b>A 修订记录.....</b>	<b>89</b>

# 1 简介

---

本文档提供了对象存储服务OBS（Object Storage Service）PHP 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 PHP 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PHP SDK开发指南》](#)。

# 2 初始化

## 2.1 ObsClient 初始化

### 功能说明

ObsClient是访问OBS服务的PHP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。

### 命名空间

类名	父命名空间
ObsClient	Obs

### 方法定义

- 构造函数形式：ObsClient(array \$parameter)
- 工厂方法形式：ObsClient::factory(array \$parameter)

### 参数描述

字段名	类型	约束	说明
key	string	必选	访问密钥中的AK。
secret	string	必选	访问密钥中的SK。
endpoint	string	必选	连接OBS的服务地址。包含协议类型、域名（或IP）、端口号。示例： https://your-endpoint:443。 您可以从 <a href="#">这里</a> 查看OBS当前开通的服务地址。

字段名	类型	约束	说明
ssl_verify	boolean 或 string	可选	验证服务端证书参数。可能的取值： <ul style="list-style-type: none"> <li>• 服务端pem格式根证书文件路径；</li> <li>• true：使用默认的CAs验证服务端证书；</li> <li>• false：表示不验证服务端证书。</li> </ul> 默认为false。
max_retry_count	integer	可选	HTTP/HTTPS连接异常时的请求重试次数。默认为3次。
socket_timeout	integer	可选	Socket层传输数据的超时时间（单位：秒）。默认为60秒。
connect_timeout	integer	可选	建立HTTP/HTTPS连接的超时时间（单位：秒）。默认为60秒。
chunk_size	integer	可选	读socket流时的块大小（单位：字节）。默认为65536字节。

## 代码样例

```
// 引入依赖库
require 'vendor/autoload.php';
// 使用源码安装时引入SDK代码库
// require 'obs-autoloader.php';
// 声明命名空间
use Obs\ObsClient;

// 创建ObsClient实例
$obsClient = new ObsClient([
    //推荐通过环境变量获取AKSK，这里也可以使用其他外部引入方式传入，如果使用硬编码可能会存在泄露风险。
    //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/intl/zh-cn/usermanual-ca/ca\_01\_0003.html
    'key' => getenv('ACCESS_KEY_ID'),
    'secret' => getenv('SECRET_ACCESS_KEY'),
    'endpoint' => 'https://your-endpoint',
    'ssl_verify' => false,
    'max_retry_count' => 1,
    'socket_timeout' => 20,
    'connect_timeout' => 20,
    'chunk_size' => 8196
]);
```

## 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(array $parameter)
```

## 参数描述

字段名	类型	约束	说明
FilePath	string	必选	保存日志文件的目录。
FileName	string	必选	日志文件名。
MaxFiles	integer	必选	最大可保留的日志文件个数。
Level	integer	必选	日志级别，SDK内部定义了四个integer类型的常量以对应不同的日志级别，有效值： <ul style="list-style-type: none"><li>• DEBUG ( 100 )</li><li>• INFO ( 200 )</li><li>• WARN ( 300 )</li><li>• ERROR ( 400 )</li></ul>

## 代码样例

```
$obsClient->initLog ([  
    'FilePath' => './logs',  
    'FileName' => 'OBS-SDK.log',  
    'MaxFiles' => 10,  
    'Level' => INFO  
]);
```

## 2.3 请求数组

### 功能说明

调用ObsClient的相关接口均需要传入请求数组（associative array类型）作为输入。对于桶操作接口，请求数组中固定包含Bucket用于指定桶名（ObsClient->listBuckets除外）；对于对象操作接口，请求数组中固定包含Bucket和Key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。
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ObsClient的相关接口完成后，没有异常抛出，则会返回SDK公共结果对象，表明操作成功；若抛出异常，则表明操作失败，此时应从[SDK自定义异常](#)实例中获取错误信息。

### 命名空间

类名	父命名空间
Model	Obs\Internal\Common

### 参数描述

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
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 SDK 自定义异常

### 功能说明

SDK自定义异常是由ObsClient统一抛出的异常，继承自\RuntimeException类。通常是OBS服务端错误，包含OBS错误码、错误信息等，便于用户定位问题，并做出适当的处理。

### 命名空间

类名	父命名空间
ObsException	Obs

## 方法描述

方法名	返回值类型	说明
ObsException->getExceptionCode	string	OBS服务端错误码。
ObsException->getExceptionMessage	string	OBS服务端错误描述。
ObsException->getRequestId	string	OBS服务端返回的请求ID。
ObsException->getHostId	string	请求的服务端ID。
ObsException->getResponse	GuzzleHttp \Psr7\Response	HTTP响应对象。
ObsException->getRequest	GuzzleHttp \Psr7\Request	HTTP请求对象。
ObsException->getStatusCode	integer	HTTP状态码。

## 2.6 异步调用

### 功能说明

桶相关接口和对象相关接口均支持以“Async”结尾的方式进行异步调用（例如，同步方法为ObsClient->putObject，则异步方法为ObsClient->putObjectAsync）。异步调用完成后会将返回结果输出到回调函数中，回调函数依次包含[SDK自定义异常](#)和[SDK公共结果对象](#)两个参数，如果回调函数中SDK自定义异常参数不为空，则表明操作失败；反之，则表明操作成功。

### 代码样例

```
$promise = $obsClient->putObjectAsync ( [
    'Bucket' => 'bucketname',
    'Key' => 'objectkey',
    'Body' => 'Hello OBS'
], function ($obsException, $resp) {
    if ($obsException === null) {
        printf ( "RequestId:%s\n", $resp ['RequestId'] );
    } else {
        printf ( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
        printf ( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
    }
});
$promise->wait ();
```

#### 说明

异步调用会立即返回异步调用结果对象（GuzzleHttp\Promise\Promise），需要调用该对象的wait方法以等待异步调用完成。

# 3 预定义常量

## 3.1 权限类型

访问方式	类型	说明
ObsClient::PermissionRead	string	若有桶的读权限，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。 若有对象的读权限，则可以获取该对象内容和元数据。
ObsClient::PermissionWrite	string	若有桶的写权限，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::PermissionReadAcp	string	若有读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::PermissionWriteAcp	string	若有写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::PermissionFullControl	string	若有桶的完全控制权限意味着拥有 PermissionRead、PermissionWrite、PermissionReadAcp和 PermissionWriteAcp的权限。 若有对象的完全控制权限意味着拥有 PermissionRead、PermissionWriteAcp和 PermissionWriteAcp的权限。

## 3.2 可被授权的用户组

访问方式	类型	说明
ObsClient::GranteeGroup	string	授权给用户组。
ObsClient::GranteeUser	string	授权给单个用户。

## 3.3 可被授权用户类型

访问方式	类型	说明
ObsClient::GroupAllUsers	string	所有用户。
ObsClient::GroupAuthenticatedUsers	string	授权用户，已废弃。
ObsClient::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::AclPrivate	string	私有读写。
ObsClient::AclPublicRead	string	公共读。
ObsClient::AclPublicReadWrite	string	公共读写。
ObsClient::AclPublicReadDelivered	string	桶公共读，桶内对象公共读。
ObsClient::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::StorageClassStandard	string	标准存储。
ObsClient::StorageClassWarm	string	低频访问存储。
ObsClient::StorageClassCold	string	归档存储。

## 3.6 取回选项

访问方式	类型	说明
ObsClient::RestoreTierExpedited	string	快速取回，取回耗时1~5分钟。
ObsClient::RestoreTierStandard	string	标准取回，取回耗时3~5小时。

## 3.7 元数据复制策略

访问方式	类型	说明
ObsClient::CopyMetadata	string	复制元数据。
ObsClient::ReplaceMetadata	string	替换元数据。

# 4 桶相关接口

## 4.1 创建桶

### 功能说明

按照用户指定的桶名创建一个新桶。新创建桶的桶名在OBS中必须是唯一的。除同一个用户重复创建同一区域同名桶外，其他场景重复创建同名桶均会失败。一个用户可以拥有的桶的数量不能超过100个。

### 方法定义

1. ObsClient->createBucket(array \$parameter)
2. ObsClient->createBucketAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。 桶命名规则如下： <ul style="list-style-type: none"><li>• 3~63个字符，数字或字母开头，支持小写字母、数字、“-”、“.”。</li><li>• 禁止使用类IP地址。</li><li>• 禁止以“-”或“.”开头及结尾。</li><li>• 禁止两个“.”相邻（如：“my..bucket”）。</li><li>• 禁止“.”和“-”相邻（如：“my-.bucket”和“my.-bucket”）。</li></ul>
ACL	string	可选	创桶时可指定的 <a href="#">预定义访问策略</a> 。
StorageClass	string	可选	创桶时可指定的桶的 <a href="#">存储类型</a> 。

字段名	类型	约束	说明
LocationConstraint	string	如果请求的OBS服务地址所在区域为默认区域，则可为空，否则为必选。	桶所在的区域。 该参数定义了桶将会被创建在哪个区域，如果使用的终端节点归属于默认区域华北-北京一（cn-north-1），可以不携带此参数；如果使用的终端节点归属于其他区域，则必须携带此参数。当前有效的区域名称可从 <a href="#">这里</a> 查询。有关OBS区域和终端节点的更多信息，请参考 <a href="#">地区和终端节点</a> 。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> createBucket([
        'Bucket' => 'bucketname',
        'ACL' => 'private',
        'StorageClass' => ObsClient::StorageClassStandard
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.2 获取桶列表

### 功能说明

查询桶列表，返回结果按照桶名字典顺序排列。

### 方法定义

1. ObsClient->listBuckets(array \$parameter)
2. ObsClient->listBucketsAsync(array \$parameter, callable \$callback)

## 请求参数

字段名	类型	约束	说明
QueryLocation	boolean	可选	是否同时查询桶的区域位置。

## 返回结果

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
Buckets	indexed array	桶列表。	
	Name	string	桶名。
	CreationDate	string	桶的创建时间。
	Location	string	桶的区域位置。
Owner	associative array	桶的所有者。	
	ID	string	桶所有者的DomainId。

## 代码样例

```
try{
    $resp = $obsClient -> listBuckets([
        'QueryLocation' => true
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("Owner[ID]:%s\n", $resp['Owner']['ID']);

    foreach ($resp['Buckets'] as $index => $bucket){
        printf("Buckets[%d]\n", $index + 1);
        printf("Name:%s\n", $bucket['Name']);
        printf("CreationDate:%s\n", $bucket['CreationDate']);
        printf("Location:%s\n", $bucket['Location']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.3 判断桶是否存在

### 功能说明

判断桶是否存在，抛出异常中HTTP状态码为200表明桶存在，否则返回404表明桶不存在。

## 方法定义

1. ObsClient->headBucket(array \$parameter)
2. ObsClient->headBucketAsync(array \$parameter, callable callback)

## 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> headBucket([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("Bucket exists\n");
} catch (Obs\Common\ObsException $obsException){
    if($obsException->getStatusCode() === 404){
        printf("Bucket does not exist\n");
    }else{
        printf("StatusCode:%d\n", $obsException->getStatusCode());
    }
}
```

## 4.4 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

1. ObsClient->deleteBucket(array \$parameter)
2. ObsClient->deleteBucketAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> deleteBucket([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.5 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

1. ObsClient->listObjects(array \$parameter)
2. ObsClient->listObjectsAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
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	integer	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

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
Location	string	桶的区域位置。	
Name	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	boolean	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	integer	列举对象的最大数目，与请求中的该参数对应。	
Contents	indexed 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	integer	对象的字节数。
	Key	string	对象名。
	LastModified	string	对象最近一次被修改的时间。
	Owner	associative array	对象的所有者。
	ID	string	对象所有者的DomainId。
	StorageClass	string	对象的存储类型。
CommonPrefixes	indexed 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

```
try{
    $resp = $obsClient -> listObjects([
        'Bucket' => 'bucketname',
        'Prefix' => 'prefix',
        'MaxKeys' => 100
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    foreach ($resp['Contents'] as $index => $content){
        printf("Contents[%d]\n", $index + 1);
        printf("ETag:%s\n", $content['ETag']);
        printf("Size:%s\n", $content['Size']);
        printf("StorageClass:%s\n", $content['StorageClass']);
        printf("Key:%s\n", $content['Key']);
        printf("LastModified:%s\n", $content['LastModified']);
        printf("Owner[ID]:%s\n", $content['Owner']['ID']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.6 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(array \$parameter)
- ObsClient->listVersionsAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
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	integer	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。

字段名	类型	约束	说明
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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
Location	string	桶的区域位置。
Name	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	boolean	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。
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。

字段名		类型	说明	
MaxKeys		integer	列举多版本对象的最大数目，与请求中的该参数对应。	
Versions		indexed array	多版本对象列表。	
	ETag	string	对象的MD5值。	
	Size	integer	对象的字节数。	
	Key	string	对象名。	
	VersionId	string	对象的版本号。	
	IsLatest	boolean	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	associative array	对象的拥有者。	
	ID	string	对象所有者的DomainId。	
	StorageClass	string	对象的存储类型	
DeleteMarkers		indexed array	删除标记列表。	
	Owner	associative array	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	boolean	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		indexed 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

```
try{
    $resp = $obsClient -> listVersions([
        'Bucket' => 'bucketname',
        'Prefix' => 'prefix',
        'MaxKeys' => 100
    ]);
}
```

```

printf("RequestId:%s\n", $resp['RequestId']);
printf("Versions:\n");
foreach ($resp['Versions'] as $index => $version){
    printf("Versions[%d]\n", $index + 1);
    printf("ETag:%s\n", $version['ETag']);
    printf("Size:%s\n", $version['Size']);
    printf("StorageClass:%s\n", $version['StorageClass']);
    printf("Key:%s\n", $version['Key']);
    printf("VersionId:%s\n", $version['VersionId']);
    printf("LastModified:%s\n", $version['LastModified']);
    printf("Owner[ID]:%s\n", $version['Owner']['ID']);
}

printf("DeleteMarkers:\n");
foreach ($resp['DeleteMarkers'] as $index => $deleteMarker){
    printf("DeleteMarkers[%d]\n", $index + 1);
    printf("Key:%s\n", $deleteMarker['Key']);
    printf("VersionId:%s\n", $deleteMarker['VersionId']);
    printf("IsLatest:%s\n", $deleteMarker['IsLatest']);
    printf("LastModified:%s\n", $deleteMarker['LastModified']);
    printf("Owner[ID]:%s\n", $deleteMarker['Owner']['ID']);
}
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}

```

## 4.7 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

1. ObsClient->listMultipartUploads(array \$parameter)
2. ObsClient->listMultipartUploadsAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
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	integer	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。

字段名	类型	约束	说明
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

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
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。	
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	integer	列举分段上传任务的最大数目，与请求中的该参数对应。	
IsTruncated	boolean	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	indexed 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。

字段名		类型	说明
	Initiator	associative array	分段上传任务的创建者。
		ID	创建者的DomainId。
	Owner	associative array	和Initiator相同，代表分段上传任务的创建者。
		ID	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		indexed 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

```
try{
    $resp = $obsClient -> listMultipartUploads([
        'Bucket' => 'bucketname',
        'Prefix' => 'prefix',
        'MaxUploads' => 100
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    foreach ($resp['Uploads'] as $index => $upload){
        printf("Versions[%d]\n", $index + 1);
        printf("UploadId:%s\n", $upload['UploadId']);
        printf("Initiated:%s\n", $upload['Initiated']);
        printf("StorageClass:%s\n", $upload['StorageClass']);
        printf("Key:%s\n", $upload['Key']);
        printf("Initiator[ID]:%s\n", $upload['Initiator']['ID']);
        printf("Initiator[DisplayName]:%s\n", $upload['Initiator']['DisplayName']);
        printf("Owner[ID]:%s\n", $upload['Owner']['ID']);
        printf("Owner[DisplayName]:%s\n", $upload['Owner']['DisplayName']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.8 获取桶元数据

### 功能说明

对桶发送HEAD请求，获取桶的存储类型，CORS规则（若已设置）等信息。

## 方法定义

1. ObsClient->getBucketMetadata(array \$parameter)
2. ObsClient->getBucketMetadata(array \$parameter, callable \$callback)

## 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
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。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
Location	string	桶的区域位置。
StorageClass	string	桶的存储类型，当桶存储类型是标准存储时，该值为空。
AllowOrigin	string	如果请求中的Origin满足服务端的CORS规则，则返回服务端CORS配置中的AllowedOrigin。
AllowHeader	string	如果请求的RequestHeader满足服务端的CORS规则，则返回服务端CORS配置中的AllowedHeader。
AllowMethod	string	服务端CORS规则中的AllowedMethod。
ExposeHeader	string	服务端CORS规则中的ExposeHeader。
MaxAgeSeconds	integer	服务端CORS规则中的MaxAgeSeconds。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketMetadata([
        'Bucket' => 'bucketname',
        'Origin' => 'http://www.a.com',
        'RequestHeader' => 'x-obs-header'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("StorageClass:%s\n", $resp['StorageClass']);
    printf("AllowOrigin:%s\n", $resp['AllowOrigin']);
    printf("AllowHeader:%s\n", $resp['AllowHeader']);
}
```

```
printf("AllowMethod:%s\n", $resp['AllowMethod']);  
printf("ExposeHeader:%s\n", $resp['ExposeHeader']);  
printf("MaxAgeSeconds:%s\n", $resp['MaxAgeSeconds']);  
}catch (Obs\Common\ObsException $obsException){  
    printf("StatusCode:%s\n", $obsException->getStatusCode());  
}
```

## 4.9 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

1. ObsClient->getBucketLocation(array \$parameter)
2. ObsClient->getBucketLocationAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
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

```
try{  
    $resp = $obsClient -> getBucketLocation([  
        'Bucket' => 'bucketname'  
    ]);  
    printf("RequestId:%s\n", $resp['RequestId']);  
    printf("Location:%s\n", $resp['Location']);  
}catch (Obs\Common\ObsException $obsException){  
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());  
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());  
}
```

## 4.10 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

1. ObsClient->getBucketStorageInfo(array \$parameter)
2. ObsClient->getBucketStorageInfo(array \$parameter, callback \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果 ( InterfaceResult )

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
Size	double	桶的空间大小。
ObjectNumber	integer	桶内对象个数。

### 代码样例

```
try{
    $resp = $obsClient -> getBucketStorageInfo([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("Size:%s\n", $resp['Size']);
    printf("ObjectNumber:%s\n", $resp['ObjectNumber']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.11 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

1. ObsClient->setBucketQuota(array \$parameter)
2. ObsClient->setBucketQuotaAsync(array \$parameter, callable \$callback)

## 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
StorageQuota	integer	必选	桶的配额值，非负整数。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketQuota([
        'Bucket' => 'bucketname',
        'StorageQuota' => 100 * 1024 * 1024
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.12 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

1. ObsClient->getBucketQuota(array \$parameter)
2. ObsClient->getBucketQuotaAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
StorageQuota	integer	桶的配额值。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketQuota([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("StorageQuota:%s\n", $resp['StorageQuota']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.13 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

1. ObsClient->setBucketStoragePolicy(array \$parameter)
2. ObsClient->setBucketStoragePolicyAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
StorageClass	string	必选	桶的 <a href="#">存储类型</a> 。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketStoragePolicy([
        'Bucket' => 'bucketname',
        'StorageClass' => ObsClient::StorageClassWarm
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
}catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.14 获取桶存储类型

### 功能说明

获取桶的存储类型。

### 方法定义

1. ObsClient->getBucketStoragePolicy(array \$parameter)
2. ObsClient->getBucketStoragePolicyAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
StorageClass	string	桶的存储类型。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketStoragePolicy([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("StorageClass:%s\n", $resp['StorageClass']);
}catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.15 设置桶 ACL

### 功能说明

设置桶的访问权限。

### 方法定义

1. ObsClient->setBucketAcl(array \$parameter)
2. ObsClient->setBucketAclAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明		
Bucket	string	必选	桶名。		
ACL	string	可选	<a href="#">预定义访问策略</a> 。		
Owner	associative array	可选	桶的所有者。		
	ID	string	必选	桶所有者的DomainId。	
	DisplayName	string	可选	桶所有者的名字。	
Grants	indexed array	可选	被授权用户权限信息列表。		
	Grantee	associative array	必选	被授权用户。	
		Type	string	必选	被授权的 <a href="#">用户类型</a> 。
		ID	string	如果Type为“Canonical User”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 <a href="#">用户组</a> 。
	Permission	string	必选	被授予的 <a href="#">权限</a> 。	

字段名	类型	约束	说明
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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketAcl([
        'Bucket' => 'bucketname',
        'Owner' => ['ID' => 'ownerid', 'DisplayName' => 'ownername'],
        'Grants' => [
            ['Grantee' => ['Type' => 'CanonicalUser', 'ID' => 'userid'], 'Permission' =>
            ObsClient::PermissionRead],
            ['Grantee' => ['Type' => 'CanonicalUser', 'ID' => 'userid'], 'Permission' =>
            ObsClient::PermissionWrite],
            ['Grantee' => ['Type' => 'Group', 'URI' => ObsClient::GroupLogDelivery], 'Permission' =>
            ObsClient::PermissionWrite],
            ['Grantee' => ['Type' => 'Group', 'URI' => ObsClient::GroupLogDelivery], 'Permission' =>
            ObsClient::PermissionReadAcp],
        ]
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.16 获取桶 ACL

### 功能说明

获取桶的访问权限。

### 方法定义

1. ObsClient->getBucketAcl(array \$parameter)
2. ObsClient->getBucketAclAsync(array \$parameter, callable \$callback)

## 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
Owner	associative array	桶的所有者。
	ID	桶所有者的DomainId。
Grants	indexed array	被授权用户权限信息列表。
	Grantee	被授权用户。
	ID	被授权用户的DomainId，当用户类型是Group时空。
	URI	被授权的用户组，当用户类型是CanonicalUser时空。
	Permission	被授予的权限。
	Delivered	桶内对象ACL是否继承桶的ACL。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketAcl([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("Owner[ID]:%s\n", $resp['Owner']['ID']);
    printf("Owner[DisplayName]:%s\n", $resp['Owner']['DisplayName']);
    printf("Grants\n");
    foreach ($resp['Grants'] as $index => $grant){
        printf("Grants[%d]", $index + 1);
        printf("Grantee[ID]:%s\n", $grant['Grantee']['ID']);
        printf("Grantee[DisplayName]:%s\n", $grant['Grantee']['DisplayName']);
        printf("Grantee[URI]:%s\n", $grant['Grantee']['URI']);
        printf("Permission:%s\n", $grant['Permission']);
    }
}
```

```

}catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
    
```

## 4.17 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

1. ObsClient->setBucketLogging(array \$parameter)
2. ObsClient->setBucketLoggingAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明		
Bucket	string	必选	桶名。		
Agency	string	如果是设置桶日志配置则必选	委托名。		
LoggingEnabled	associative array	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	indexed array	可选	被授权用户权限信息列表。	
		Grantee	associative array	可选	被授权用户。
		Type	string	必选	被授权的 <a href="#">用户类型</a> 。



```

]);
printf("RequestId:%s\n", $resp['RequestId']);
}catch (Obs\Common\ObsException $obsException){
printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}

```

## 4.18 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

1. ObsClient->getBucketLogging(array \$parameter)
2. ObsClient->getBucketLoggingAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
Agency	string	委托名。	
LoggingEnabled	associative array	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	indexed array	被授权用户权限信息列表。
	Grantee	associative array	被授权用户。
		ID	string
URI		string	被授权的用户组, 当用户类型是CanonicalUser时为空。

字段名	类型	说明
Permission	string	被授予的权限。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketLogging([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("LoggingEnabled[TargetBucket]:%s\n", $resp['LoggingEnabled']['TargetBucket']);
    printf("LoggingEnabled[TargetPrefix]:%s\n", $resp['LoggingEnabled']['TargetPrefix']);
    printf("TargetGrants\n");
    foreach ($resp['LoggingEnabled']['TargetGrants'] as $index => $grant){
        printf("Grants[%d]", $index + 1);
        printf("Grantee[ID]:%s\n", $grant['Grantee']['ID']);
        printf("Grantee[DisplayName]:%s\n", $grant['Grantee']['DisplayName']);
        printf("Grantee[URI]:%s\n", $grant['Grantee']['URI']);
        printf("Permission:%s\n", $grant['Permission']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.19 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(array \$parameter)
- ObsClient->setBucketPolicyAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
Policy	string	必选	策略信息，JSON格式的字符串。具体格式请参考 <a href="#">Policy格式</a> 。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。

字段名	类型	说明
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketPolicy([
        'Bucket' => 'bucketname',
        'Policy' => 'your policy'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

### 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。

## 4.20 获取桶策略

### 功能说明

获取桶的策略。

### 方法定义

- ObsClient->getBucketPolicy(array \$parameter)
- ObsClient->getBucketPolicyAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

字段名	类型	说明
Policy	string	策略信息，JSON格式的字符串。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketPolicy([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("Policy:%s\n", $resp['Policy']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.21 删除桶策略

### 功能说明

删除桶的策略。

### 方法定义

- ObsClient->deleteBucketPolicy(array \$parameter)
- ObsClient->deleteBucketPolicyAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> deleteBucketPolicy([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
```

```
printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.22 设置桶的生命周期配置

### 功能说明

配置桶的生命周期规则，实现定时删除桶中对象的功能。

### 方法定义

1. ObsClient->setBucketLifecycle(array \$parameter)
2. ObsClient->setBucketLifecycleAsync(array \$parameter, callable \$callback)

### 请求参数

字段名		类型	约束	说明
Bucket		string	必选	桶名。
Rules		indexed array	必选	桶生命周期规则列表。
	Transitions	indexed array	可选	对象转换策略列表。
	StorageClass	string	必选	对象转换后的 <b>存储类型</b> 。 <b>说明</b> 不支持“标准存储”类型。
	Date	string 或 \DateTime	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	integer	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		associative array	可选	对象过期时间配置。

字段名		类型	约束	说明
	Date	string 或 \DateTime	如果没有设置Days则必选	表示对象过期的日期。该值为字符串时必须兼容ISO8601格式（例如：2018-01-01T00:00:00Z），而且必须是UTC午夜0点。
	Days	integer	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： <ul style="list-style-type: none"> <li>• Enabled</li> <li>• Disabled</li> </ul>
NoncurrentVersionTransitions		indexed array	可选	历史版本对象转换策略列表。
	StorageClasses	string	必选	历史版本对象转换后的 <a href="#">存储类型</a> 。

字段名		类型	约束	说明
	Noncurrent Days	integer	必选	表示对象成为历史版本后第几天时转换，正整数。
	NoncurrentVersionExpiration	associative array	可选	历史版本对象过期时间配置。
	Noncurrent Days	integer	必选	表示对象成为历史版本后第几天时过期，正整数。

### 📖 说明

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

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $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']]
        ]
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.23 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

1. ObsClient->getBucketLifecycle(array \$parameter)
2. ObsClient->getBucketLifecycleAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
Rules	indexed array	桶生命周期规则列表。	
Transitions	indexed array	对象转换策略列表。	
	Storage Class	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	associative array	对象过期时间配置。	
	Date	string	表示对象过期的日期。
	Days	integer	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。	

字段名	类型	说明
NoncurrentVersionTransitions	indexed array	历史版本对象转换策略列表。
Storage Class	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	associative array	历史版本对象过期时间配置。
NoncurrentDays	integer	表示对象成为历史版本后第几天时过期。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketLifecycle([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    foreach ($resp['Rules'] as $index => $rule){
        printf("Rules[%d]\n", $index + 1);
        printf("ID:%s\n", $rule['ID']);
        printf("Prefix:%s\n", $rule['Prefix']);
        printf("Status:%s\n", $rule['Status']);
        printf("Expiration[Days]:%s\n", $rule['Expiration']['Days']);
        printf("Expiration[Date]:%s\n", $rule['Expiration']['Date']);
        printf("NoncurrentVersionExpiration[NoncurrentDays]:%s\n", $rule['NoncurrentVersionExpiration']
['NoncurrentDays']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.24 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(array \$parameter)
- ObsClient->deleteBucketLifecycleAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> deleteBucketLifecycle([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.25 设置桶的 Website 配置

### 功能说明

设置桶的Website配置。

### 方法定义

1. ObsClient->setBucketWebsite(array \$parameter)
2. ObsClient->setBucketWebsiteAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明	
Bucket	string	必选	桶名。	
RedirectAllRequestsTo	associative array	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： <ul style="list-style-type: none"> <li>• http（默认）</li> <li>• https</li> </ul>
ErrorDocument	associative array	可选	错误页面配置。	

字段名		类型	约束	说明
	Key	string	可选	指定当4XX错误出现时返回的页面。
IndexDocument		associative array	可选	默认页面配置。
	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		indexed array	可选	重定向规则列表。
	Condition	associative array	可选	重定向规则的匹配条件。
	HttpErrorC odeReturn edEquals	string	可选	重定向规则生效需要匹配的HTTP错误码。
	KeyPrefixE quals	string	可选	重定向规则生效需要匹配的对象名前缀。
Redirect		associative array	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： <ul style="list-style-type: none"> <li>• http</li> <li>• https</li> </ul>
	HostName	string	可选	重定向请求时使用的站点名。
	ReplaceKey PrefixWith	string	可选	重定向请求时使用的对象名前缀。
	ReplaceKey With	string	可选	重定向请求时使用的对象名。不可与ReplaceKeyPrefixWith同时存在。
	HttpRedire ctCode	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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

### 代码样例

```
try{
    $resp = $obsClient -> setBucketWebsite([
        'Bucket' => 'bucketname',
        'RedirectAllRequestsTo' => ['HostName' => 'www.example.com', 'Protocol' => 'https'],
        'IndexDocument' => ['Suffix' => 'index.html'],
        'ErrorDocument' => ['Key' => 'error.html'],
        'RoutingRules' => [
            ['Condition' => ['HttpErrorCodeReturnedEquals' => 404, 'KeyPrefixEquals' => 'prefix'],
            'Redirect' => ['Protocol' => 'http', 'ReplaceKeyWith' => 'key']],
            ['Condition' => ['HttpErrorCodeReturnedEquals' => 404, 'KeyPrefixEquals' => 'prefix'],
            'Redirect' => ['Protocol' => 'http', 'ReplaceKeyWith' => 'key']]
        ]
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.26 获取桶的 Website 配置

### 功能说明

获取桶的Website配置。

### 方法定义

1. ObsClient->getBucketWebsite(array \$parameter)
2. ObsClient->getBucketWebsiteAsync(array \$parameter, callable \$callback)

## 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
RedirectAllRequestsTo	associative array	所有请求重定向的规则。	
	Host Name	string	重定向时使用的站点名。
	Protocol	string	重定向时使用的协议。
ErrorDocument	associative array	错误页面配置。	
	Key	string	指定当4XX错误出现时返回的页面。
IndexDocument	associative array	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	indexed array	重定向规则列表。	
	Condition	associative array	重定向规则的匹配条件。

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

## 代码样例

```
try{
    $resp = $obsClient -> getBucketWebsite([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("ErrorDocument[Key]:%s\n", $resp['ErrorDocument']['Key']);
    printf("IndexDocument[Suffix]:%s\n", $resp['IndexDocument']['Suffix']);
    foreach ($resp['RoutingRules'] as $index => $routingRule){
        printf("RoutingRules[%d]", $index + 1);
        printf("Condition[HttpErrorReturnedEquals]:%s\n", $routingRule['Condition']
            ['HttpErrorReturnedEquals']);
        printf("Condition[KeyPrefixEquals]:%s\n", $routingRule['Condition']['KeyPrefixEquals']);
        printf("Redirect[Protocol]:%s\n", $routingRule['Redirect']['Protocol']);
        printf("Redirect[ReplaceKeyWith]:%s\n", $routingRule['Redirect']['ReplaceKeyWith']);
        printf("Redirect[HttpRedirectCode]:%s\n", $routingRule['Redirect']['HttpRedirectCode']);
        printf("Redirect[HostName]:%s\n", $routingRule['Redirect']['HostName']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.27 删除桶的 Website 配置

### 功能说明

删除指定桶的Website配置。

## 方法定义

1. ObsClient->deleteBucketWebsite(array \$parameter)
2. ObsClient->deleteBucketWebsiteAsync(array \$parameter, callable \$callback)

## 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> deleteBucketWebsite([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
}catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.28 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

1. ObsClient->setBucketVersioning(array \$parameter)
2. ObsClient->getBucketVersioningAsync(array \$parameter, callable \$callback)

### 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
Status	string	必选	桶的多版本状态，支持的值： <ul style="list-style-type: none"> <li>• Enabled</li> <li>• Suspended</li> </ul>

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketVersioning([
        'Bucket' => 'bucketname',
        'Status' => 'Enabled'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.29 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

1. ObsClient->getBucketVersioning(array \$parameter)
2. ObsClient->getBucketVersioningAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
Status	string	桶的多版本状态。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketVersioning([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("Status:%s\n", $resp['Status']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.30 设置桶的 CORS 配置

### 功能说明

设置桶的跨域资源共享规则，以允许客户端浏览器进行跨域请求。

### 方法定义

- ObsClient->setBucketCors(array \$parameter)
- ObsClient->setBucketCorsAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。
CorsRules	indexed array	必选	桶的CORS规则列表。
	ID	可选	CORS规则ID，由不超过255个字符的字符串组成。

字段名	类型	约束	说明
AllowedMethod	indexed array of strings	必选	CORS规则允许的HTTP方法，支持的值： <ul style="list-style-type: none"> <li>• GET</li> <li>• PUT</li> <li>• HEAD</li> <li>• POST</li> <li>• DELETE</li> </ul>
AllowedOrigin	indexed array of strings	必选	CORS规则允许的请求来源（表示域名的字符串）。可以带一个匹配符“*”，每一个AllowedOrigin最多可以带一个“*”通配符。
AllowedHeader	indexed array of strings	可选	CORS规则允许请求中可携带的头域，不可出现空格。可以带一个匹配符“*”，且每一个AllowedHeader最多可以带一个“*”通配符。
MaxAgeSeconds	integer	可选	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，整数类型。
ExposeHeader	indexed array of strings	可选	CORS规则允许响应中可返回的附加头域，不可出现空格。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketCors([
        'Bucket' => 'bucketname',
        'CorsRules' => [
            [
                'ID' => 'rule1',
                'AllowedMethod' => ['PUT','POST','GET','DELETE','HEAD'],
                'AllowedOrigin' => ['obs.hostname','obs.hostname1'],
                'AllowedHeader' => ['obs-header-1'],
                'MaxAgeSeconds' => 60
            ],
            [

```

```

        'ID' => 'rule2',
        'AllowedMethod' => ['PUT','POST','GET'],
        'AllowedOrigin' => ['obs.hostname','obs.hostname1'],
        'AllowedHeader' => ['header-1','header-2'],
        'MaxAgeSeconds' => 50
    ]
    ];
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}

```

## 4.31 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

1. ObsClient->setBucketCors(array \$parameter)
2. ObsClient->setBucketCorsAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
CorsRules	indexed array	桶的CORS规则列表。	
	ID	string	CORS规则ID。
	AllowedMethod	indexed array of strings	CORS规则允许的HTTP方法。
	AllowedOrigin	indexed array of strings	CORS规则允许的请求来源（表示域名的字符串）。

字段名	类型	说明
AllowedHeader	indexed array of strings	CORS规则允许请求中可携带的头域。
MaxAgeSecond	integer	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	indexed array of strings	CORS规则允许响应中可返回的附加头域。

## 代码样例

```
try{
    $resp = $obsClient -> getBucketCors([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    foreach ($resp['CorsRules'] as $index => $corsRule){
        printf("CorsRule[%d]\n", $index + 1);
        printf("MaxAgeSeconds:%s\n", print_r($corsRule['MaxAgeSeconds'], true));
        printf("AllowedMethod:%s\n", print_r($corsRule['AllowedMethod'], true));
        printf("AllowedOrigin:%s\n", print_r($corsRule['AllowedOrigin'], true));
        printf("AllowedHeader:%s\n", print_r($corsRule['AllowedHeader'], true));
        printf("ExposeHeader:%s\n", print_r($corsRule['ExposeHeader'], true));
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.32 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

1. ObsClient->setBucketCors(array \$parameter)
2. ObsClient->setBucketCorsAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> deleteBucketCors([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.33 设置桶标签

### 功能说明

设置桶的标签。

### 方法定义

1. ObsClient->setBucketTagging(array \$parameter)
2. ObsClient->setBucketTaggingAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明	
Bucket	string	必选	桶名。	
Tags	indexed 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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> setBucketTagging([
        'Bucket' => 'bucketname',
        'Tags' => [
            ['Key' => 'tag1', 'Value' => 'value1'],
            ['Key' => 'tag2', 'Value' => 'value2']
        ]
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.34 获取桶标签

### 功能说明

获取指定桶的标签。

### 方法定义

- ObsClient->getBucketTagging(array \$parameter)
- ObsClient->getBucketTaggingAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

字段名	类型	说明
Tags	indexed 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

```
try{
    $resp = $obsClient -> getBucketTagging([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    foreach ($resp['Tags'] as $index => $tag){
        printf("TagSet[%d]\n", $index + 1);
        printf("Key:%s\n", $tag['Key']);
        printf("Value:%s\n", $tag['Value']);
    }
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 4.35 删除桶标签

### 功能说明

删除指定桶的标签。

### 方法定义

- ObsClient->deleteBucketTagging(array \$parameter)
- ObsClient->deleteBucketTaggingAsync(array \$parameter, callable \$callback)

### 请求参数

字段名	类型	约束	说明
Bucket	string	必选	桶名。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try{
    $resp = $obsClient -> deleteBucketTagging([
        'Bucket' => 'bucketname'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

# 5 对象相关接口

## 5.1 上传对象

### 功能说明

上传单个对象到指定桶中。

### 方法定义

1. ObsClient->putObject(array \$parameter)
2. ObsClient->putObjectAsync(array \$parameter, callable \$callback)

### 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。
ACL	string	可选	创建对象时可指定的 <a href="#">预定义访问策略</a> 。
StorageClass	string	可选	创建对象时可指定的对象的 <a href="#">存储类型</a> 。
Body	string 或 resource 或 GuzzleHttp \Psr7\StreamInterface	可选	待上传对象的内容。
SourceFile	string	可选	待上传对象的源文件路径。

字段名	类型	约束	说明
Metadata	associative array	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。
ContentType	string	可选	待上传对象的MIME类型。
ContentLength	integer	可选	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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
ObjectURL	string	对象的地址。
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。

字段名	类型	说明
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。

## 代码样例

```
try{
    $resp = $obsClient -> putObject([
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'Metadata' => ['meta1' => 'value1', 'meta2' => 'value2'],
//
        'SourceFile' => 'localfile',
        'Body' => 'Hello OBS',
        'ContentType' => 'text/plain'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("VersionId:%s\n", $resp['VersionId']);
    printf("StorageClass:%s\n", $resp['StorageClass']);
    printf("ETag:%s\n", $resp['ETag']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 5.2 下载对象

### 功能说明

下载指定桶中的对象。

### 方法定义

1. ObsClient->getObject(array \$parameter)
2. ObsClient->getObjectAsync(array \$parameter, callable \$callback)

### 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。
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，否则抛出异常。

字段名	类型	约束	说明
IfModifiedSince	string 或 \DateTime	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，否则抛出异常。该参数值为字符串时必须符合 <a href="http://www.ietf.org/rfc/rfc2616.txt">http://www.ietf.org/rfc/rfc2616.txt</a> 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，否则抛出异常。
IfUnmodifiedSince	string 或 \DateTime	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，否则抛出异常。该参数值为字符串时必须符合 <a href="http://www.ietf.org/rfc/rfc2616.txt">http://www.ietf.org/rfc/rfc2616.txt</a> 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。
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。
SaveAsFile	string	可选	下载对象的目标路径，包含文件名。
SaveAsStream	boolean	可选	是否将对象以数据流的形式返回。
FilePath	string	可选	废弃参数，与旧版本保持兼容。下载对象的目标路径，包含文件名。
SseC	string	可选	以SSE-C方式解密对象，支持的值： <ul style="list-style-type: none"> <li>• AES256</li> </ul>

字段名	类型	约束	说明
SseCKey	string	可选	SSE-C方式下解密的密钥，由AES256算法算出。

#### 📖 说明

- 当SaveAsStream为true时不能与SaveAsFile或FilePath一起使用。
- SaveAsFile与FilePath不能一起使用。
- 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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
DeleteMarker	boolean	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	integer	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。
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STANDARD时，该值为空。
Restore	string	归档存储类型对象的取回状态。

字段名	类型	说明
AllowOrigin	string	如果请求中的Origin满足桶的CORS规则，则返回CORS规则中的AllowedOrigin。
AllowHeader	string	如果请求的RequestHeader满足桶的CORS规则，则返回CORS规则中的AllowedHeader。
AllowMethod	string	桶CORS规则中的AllowedMethod。
ExposeHeader	string	桶CORS规则中的ExposeHeader。
MaxAgeSeconds	string	桶CORS规则中的MaxAgeSeconds。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的主密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。
Expiration	string	对象的详细过期信息。
Body	GuzzleHttp\Psr7\Stream	对象的内容。当设置了SaveAsFile时该值为空；当设置了SaveAsStream且为true时该值为可读流，需要调用GuzzleHttp\Psr7\Stream->read方法读取数据。
SaveAsFile	string	下载对象的目标路径，包含文件名，与请求中的该参数对应。
Metadata	associative array	对象自定义元数据。

## 代码样例

```
try{
    $resp = $obsClient -> getObject([
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        // 'SaveAsFile' => 'localfile',
        // 'SaveAsStream' => true,
        'Range' => 'bytes=0-10'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("ETag:%s\n", $resp['ETag']);
    printf("VersionId:%s\n", $resp['VersionId']);
    printf("StorageClass:%s\n", $resp['StorageClass']);
    printf("ContentLength:%s\n", $resp['ContentLength']);
    printf("DeleteMarker:%s\n", $resp['DeleteMarker']);
    printf("LastModified:%s\n", $resp['LastModified']);
    printf("Body:%s\n", $resp['Body']);
    printf("Metadata:%s\n", print_r($resp['Metadata'], true));
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 5.3 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

1. ObsClient->copyObject(array \$parameter)
2. ObsClient->copyObjectAsync(array \$parameter, callable \$callback)

### 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。
ACL	string	可选	复制对象时可指定的 <a href="#">预定义访问策略</a> 。
StorageClass	string	可选	复制时设置对象的 <a href="#">存储类型</a> 。
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，否则抛出异常。
CopySourceIfModifiedSince	string 或 \DateTime	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，否则抛出异常。该参数值为字符串时必须符合 <a href="http://www.ietf.org/rfc/rfc2616.txt">http://www.ietf.org/rfc/rfc2616.txt</a> 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，否则抛出异常。
CopySourceIfUnmodifiedSince	string 或 \DateTime	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，否则抛出异常。该参数值为字符串时必须符合 <a href="http://www.ietf.org/rfc/rfc2616.txt">http://www.ietf.org/rfc/rfc2616.txt</a> 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。

字段名	类型	约束	说明
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	<a href="#">复制策略</a> 。
Metadata	associative array	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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
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。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的主密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。

## 代码样例

```
try{
    $resp = $obsClient -> copyObject([
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'CopySource' => 'srcbucketname/srcobjectkey',
        'Metadata' => ['meta1' => 'value1']
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
    printf("ETag:%s\n", $resp['ETag']);
    printf("VersionId:%s\n", $resp['VersionId']);
    printf("CopySourceVersionId:%s\n", $resp['CopySourceVersionId']);
    printf("LastModified:%s\n", $resp['LastModified']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

## 5.4 删除对象

### 功能说明

删除指定桶中的对象。

### 方法定义

- ObsClient->deleteObject(array \$parameter)
- ObsClient->deleteObject(array \$parameter, callable \$callback)

## 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。
VersionId	string	可选	待删除对象的版本号。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
DeleteMarker	boolean	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

```
try{
    $resp = $obsClient -> deleteObject([
        'Bucket' => 'bucketname',
        'Key' => 'objectkey'
    ]);
    printf("RequestId:%s\n", $resp['RequestId']);
} catch (Obs\Common\ObsException $obsException){
    printf("ExceptionCode:%s\n", $obsException->getExceptionCode());
    printf("ExceptionMessage:%s\n", $obsException->getExceptionMessage());
}
```

# 5.5 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(array \$parameter)
- ObsClient->deleteObjectsAsync(array \$parameter, callable \$callback)

## 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	
Objects	indexed 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

字段名	类型	说明	
HttpStatusCode	integer	HTTP 状态码。	
Reason	string	HTTP 文本描述。	
RequestId	string	OBS 服务端返回的请求 ID。	
Deleted	indexed 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	boolean	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	indexed 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

```
try {
    $resp = $obsClient->deleteObjects ( [
        'Bucket' => 'bucketname',
        'Quiet' => false,
        'Objects' => [
            [
```

```

        'Key' => 'objectkey1',
        'VersionId' => null
    ],
    [
        'Key' => 'objectkey2',
        'VersionId' => null
    ]
]
]);
printf( "RequestId:%s\n", $resp ['RequestId'] );
printf( "Deleted:s\n" );
foreach ( $resp ['Deleted:s'] as $index => $deleted ) {
    printf( "Deleted[%d]", $index + 1 );
    printf( "Key:%s\n", $deleted ['Key'] );
    printf( "VersionId:%s\n", $deleted ['VersionId'] );
    printf( "DeleteMarker:%s\n", $deleted ['DeleteMarker'] );
    printf( "DeleteMarkerVersionId:%s\n", $deleted ['DeleteMarkerVersionId'] );
}
printf( "Errors:\n" );
foreach ( $resp ['Errors'] as $index => $error ) {
    printf( "Errors[%d]", $index + 1 );
    printf( "Key:%s\n", $error ['Key'] );
    printf( "VersionId:%s\n", $error ['VersionId'] );
    printf( "Code:%s\n", $error ['Code'] );
    printf( "Message:%s\n", $error ['Message'] );
}
} catch ( Obs\Common\ObsException $obsException ) {
    printf("StatusCode:%s\n", $obsException->getStatusCode());
}

```

## 5.6 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

1. ObsClient->getObjectMetadata(array \$parameter)
2. ObsClient->getObjectMetadataAsync(array \$parameter, callable \$callback)

### 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。
VersionId	string	可选	对象的版本号。
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。
SseC	string	可选	以SSE-C方式解密对象，支持的值： <ul style="list-style-type: none"> <li>• AES256</li> </ul>

字段名	类型	约束	说明
SseCKey	string	可选	SSE-C方式下解密的密钥，由AES256算法算出。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
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	integer	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取回状态。
AllowOrigin	string	如果请求中的Origin满足桶的CORS规则，则返回CORS规则中的AllowedOrigin。
AllowHeader	string	如果请求的RequestHeader满足桶的CORS规则，则返回CORS规则中的AllowedHeader。
AllowMethod	string	桶CORS规则中的AllowedMethod。
ExposeHeader	string	桶CORS规则中的ExposeHeader。
MaxAgeSeconds	integer	桶CORS规则中的MaxAgeSeconds。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的主密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。
Expiration	string	对象的详细过期信息。

字段名	类型	说明
Metadata	associative array	对象自定义元数据。

## 代码样例

```
try {
    $resp = $obsClient->getObjectMetadata( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey'
    ] );
    printf( "RequestId:%s\n", $resp ['RequestId'] );
    printf( "ETag:%s\n", $resp ['ETag'] );
    printf( "VersionId:%s\n", $resp ['VersionId'] );
    printf( "ContentLength:%s\n", $resp ['ContentLength'] );
    printf( "LastModified:%s\n", $resp ['LastModified'] );
    printf( "Expiration:%s\n", $resp ['Expiration'] );
    printf( "StorageClass:%s\n", $resp ['StorageClass'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

## 5.7 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(array \$parameter)
- ObsClient->setObjectAclAsync(array \$parameter, callable \$callback)

### 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。	
VersionId	string	可选	对象的版本号。	
ACL	string	可选	<a href="#">预定义访问策略</a> 。	
Owner	associative array	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。	

字段名		类型	约束	说明
Grants		indexed 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 <a href="#">用户类型</a> 。
	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 <a href="#">用户组</a> 。
Permission		string	必选	被授予的 <a href="#">权限</a> 。

### 📖 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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	String	OBS服务端返回的请求ID。

## 代码样例

```
try {
    $resp = $obsClient->setObjectAcl([
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'Owner' => ['ID' => 'ownerid'],
        'Grants' => [
```

```

        ['Grantee' => ['Type' => 'CanonicalUser', 'ID' => 'userid'], 'Permission' =>
        ObsClient::PermissionRead],
        ['Grantee' => ['Type' => 'CanonicalUser', 'ID' => 'userid'], 'Permission' =>
        ObsClient::PermissionWriteAcp],
        ['Grantee' => ['Type' => 'Group', 'URI' => ObsClient::GroupAuthenticatedUsers],
        'Permission' => ObsClient::PermissionWriteAcp],
        ['Grantee' => ['Type' => 'Group', 'URI' => ObsClient::GroupAuthenticatedUsers],
        'Permission' => ObsClient::PermissionRead],
    ]
    ];
    printf ( "RequestId:%s\n", $resp ['RequestId'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf ( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf ( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}

```

## 5.8 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

1. ObsClient->getObjectAcl(array \$parameter)
2. ObsClient->getObjectAclAsync(array \$parameter, callable \$callback)

### 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。
VersionId	string	可选	对象的版本号。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
VersionId	string	对象的版本号。
Owner	associative array	对象的所有者。
	ID	对象所有者的DomainId。
Delivered	boolean	桶的ACL是否向桶内对象传递。

字段名		类型	说明
Grants		indexed array	被授权用户权限信息列表。
	Grantee	associative array	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

```
try {
    $resp = $obsClient->getObjectAcl( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey'
    ] );
    printf( "RequestId:%s\n", $resp ['RequestId'] );
    printf("Owner[ID]:%s\n", $resp['Owner']['ID']);
    printf("Grants\n");
    foreach ($resp['Grants'] as $index => $grant){
        printf("Grants[%d]", $index + 1);
        printf("Grantee[ID]:%s\n", $grant['Grantee']['ID']);
        printf("Grantee[URI]:%s\n", $grant['Grantee']['URI']);
        printf("Permission:%s\n", $grant['Permission']);
    }
} catch ( Obs\Common\ObsException $obsException ) {
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

## 5.9 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

1. ObsClient->initiateMultipartUpload(array \$parameter)
2. ObsClient->initiateMultipartUploadAsync(array \$parameter, callable \$callback)

### 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
ACL	string	可选	<a href="#">预定义访问策略</a> 。
StorageClass	string	可选	对象的 <a href="#">存储类型</a> 。
Metadata	associative array	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。
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： <ul style="list-style-type: none"> <li>kms</li> </ul>
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： <ul style="list-style-type: none"> <li>AES256</li> </ul>
SseCKey	string	可选	SSE-C方式下加密的密钥，由AES256算法得到。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
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。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。

## 代码样例

```
try {
    $resp = $obsClient->initiateMultipartUpload( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'ContentType' => 'text/plain'
    ] );
    printf ( "RequestId:%s\n", $resp ['RequestId'] );
    printf ( "Bucket:%s\n", $resp ['Bucket'] );
    printf ( "Key:%s\n", $resp ['Key'] );
    printf ( "UploadId:%s\n", $resp ['UploadId'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf ( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf ( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

## 5.10 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；最后一段的大小范围是0~5GB。上传的段号也有范围限制，其范围是1~10000。

### 方法定义

1. ObsClient->uploadPart(array \$parameter)
2. ObsClient->uploadPartAsync(array \$parameter, callable \$callback)

### 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。
PartNumber	integer	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	待上传段数据的MD5值（经过Base64编码），提供给OBS服务端，校验数据完整性。
Body	string 或 resource 或 GuzzleHttp\Psr7\StreamInterface	可选	待上传段的内容。

字段名	类型	约束	说明
SourceFile	string	可选	待上传段的源文件路径。
Offset	integer	可选	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	integer	可选	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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。
ETag	string	当前上传段的ETag值。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。

## 代码样例

```
try {
    $resp = $obsClient->uploadPart( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'UploadId' => 'uploadid',
        'PartNumber' => 1,
        'Body' => 'Hello OBS'
    ] );
}
```

```

    ]);
    printf( "RequestId:%s\n", $resp ['RequestId'] );
    printf( "ETag:%s\n", $resp ['ETag'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}

```

## 5.11 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

1. ObsClient->copyPart(array \$parameter)
2. ObsClient->copyPartAsync(array \$parameter, callable \$callback)

### 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。
PartNumber	integer	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，仍然取源对象长度-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

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
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。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。

## 代码样例

```
try {
    $resp = $obsClient->copyPart( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'UploadId' => 'uploadid',
        'PartNumber' => 1,
        'CopySource' => 'sourcebucketname/sourceobjectkey',
        'CopySourceRange' => 'bytes=0-10'
    ] );
    printf ( "RequestId:%s\n", $resp ['RequestId'] );
    printf ( "ETag:%s\n", $resp ['ETag'] );
    printf ( "LastModified:%s\n", $resp ['LastModified'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf ( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf ( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

## 5.12 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

1. ObsClient->listParts(array \$parameter)
2. ObsClient->listPartsAsync(array \$parameter, callable \$callback)

### 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
Key	string	必选	对象名。
UploadId	string	必选	分段上传任务的ID。
MaxParts	integer	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	integer	可选	列举已上传段的起始位置，只有Part Number大于该参数的段会被列出。

## 返回结果

字段名	类型	说明	
HttpStatusCode	integer	HTTP状态码。	
Reason	string	HTTP文本描述。	
RequestId	string	OBS服务端返回的请求ID。	
Bucket	string	桶名。	
Key	string	对象名。	
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	boolean	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	indexed array	已上传段列表。	
	PartNumber	integer	段号。
	LastModified	string	段的最后上传时间。
	ETag	string	段的ETag值。
	Size	integer	段的大小。
Initiator	associative array	分段上传任务的创建者。	
	ID	string	创建者的DomainId。

字段名	类型	说明
DisplayNam e	string	创建者的名字。
Owner	associati ve array	和Initiator相同，代表分段上传任务的创建者。
ID	string	创建者的DomainId。
DisplayNam e	string	创建者的名字。
StorageClass	string	待分段上传对象的存储类型。

## 代码样例

```
try {
    $resp = $obsClient->listParts( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'UploadId' => 'uploadid',
        'MaxParts' => 10
    ] );
    printf( "RequestId:%s\n", $resp ['RequestId'] );
    printf( "Initiator[ID]:%s\n", $resp ['Initiator']['ID'] );
    printf( "Initiator[DisplayName]:%s\n", $resp ['Initiator']['DisplayName'] );
    foreach ($resp['Parts'] as $index => $part){
        printf("Parts[%d]\n", $index + 1);
        printf( "PartNumber:%s\n", $part['PartNumber'] );
        printf( "LastModified:%s\n", $part['LastModified'] );
        printf( "ETag:%s\n", $part['ETag'] );
        printf( "Size:%s\n", $part['Size'] );
    }
} catch ( Obs\Common\ObsException $obsException ) {
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

## 5.13 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

1. ObsClient->completeMultipartUpload(array \$parameter)
2. ObsClient->completeMultipartUploadAsync(array \$parameter, callable \$callback)

### 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	
UploadId	string	必选	分段上传任务的ID。	
Parts	indexed array	必选	合并的段列表。	
	PartNumber	integer	必选	段号。
	ETag	string	必选	段的ETag值。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
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。
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。
SseKms	string	SSE-KMS方式的算法。
SseKmsKey	string	SSE-KMS方式的主密钥。
SseC	string	SSE-C方式的算法。
SseCKeyMd5	string	SSE-C方式的密钥的MD5值。

## 代码样例

```
try {
    $resp = $obsClient->completeMultipartUpload( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'UploadId' => 'uploadid',
        'Parts' => [
            ['PartNumber' => 1, 'ETag' => 'etag1'],
            ['PartNumber' => 2, 'ETag' => 'etag2']
        ]
    ] );
    printf ( "RequestId:%s\n", $resp ['RequestId'] );
    printf ( "Bucket:%s\n", $resp ['Bucket'] );
    printf ( "Key:%s\n", $resp ['Key'] );
    printf ( "ETag:%s\n", $resp ['ETag'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf ( "ExceptionCode:%s\n", $obsException->getExceptionCode ( ) );
}
```

```
}  
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage ( ) );  
}
```

## 5.14 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

1. ObsClient->abortMultipartUpload(array \$parameter)
2. ObsClient->abortMultipartUploadAsync(array \$parameter, callable \$callback)

### 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。
UploadId	string	必选	分段上传任务的ID。

### 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	String	OBS服务端返回的请求ID。

### 代码样例

```
try {  
    $resp = $obsClient->abortMultipartUpload( [  
        'Bucket' => 'bucketname',  
        'Key' => 'objectkey',  
        'UploadId' => 'uploadid'  
    ] );  
    printf( "RequestId:%s\n", $resp ['RequestId'] );  
} catch ( Obs\Common\ObsException $obsException ) {  
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode ( ) );  
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage ( ) );  
}
```

## 5.15 取回归档存储对象

### 功能说明

取回指定桶中的归档存储对象。

## 方法定义

1. ObsClient->restoreObject(array \$parameter)
2. ObsClient->restoreObjectAsync(array \$parameter, callable \$callback)

## 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。
VersionId	string	可选	待取回归档存储对象的版本号。
Days	integer	必选	取回对象的保存时间（单位：天），取值范围：[1, 30]。
Tier	string	可选	<a href="#">取回选项</a> 。

## 返回结果

字段名	类型	说明
HttpStatusCode	integer	HTTP状态码。
Reason	string	HTTP文本描述。
RequestId	string	OBS服务端返回的请求ID。

## 代码样例

```
try {
    $resp = $obsClient->restoreObject( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'Days' => 1,
        'Tier' => ObsClient::RestoreTierExpedited
    ] );
    printf( "RequestId:%s\n", $resp ['RequestId'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

# 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->createSignedUrl(array $parameter)
```

### 请求参数

字段名	类型	约束	说明
Method	string	必选	HTTP方法类型，支持的值： <ul style="list-style-type: none"><li>• GET</li><li>• POST</li><li>• PUT</li><li>• DELETE</li><li>• HEAD</li></ul>
Bucket	string	可选	桶名。
Key	string	可选	对象名。

字段名	类型	约束	说明
SpecialParam	string	可选	特殊操作符，代表要操作的子资源，支持的值： <ul style="list-style-type: none"> <li>• versions</li> <li>• uploads</li> <li>• location</li> <li>• storageinfo</li> <li>• quota</li> <li>• storagePolicy</li> <li>• acl</li> <li>• logging</li> <li>• policy</li> <li>• lifecycle</li> <li>• website</li> <li>• versioning</li> <li>• cors</li> <li>• notification</li> <li>• tagging</li> <li>• delete</li> <li>• restore</li> </ul>
Expires	integer	可选	带授权信息的URL的过期时间（单位：秒），默认值：300。
Headers	associative array	可选	请求中携带的头域。
QueryParams	associative array	可选	请求中携带的查询参数。

## 返回结果

字段名	类型	说明
SignedUrl	string	带授权信息的URL。
ActualSignedRequest Headers	associative array	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

```
try {
    // 生成创建桶的带授权信息的URL
    $resp = $obsClient->createSignedUrl([
        'Method' => 'PUT',
```

```
'Bucket' => 'bucketname',
'Expires' => 3600,
]);
printf( "SignedUrl:%s\n", $resp ['SignedUrl'] );
printf( "ActualSignedRequestHeaders:%s\n", print_r($resp ['ActualSignedRequestHeaders'], true) );

// 生成上传对象的带授权信息的URL
$resp = $obsClient->createSignedUrl( [
    'Method' => 'PUT',
    'Bucket' => 'bucketname',
    'Key' => 'objectkey',
    'Expires' => 3600,
    'Headers' => ['content-type' => 'text/plain']
]);
printf( "SignedUrl:%s\n", $resp ['SignedUrl'] );
printf( "ActualSignedRequestHeaders:%s\n", print_r($resp ['ActualSignedRequestHeaders'], true) );

// 生成设置对象ACL的带授权信息的URL
$resp = $obsClient->createSignedUrl( [
    'Method' => 'PUT',
    'Bucket' => 'bucketname',
    'Key' => 'objectkey',
    'Expires' => 3600,
    'SpecialParam' => 'acl',
    'Headers' => ['x-obs-acl' => 'public-read']
]);
printf( "SignedUrl:%s\n", $resp ['SignedUrl'] );
printf( "ActualSignedRequestHeaders:%s\n", print_r($resp ['ActualSignedRequestHeaders'], true) );

// 生成下载对象的带授权信息的URL
$resp = $obsClient->createSignedUrl( [
    'Method' => 'GET',
    'Bucket' => 'bucketname',
    'Key' => 'objectkey',
    'Expires' => 3600
]);
printf( "SignedUrl:%s\n", $resp ['SignedUrl'] );
printf( "ActualSignedRequestHeaders:%s\n", print_r($resp ['ActualSignedRequestHeaders'], true) );

// 生成删除对象的带授权信息的URL
$resp = $obsClient->createSignedUrl( [
    'Method' => 'DELETE',
    'Bucket' => 'bucketname',
    'Key' => 'objectkey',
    'Expires' => 3600
]);
printf( "SignedUrl:%s\n", $resp ['SignedUrl'] );
printf( "ActualSignedRequestHeaders:%s\n", print_r($resp ['ActualSignedRequestHeaders'], true) );

// 生成删除桶的带授权信息的URL
$resp = $obsClient->createSignedUrl( [
    'Method' => 'DELETE',
    'Bucket' => 'bucketname',
    'Expires' => 3600
]);
printf( "SignedUrl:%s\n", $resp ['SignedUrl'] );
printf( "ActualSignedRequestHeaders:%s\n", print_r($resp ['ActualSignedRequestHeaders'], true) );
} catch ( Obs\Common\ObsException $obsException ) {
    printf( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

## 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->createPostSignature(array $parameter)
```

### 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	integer	可选	表单上传鉴权参数的过期时间（单位：秒），默认值300。
FormParams	associative array	可选	除key、policy、signature外，表单上传时的其他参数，支持的值： <ul style="list-style-type: none"><li>• acl</li><li>• cache-control</li><li>• content-type</li><li>• content-disposition</li><li>• content-encoding</li><li>• expires</li></ul>

### 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

```
try {
    $resp = $obsClient->createPostSignature( [
        'Bucket' => 'bucketname',
        'Key' => 'objectkey',
        'Expires' => 3600,
        'FormParams' => [
            'acl' => 'public-read',
            'content-type' => 'text/plain',
        ]
    ] );
    printf ( "Policy:%s\n", $resp ['Policy'] );
    printf ( "Signature:%s\n", $resp ['Signature'] );
} catch ( Obs\Common\ObsException $obsException ) {
    printf ( "ExceptionCode:%s\n", $obsException->getExceptionCode () );
    printf ( "ExceptionMessage:%s\n", $obsException->getExceptionMessage () );
}
```

# A 修订记录

---

发布日期	修订记录
2020-03-31	第一次正式发布。