ZOS对象存储PHP_SDK使用手册

环境配置

SDK安装

环境要求

对象存储ZOS的PHP SDK推荐使用PHP版本`7.2.5`及以上版本。

SDK下载

- SDK安装包快速下载地址: 见帮助中心

安装方式

对象存储ZOS的PHP SDK提供通过ZIP包和phar包两种方式安装，具体安装方式如下：

1. 通过ZIP包安装
   将SDK压缩包解压到项目目录下，在项目文件中引入`aws-autoloader.php`文件即可使用PHP SDK。

   ```
   <?php
       require '/path/to/aws-autoloader.php';
   >
   ```

2. 通过phar包安装
   下载SDK的phar包，在项目文件中引入`aws.phar`文件即可使用PHP SDK。

   ```
   <?php
       require '/path/to/aws.phar';
   >
   ```

连接ZOS

获取访问凭证

`AccessKey`、`SecretAccessKey`和`Endpoint`是调用ZOS服务必要的三个参数，具体获取方式如下：

- `AccessKey`和`SecretAccessKey`：可在控制台查看或通过`OpenAPI`的get-keys接口查询。
- `Endpoint`：可在控制台查看或通过`OpenAPI`的get-endpoint接口查询。

创建客户端

使用对象存储ZOS的PHP SDK，首先需要创建一个ZOS客户端，创建客户端时需要传入`AccessKey`、`SecretAccessKey`和`Endpoint`，具体代码如下：

```
<?php
    // require '/path/to/aws-autoloader.php';
    require './aws/aws-autoloader.php';

    use Aws\S3\S3Client;
```

```php
    $accessKey = 'your-access-key';
    $secretAccessKey = 'your-secret-access-key';
    $endpoint = 'your-endpoint';

    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
                'key'    => $accessKey,
                'secret' => $secretAccessKey,
            ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => $endpoint,
    ]);
?>
```

桶操作

创建桶

功能介绍

在指定账号下创建一个新的桶。

方法原型

```php
function createBucket(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
ACL	string	配置创建桶预定义的标准ACL信息，例如 `private`、`public-read` 等	否
Bucket	string	创建桶的名称	是
AZPolicy	string	桶的数据冗余策略，可选值为 `single-az` 和 `multi-az`，分别表示单 AZ 存储和多 AZ 存储。默认为单 AZ 存储	否
StorageClass	string	桶的存储类型，可选值为 `STANDARD`、`STANDARD_IA` 和 `GLACIER`，分别表示标准、低频、归档。默认为标准存储	否
ObjectLockEnabledForBucket	string	是否开启对象锁，可选值为 `true` 和 `false`，默认为 `false`	否

返回结果及说明


```
{
    "Location": "",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket\/",
        "headers": {
            "x-amz-request-id": "tx000000000000000000001a-0067160f8a-fa6d-
default",
            "content-length": "0",
            "date": "Mon, 21 Oct 2024 08:23:38 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
Location	string	桶的位置
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';

//Create a bucket
try {
    $result = $s3Client->createBucket([
        'Bucket' => $bucket
    ]);
```

```
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取桶列表

功能介绍

查询请求用户拥有的所有桶的列表。

方法原型

```
function listBuckets()
```

参数说明

无

返回结果及说明

```
{
    "Buckets": [
        {
            "Name": "bucket1",
            "CreationDate": "2024-10-22T03:05:25+00:00"
        },
        {
            "Name": "bucket2",
            "CreationDate": "2024-10-22T03:05:27+00:00"
        }
    ],
    "Owner": {
        "DisplayName": "your_name",
        "ID": "your_id"
    },
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint",
        "headers": {
            "transfer-encoding": "chunked",
            "x-amz-request-id": "tx0000000000000000000007-0067171694-1217d-
default",
            "content-type": "application\/xml",
            "date": "Tue, 22 Oct 2024 03:05:56 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
```

```
            ]
        }
    }
}
```

返回结果	类型	说明
Buckets	Bucket	桶信息数组，包含了每个桶的名字与创建时间
Owner	Owner	桶的拥有者信息
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

//list buckets
try {
    $result = $s3Client->listBuckets();
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取桶信息

功能介绍

查询用户账号下某个桶是否存在并且用户是否有权访问。

方法原型

```
function headBucket(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "ObjectCount": "1",
    "BytesUsed": 26214400,
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint/bucket/",
        "headers": {
            "x-rgw-object-count": "1",
            "x-rgw-bytes-used": "26214400",
            "x-amz-request-id": "tx000000000000000000016-00671722d1-1217d-
default",
            "content-length": "0",
            "date": "Tue, 22 Oct 2024 03:58:09 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
ObjectCount	Int	Bucket 内实际存在的文件数量，非可见的文件数量，和 Bucket 多版本相关，参考 Put Bucket Versioning，有些文件被删除的时候并没有真正删除，主要用于数据恢复
BytesUsed	Long	当前 Bucket 文件占据的实际空间，单位字节
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);
$bucket = 'your_bucket_name';
//head bucket
try {
    $result = $s3Client->headBucket(
        'Bucket' => $bucket
    );
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

删除桶

功能介绍

删除指定桶。删除一个桶前，需要先删除该桶中的全部对象。

方法原型

```php
function deleteBucket(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 204,
        "effectiveUri": "http:\/\/endpoint\/bucket\/",
        "headers": {
            "x-amz-request-id": "tx000000000000000000001d-006716164b-fa6d-
default",
            "date": "Mon, 21 Oct 2024 08:52:27 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';

//Delete a bucket
try {
    $result = $s3Client->deleteBucket([
        'Bucket' => $bucket
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
```

```php
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

列出桶中对象

功能介绍

列举出桶中的对象。

方法原型

```php
function listObjects(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Delimiter	string	一个分隔符，用于分组键	否
MaxKeys	integer	限制响应中返回的最大项目数。默认值为 1000，最大值为 1000	否
Marker	string	指定开始列出对象的分隔 `key` （按字典序）	否
Prefix	string	返回以指定前缀开头的 `key`	否

返回结果及说明

```
{
    "IsTruncated": false,
    "Marker": "",
    "NextMarker": "next_key",
    "Contents": [
        {
            "Key": "key",
            "LastModified": "2024-10-28T02:19:45+00:00",
            "ETag": "\"ae6c609a6f1f602a72de860f1d97abcc-1\"",
            "Size": "6",
            "StorageClass": "GLACIER",
            "Owner": {
                "DisplayName": "user",
                "ID": "id"
            }
        },
        {
            "Key": "key",
            "LastModified": "2024-10-25T02:45:12+00:00",
            "ETag": "\"5a8dd3ad0756a93ded72b823b19dd877\"",
            "Size": "6",
```

```
            "StorageClass": "STANDARD",
            "Owner": {
                "DisplayName": "user",
                "ID": "id"
            }
        }
    ],
    "Name": "bucket",
    "Prefix": "1",
    "Delimiter": "\/",
    "MaxKeys": 2,
    "EncodingType": "url",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket\/?max-
keys=2&delimiter=%2F&prefix=1&encoding-type=url",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Tue, 29 Oct 2024 02:29:38 GMT",
            "content-type": "application\/xml",
            "transfer-encoding": "chunked",
            "connection": "keep-alive",
            "vary": "Accept-Encoding",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
CommonPrefixes	dict	当请求中设置了 `Delimiter` 和 `Prefix` 属性时，所有包含指定 `Prefix` 且第一次出现 `Delimiter` 字符的对象 `key` 作为一组
Contents	array	对象的元数据
Delimiter	string	与请求中设置的 `Delimiter` 相同
IsTruncated	bool	表示响应结果是否被截断
Marker	string	与请求中设置的 `Marker` 相同
MaxKeys	int	返回结果的对象数量的最大值
Name	string	执行本操作的桶名称
NextMarker	string	当响应被截断时（`IsTruncated` 为 true），可以在下次查询请求时用该字段作为 `Marker` 来获取下一组对象
Prefix	string	与请求中设置的 `Prefix` 一致

代码示例

```php
<?php
require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';

    //list objects
    $result = $s3Client->listObjects([
        'Bucket' => $bucket,
        'MaxKeys' => 100
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
```

```
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

列出对象版本信息

功能介绍

获取对象版本信息（列出指定桶中的全部对象的版本信息，该操作最多返回1000个对象信息，可以通过设置过滤条件来列出桶中符合特定条件的对象）。

方法原型

```
function listObjectVersions(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Delimiter	string	若设置了 `Prefix`，所有包含指定 `Prefix` 且第一次出现 `Delimiter` 字符的对象 `key` 作为一组。若未指定 `Prefix` 参数，按 `Delimiter` 对所有对象 `key` 进行分割	否
KeyMarker	integer	指示从哪里开始列出， ZOS 会在这个指定的键之后开始列出，即列出名称大于 `keyMarker` 的对象。标记可以是桶中的任何键。默认值为空字符串 ""	否
MaxKeys	int	响应中返回的对象 `key` 的数量，最大值和默认值均为1000	否
Prefix	string	返回的 key 的前缀，注意 `key` 与 `Prefix` 完全相同的对象不会返回。默认值为空字符串 ""	否
VersionIdMarker	string	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	否

返回结果及说明

```
{
    "RequestCharged": "",
    "IsTruncated": false,
    "KeyMarker": "",
    "VersionIdMarker": "",
    "Versions": [
```

```
        {
            "ETag": "\"c81e728d9d4c2f636f067f89cc14862c\"",
            "Size": "1",
            "StorageClass": "STANDARD",
            "Key": "key",
            "VersionId": "p5SfjRIBS2ECoQNoiwNmqV3dwOgAdNe",
            "IsLatest": true,
            "LastModified": "2024-12-16T04:06:32+00:00",
            "Owner": {
                "DisplayName": "user_name",
                "ID": "id"
            }
        },
        {
            "ETag": "\"c4ca4238a0b923820dcc509a6f75849b\"",
            "Size": "1",
            "StorageClass": "STANDARD",
            "Key": "key",
            "VersionId": "byEfw6-kbXxS4jkjAEDxk9nCjEC94DL",
            "IsLatest": false,
            "LastModified": "2024-12-16T04:06:14+00:00",
            "Owner": {
                "DisplayName": "user_name",
                "ID": "id"
            }
        },
        {
            "ETag": "\"d63a718444eb4cc3dc10652720832c77\"",
            "Size": "22",
            "StorageClass": "STANDARD",
            "Key": "key",
            "VersionId": "FBXlZSilz03GubBazZU0rpMJonXMtS4",
            "IsLatest": true,
            "LastModified": "2024-12-16T04:10:19+00:00",
            "Owner": {
                "DisplayName": "user_name",
                "ID": "id"
            }
        }
    ],
    "Name": "bucket",
    "Prefix": "",
    "MaxKeys": 1000,
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket?versions",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Mon, 16 Dec 2024 04:10:22 GMT",
            "content-type": "application\/xml",
            "transfer-encoding": "chunked",
            "connection": "keep-alive",
            "vary": "Accept-Encoding",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
```

```
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
CommonPrefixes	array	当请求中设置了 Delimiter 和 Prefix 属性时，所有包含指定 Prefix 且第一次出现Delimiter 字符的对象 key 作为一组
DeleteMarkers	array	对象删除标记数组，每一项包含了是否为最新版本，对象 key ， 最新修改时间，拥有者和版本号等信息
Versions	array	对象版本信息数组
EncodingType	string	返回对象 key 的字符编码类型
Delimiter	string	与请求中设置的 `Delimiter` 相同
IsTruncated	bool	表示响应结果是否被截断
KeyMarker	string	与请求中设置的 KeyMarker 相同
MaxKeys	int	返回结果的对象数量的最大值
Name	string	执行本操作的桶名称
NextKeyMarker	string	当响应被截断时（ IsTruncated 为true ），可以在下次查询请求时用该字段作为 KeyMarker 来获取下一部分的对象版本信息
NextVersionIdMarker	string	本次查询返回的最后一个对象的版本号
Prefix	string	与请求中设置的 `Prefix` 一致
VersionId	string	版本号
IsLatest	bool	是否是最新版本
Owner	array	所有者信息
DisplayName	string	所有者名称
ID	string	所有者id

代码示例

```php
<?php
require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;
```

```php
try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';

    //list Object Versions
    $result = $s3Client->listObjectVersions([
        'Bucket' => $bucket,
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置桶标签

功能介绍

设置桶标签。

方法原型

```php
function putBucketTagging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Tagging	Tagging	封装标签键值对的数组	是

`Tagging` 属性表

属性名	类型	说明	是否必要
TagSet	Tag	标签集合	是

`Tag` 属性表

属性名	类型	说明	是否必要
Key	string	键	是
Value	string	值	是

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket?tagging",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Thu, 24 Oct 2024 09:00:09 GMT",
            "content-type": "application\/xml",
            "content-length": "0",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功

代码示例

```
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
```

```
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $tag = [['Key' => 'key1', 'Value' => 'value1'],['Key' => 'key2', 'Value' => 'value2']]; // This is an example

    //Set the Tagging for the bucket
    $result = $s3Client->putBucketTagging([
            'Bucket' => $bucket,
            'Tagging' => [
                'TagSet' => $tag //You can set one or more in the form of two-dimensional arrays.
            ]
        ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取桶标签

功能介绍

获取桶标签。

方法原型

```
function getBucketTagging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "TagSet": [
        {
            "Key": "key1",
            "Value": "value1"
        },
        {
            "Key": "key2",
            "Value": "value2"
        }
    ],
```

```
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket?tagging",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 25 Oct 2024 01:05:17 GMT",
            "content-type": "application\/xml",
            "content-length": "216",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
TagSet	Tag	标签集合

`Tag` 属性表

属性名	类型	说明
Key	string	键
Value	string	值

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
```

```
    ]);

    $bucket = 'your_bucket';

    //get the tagging of the bucket
    $result = $s3Client->getBucketTagging([
        'Bucket' => $bucket,
    ]);
    echo $result;
} catch (S3Exception $e) {
    if ($e->getStatusCode() == 404) {
        return []; //This bucket has no tagging.
    }
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

删除桶标签

功能介绍

删除桶标签。

方法原型

```
function deleteBucketTagging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket?tagging",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 25 Oct 2024 01:19:58 GMT",
            "content-type": "application\/xml",
            "content-length": "0",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
```

```
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';

    //Delete the Tagging of the bucket
    $result = $s3Client->deleteBucketTagging([
        'Bucket' => $bucket
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置日志转存配置

功能介绍

设置日志转存配置。

方法原型

```
function putBucketLogging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
TargetBucket	string	用来存放日志文件的目标存储桶的名称	是
TargetPrefix	string	指定目标存储桶中日志文件的前缀	否
TargetGrants	array	指定目标存储桶中日志文件的访问权限	否
Grantee	array	描述权限授予的对象	是
DisplayName	string	描述被授予对象的显示名称，通常用于标识用户	否
ID	string	描述被授予对象的唯一标识符	是
Type	string	描述被授予对象的类型，可以是 `CanonicalUser`、`AmazonCustomerByEmail` 或 `Group`	是
Permission	string	描述授予的权限，可以是 READ 或 WRITE	是

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket?logging",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 25 Oct 2024 01:38:12 GMT",
            "content-type": "application\/xml",
            "content-length": "0",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $targetBucket = 'your_log_bucket';
    $targetPrefix = 'logs/';
    $displayName = 'example-grantee-displayname';
    $id = 'example-grantee-id';
    $type = 'CanonicalUser';
    $permission = 'READ';

    //Set log transfer configuration
    $result = $s3Client->putBucketLogging([
        'Bucket' => $bucket,
        'BucketLoggingStatus' => [
            'LoggingEnabled' => [
                'TargetBucket' => $targetBucket,
                'TargetPrefix' => $targetPrefix,
                'TargetGrants' => [
                    [
                        'Grantee' => [
                            'DisplayName' => $displayName,
                            'ID' => $id,
                            'Type' => $type
                        ],
                        'Permission' => $permission
                    ]
                ]
            ]
        ]
    ]);
    echo $result;
```

```
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取日志转存配置

功能介绍

获取日志转存配置。

方法原型

```
function getBucketLogging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明


```
{
    "LoggingEnabled": {
        "TargetBucket": "log_bucket",
        "TargetGrants": [
            {
                "Grantee": {
                    "DisplayName": "example-grantee-displayname",
                    "ID": "example-grantee-id",
                    "Type": "CanonicalUser"
                },
                "Permission": "READ"
            }
        ],
        "TargetPrefix": "logs\/"
    },
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket?logging",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 25 Oct 2024 01:57:38 GMT",
            "content-type": "application\/xml",
            "content-length": "498",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
```

```
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
LoggingEnabled	dict	日志配置相关信息

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';

    //Get log transfer configuration
    $result = $s3Client->getBucketLogging([
        'Bucket' => $bucket
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置版本控制

功能介绍

设置桶的版本控制状态。启用或者暂停 Bucket 的版本控制功能。

方法原型

```
function putBucketVersioning(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
VersioningConfiguration	VersioningConfiguration	设置版本控制状态的参数	是

`VersioningConfiguration` 属性表

属性名	类型	说明	是否必要
Status	string	桶的版本控制设置状态，可选值为 `Enabled` 和 `Suspended`	是

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint/bucket?versioning",
        "headers": {
            "x-amz-request-id": "tx00000000000000000008-0067185f9e-371d-
default",
            "content-type": "application\/xml",
            "content-length": "0",
            "date": "Wed, 23 Oct 2024 02:29:50 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';

try {
    $result = $s3Client->putBucketVersioning([
        'Bucket' => $bucket,
        'VersioningConfiguration' => [
            'Status' => 'Enabled',
        ]
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取版本控制信息

功能介绍

获取桶的版本控制状态信息，只有桶的拥有者才能获取到桶的版本控制信息。

桶的版本控制有三种状态：未开启/开启（Enabled）/暂停（Suspended）。桶在被设置版本状态前默认为未开启状态，此时调用本接口将无法获得任何信息。

方法原型

```
function getBucketVersioning(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "Status": "Enabled",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint\/test-bucket?versioning",
        "headers": {
            "x-amz-request-id": "tx00000000000000000009-00671861d0-371d-
default",
            "content-type": "application\/xml",
            "content-length": "192",
            "date": "Wed, 23 Oct 2024 02:39:12 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
Status	string	桶的版本控制设置状态，可选值为 `Enabled` 和 `Suspended`
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);
```

```php
$bucket = 'your_bucket';

try {
    $result = $s3Client->getBucketVersioning([
        'Bucket' => $bucket
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置桶访问权限

功能介绍

设置桶的访问权限。

方法原型

```php
function putBucketAcl(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
ACL	string	指定Bucket的访问权限ACL，取值范围：`private`/`public-read`/`pulic-read-write`/`authenticated-read`	该方式下必须，其它模式不能使用
AccessControlPolicy	dict	包含多个授权列表和一个桶所有者参数	该方式下必须，其它模式不能使用
Grants	list	授权列表	该方式下必须
Grantee	dict	被授权用户	该方式下必须
Type	string	被授权用户类型，取值范围：`CanonicalUser`/`AmazonCustomerByEmail`/`Group`	该方式下必须
ID(Grantee)	string	被授权用户ID	Type 为`CanonicalUser`，则该字段必须
EmailAddress	string	被授权用户邮箱	Type 为`AmazonCustomerByEmail`，则该字段必须

URI	string	被授权组URI，取值范围为
		所有用户：
		`http://acs.amazonaws.com/groups/global/AllUsers`
		所有认证用户：
		`http://acs.amazonaws.com/groups/global/AuthenticatedUsers`
			如果Type为`Group`，则该字段必须

Permission	string	向被授权用户授予的权限，取值范围：`FULL_CONTROL`/`WRITE`/`WRITE_ACP`/`READ`/`READ_ACP`	该方式下必须
Owner	dict	Bucket 所有者	该方式下必须
ID(Owner)	string	Bucket 所有者ID	该方式下必须
GrantFullControl	string	被授权用户可以对桶进行read, write, read ACP, and write ACP 操作 以下Grant*参数，格式都是"id=xxxx"或"emailAddress=xxxx"或者"uri=xxxx"以及他们的组合（用>逗号连接）	否
GrantRead	string	被授权用户可以对桶进行读操作，即list object	否
GrantWrite	string	被授权用户可以对桶进行写操作，创建新的对象，删除或覆盖写属于自己的对象	否
GrantReadACP	string	被授权用户可以读取桶的ACL	否
GrantWriteACP	string	被授权用户可以修改桶的ACL	否

返回结果及说明


```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket?acl",
        "headers": {
            "x-amz-request-id": "tx000000000000000000010-006719ad1f-d359-
default",
            "content-type": "application\/xml",
            "content-length": "0",
            "date": "Thu, 24 Oct 2024 02:12:47 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';

//putBucketAcl
try {
    $result = $s3Client->putBucketAcl([
        'Bucket' => $bucket,
        //1.ACL设置
        'ACL' => 'public-read',
        /* //2.AccessControlPolicy设置
        'AccessControlPolicy' => [
        'Grants' => [
            [
                'Grantee' => [
                    'ID' => 'admin_test',
                    'Type' => 'CanonicalUser', // REQUIRED
                ],
                'Permission' => 'FULL_CONTROL',
            ],
            [
                'Grantee' => [
                    'ID' => 'admin_test',
                    'Type' => 'CanonicalUser', // REQUIRED
                    'URI' => 'http://acs.amazonaws.com/groups/global/AllUsers'
                ],
                'Permission' => 'READ'
            ]
        ],
        'Owner' => [
            'ID' => 'admin_test',
        ],
    ]
    //3.Grant*设置
    'GrantWrite' => 'id=admin_test'
    */
    ]);
    echo $result;
} catch (S3Exception $e) {
```

```
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取桶访问权限

功能介绍

获取桶的访问权限设置。

方法原型

```
function getBucketAcl(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "Owner": {
        "DisplayName": "admin_test",
        "ID": "admin_test"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "admin_test",
                "EmailAddress": "admin_test@admin.com",
                "ID": "admin_test",
                "Type": "CanonicalUser"
            },
            "Permission": "WRITE"
        }
    ],
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket?acl",
        "headers": {
            "x-amz-request-id": "tx000000000000000000001c-00671a088a-d359-
default",
            "content-type": "application\/xml",
            "content-length": "487",
            "date": "Thu, 24 Oct 2024 08:42:50 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
```

```
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
Owner	dict	Bucket 所有者
DisplayName(Owner)	string	Bucket 所有者的展示名
ID(Owner)	string	Bucket 所有者的用户ID
Grants	list	授权列表
Grantee	dict	被授权用户
Type	string	被授权用户类型
URI	string	被授权用户URI
ID(Grantee)	string	被授权用户ID
DisplayName(Grantee)	string	被授权用户展示名
EmailAddress	string	被授权用户邮箱
Permission	string	被授权权限
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';

//getBucketAcl
try {
    $result = $s3Client->getBucketAcl([
```

```
        'Bucket' => $bucket,
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置桶的对象锁定配置

功能介绍

在指定的存储桶上增加对象锁定配置。默认规则将会应用到每一个新放入桶中的对象。**必须在创建桶时设置开启对象锁，不然后续无法进行配置。**

方法原型

```
function putObjectLockConfiguration(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
ObjectLockConfiguration	ObjectLockConfiguration	应用到指定存储桶上的对象锁定配置	否

`ObjectLockConfiguration` 属性表

属性名	类型	说明	是否必要
ObjectLockEnabled	string	指定桶的对象锁定功能是否生效。当对存储桶设置了对象锁定配置，即为生效	否
Rule	Rule	指定桶的对象锁定配置规则	否

`Rule` 属性表

属性名	类型	说明	是否必要
DefaultRetention	DefaultRetention	指定桶的对象锁定配置规则。配置需要指定模式和时间。年或日只能指定一个，不能在配置中同指定年和日	否

`DefaultRetention` 属性表

属性名	类型	说明	是否必要
Mode	string	指定对象锁定配置的保留模式。有效值：COMPLIANCE、GOVERNANCE	否
Days	int	指定对象锁定配置的保留时间	否
Years	int	指定对象锁定配置的保留时间	否

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint/bucket?object-lock",
        "headers": {
            "x-amz-request-id": "tx000000000000000000016-00671b818e-853d-
default",
            "content-length": "0",
            "date": "Fri, 25 Oct 2024 11:31:26 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
```

```
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';

try {
    $result = $s3Client->putObjectLockConfiguration([
        'Bucket' => $bucket,
        'ObjectLockConfiguration' => [
            'ObjectLockEnabled' => 'Enabled',
            'Rule' => [
                'DefaultRetention' => [
                    'Days' => 30,
                    // 'Years' => 1,
                    'Mode' => 'GOVERNANCE',
                ],
            ],
        ]
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取桶的对象锁定配置

功能介绍

获取存储桶的对象锁定配置。

方法原型

```
function getObjectLockConfiguration(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是

返回结果及说明

```
{
    "ObjectLockConfiguration": {
        "ObjectLockEnabled": "Enabled",
        "Rule": {
            "DefaultRetention": {
                "Mode": "GOVERNANCE",
                "Days": 30
```

```
            }
        }
    },
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint/bucket?object-lock",
        "headers": {
            "x-amz-request-id": "tx00000000000000000017-00671b83bc-853d-
default",
            "content-type": "application\/xml",
            "content-length": "223",
            "date": "Fri, 25 Oct 2024 11:40:44 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
ObjectLockConfiguration	ObjectLockConfiguration	应用到指定存储桶上的对象锁定配置
@metadata	dict	http 请求返回数据

`ObjectLockConfiguration` 属性表

属性名	类型	说明
ObjectLockEnabled	string	指定桶的对象锁定功能是否生效。当对存储桶设置了对象锁定配置，即为生效
Rule	Rule	指定桶的对象锁定配置规则

`Rule` 属性表

属性名	类型	说明
DefaultRetention	DefaultRetention	指定桶的对象锁定配置规则。配置需要指定模式和时间。年或日只能指定一个，不能在配置中同指定年和日

`DefaultRetention` 属性表

属性名	类型	说明
Mode	string	指定对象锁定配置的保留模式。有效值：COMPLIANCE、GOVERNANCE
Days	int	指定对象锁定配置的保留时间
Years	int	指定对象锁定配置的保留时间

代码示例

```php
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';

try {
    $result = $s3Client->getObjectLockConfiguration([
        'Bucket' => $bucket
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

对象操作

上传对象

功能介绍

将一个对象上传到指定桶。

方法原型

```php
function putObject(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的key	是
ACL	string	指定Object的访问权限，空值默认为`private`	否
Body	bytes	上传对象的数据	否
SourceFile	string	上传文件路径，可替代Body参数	否
AddContentMD5	boolean	计算上传数据的MD5值	否
StorageClass	string	上传文件类型，取值范围：`STANDARD` / `STANDARD_IA` / `GLACIER`	否
Metadata	dict	对象元数据	否

返回结果及说明


```
{
    "Expiration": "",
    "ETag": "\"c8485d15ebf9994e09eb31e6e988563f\"",
    "VersionId": "d7j7MMcnVknjcO3ok47m539gQqcD1vB",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket\/key",
        "headers": {
            "content-length": "0",
            "etag": "\"c8485d15ebf9994e09eb31e6e988563f\"",
            "accept-ranges": "bytes",
            "x-amz-version-id": "d7j7MMcnVknjcO3ok47m539gQqcD1vB",
            "x-amz-request-id": "tx000000000000000006b-00671b419d-d359-
default",
            "date": "Fri, 25 Oct 2024 06:58:37 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    },
    "ObjectURL": "http:\/\/endpoint\/bucket\/key"
}
```


返回结果	类型	说明
ETag	string	Object 的ETag
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

try {
    $filename = 'your_file_name';
    $file = fopen($filename, 'r');
    if ($file) {
        $content = fread($file, filesize($filename));
        fclose($file);
        echo $content;
    }

    $result = $s3Client->putObject([
        'Bucket' => $bucket,
        'Key' => $key,
        'Body' => $content,
        /*'Metadata' => [
        'metadata1' => 'value1',
        'metadata2' => 'value2',
    ],*/
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

下载对象

功能介绍

将一个对象下载至本地。该操作需要对目标对象具有读权限或目标对象对所有人都开放了读权限（公有读）。

方法原型

```
function getObject(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的Key	是
VersionId	string	对象的某个特定版本的版本号，没有该版本则返回404错误	否

返回结果及说明

```
{
    "Body": {},
    "LastModified": "2024-10-25T06:58:37+00:00",
    "ContentLength": 12101,
    "ETag": "\"c8485d15ebf9994e09eb31e6e988563f\"",
    "VersionId": "d7j7MMcnVknjcO3ok47m539gQqcD1vB",
    "ContentRange": "",
    "Metadata": [],
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket\/key",
        "headers": {
            "content-length": "12101",
            "accept-ranges": "bytes",
            "last-modified": "Fri, 25 Oct 2024 06:58:37 GMT",
            "x-amz-version-id": "d7j7MMcnVknjcO3ok47m539gQqcD1vB",
            "x-rgw-object-type": "Normal",
            "etag": "\"c8485d15ebf9994e09eb31e6e988563f\"",
            "x-amz-storage-class": "STANDARD",
            "x-amz-request-id": "tx000000000000000006d-00671b462d-d359-
default",
            "content-type": "application\/octet-stream",
            "date": "Fri, 25 Oct 2024 07:18:05 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
Body	streaming body	返回对象的内容
LastModified	datetime	对象的创建时间
ContentLength	integer	响应体的长度
ETag	string	对象的ETag
VersionId	string	对象的VersionId
ContentRange	String	指明对象在返回的响应中包含的部分
Metadata	dict	和对象一起存储的元数据
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

//getObject
try {
    $result = $s3Client->getObject([
        'Bucket' => $bucket,
        'Key' => $key,
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

删除对象

功能介绍

删除对象。

方法原型

```
function deleteObject(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的名称	是
VersionId	string	指定要删除的对象的版本号	否

返回结果及说明

```
{
    "DeleteMarker": false,
    "VersionId": "",
    "@metadata": {
        "statusCode": 204,
        "effectiveUri": "https:\/\/endpoint\/bucket\/key",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 25 Oct 2024 02:08:35 GMT",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，204代表请求成功
DeleteMarker	bool	桶开启版本控制下，未指定版本号将得到该值为 true 的返回值
VersionId	string	删除标记所在的版本号

代码示例

```
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';

    //delete an object
    $result = $s3Client->deleteObject([
        'Bucket' => $bucket,
        'Key'  => $key
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

批量删除对象

功能介绍

批量删除对象。

方法原型

```
function deleteObjects(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Delete	array	封装了要删除的的对象和返回模式的属性	是
Objects	array	要删除的对象列表	是
Key	string	要删除的对象名称	是
VersionId	string	要删除的对象的版本号	否

返回结果及说明


```
{
    "Deleted": [
        {
            "Key": "key1"
        },
        {
            "Key": "key2"
        }
    ],
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoing\/bucket?delete",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 25 Oct 2024 02:33:18 GMT",
            "content-type": "application\/xml",
            "transfer-encoding": "chunked",
            "connection": "keep-alive",
            "vary": "Accept-Encoding",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
Deleted	array	被删除的对象的信息数组，每项都包含了一个被成功删除的对象的删除标记、`key` 和版本号等信息
Errors	array	删除失败的对象的信息数组，每项都包含了一个被成功删除的对象的信息和错误码

代码示例

```
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    //List to be deleted
    $objects = [
        ['Key' => 'key1','VersionId'=>'VersionId1'],
        ['Key' => 'key2','VersionId'=>'VersionId2'],
    ];

    //Batch delete some objects
     $result = $s3Client->deleteObjects([
         'Bucket' => $bucket,
         'Delete' => [
             'Objects' => $objects
          ]
     ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
```

```
?>
```

复制对象

功能介绍

拷贝对象。

方法原型

```
function copyObject(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	目标桶的名称	是
CopySource	string	需要拷贝的源文件，以sourceBucket/sourceKey的形式传参。如果想指定版本号，则参照："$sourceBucket/$sourceKey?versionId=$versionId"的形式	是
key	string	目标对象名称	是
ACL	string	ACL访问权限，例如 `private`、`public-read` 等	否
StorageClass	string	存储类型，可选值为 `STANDARD`、`STANDARD_IA` 和 `GLACIER`，分别表示标准、低频、归档。默认为标准存储	否
Metadata	array	复制目的对象的元数据	否
MetadataDirective	string	从源对象复制元数据还是用请求中提供的元数据替换元数据；可选值：COPY：保留源对象的元数据（默认行为）、REPLACE：用新提供的元数据替换源对象的元数据	否

返回结果及说明

```
{
    "CopyObjectResult": {
        "ETag": "5a8dd3ad0756a93ded72b823b19dd877",
        "LastModified": "2024-11-08T09:36:10+00:00"
    },
    "VersionId": "",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket\/key",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 08 Nov 2024 09:36:10 GMT",
```

```
            "content-type": "application\/xml",
            "transfer-encoding": "chunked",
            "connection": "keep-alive",
            "vary": "Accept-Encoding",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    },
    "ObjectURL": "https:\/\/endpoint\/bucket\/key"
}
```


返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
CopyObjectResult	array	包含目标对象的 `ETag` 和最后修改时间等信息

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
     //Create a S3Client
     $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
     ]);

     $sourceBucket = 'your_source_bucket';
     $sourceKey = 'your_source_key';
     $targetBucket = 'your_target_bucket';
     $targetKey = 'your_target_key';

     //copy the object
     $result = $s3Client->copyObject([
         'Bucket' => $targetBucket,
         'CopySource' => "$sourceBucket/$sourceKey", //Or specify versionId
"$sourceBucket/$sourceKey?versionId=$versionId"
```

```
            'Key' => $targetKey,
            'ACL' => 'private',
            'StorageClass' => 'STANDARD',
        ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取对象元数据

功能介绍

获取Object元数据。

方法原型

```
function headObject(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的名称	是
VersionId	string	用于获取对象的特定版本	否

返回结果及说明


```
{
    "LastModified": "2024-11-08T08:44:39+00:00",
    "ContentLength": 73376124,
    "ETag": "\"bf97961e0231bc99294033496bd006ea-4\"",
    "VersionId": "",
    "ContentType": "binary\/octet-stream",
    "Metadata": [],
    "StorageClass": "STANDARD",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket\/key",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 08 Nov 2024 09:19:21 GMT",
            "content-type": "binary\/octet-stream",
            "content-length": "73376124",
            "connection": "keep-alive",
            "accept-ranges": "bytes",
            "last-modified": "Fri, 08 Nov 2024 08:44:39 GMT",
```

```
            "x-rgw-object-type": "Normal",
            "etag": "\"bf97961e0231bc99294033496bd006ea-4\"",
            "content-disposition": "attachment",
            "x-amz-storage-class": "STANDARD",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
ContentLength	int	以字节为单位的对象数据长度
ContentType	string	描述对象数据格式的MIME类型
ETag	string	对象的 ETag
LastModified	time	最后修改对象的时间
VersionId	string	对象版本
Metadata	dict	元数据字典

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';
```

```
    //Retrieve Object metadata
    $result = $s3Client->headObject([
        'Bucket' => $bucket,
        'Key'  => $key
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置对象访问权限

功能介绍

设置对象的访问权限。

方法原型

```
function putObjectAcl(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象名	是
VersionId	string	多版本场景下，指定对象的特定版本	否
ACL	string	指定Bucket的访问权限ACL，取值范围： `private` / `public-read` / `pulic-read-write` / `authenticated-read`	该方式下必须，其它模式不能使用
AccessControlPolicy	dict	包含多个授权列表和一个桶所有者参数	该方式下必须，其它模式不能使用
Grants	list	授权列表	该方式下必须
Grantee	dict	被授权用户	该方式下必须
Type	string	被授权用户类型，取值范围： `CanonicalUser` / `AmazonCustomerByEmail` / `Group`	该方式下必须
ID(Grantee)	string	被授权用户ID	Type 为 `CanonicalUser`，则该字段必须
EmailAddress	string	被授权用户邮箱	Type 为 `AmazonCustomerByEmail`，则该字段必须

URI	string	被授权组URI，取值范围为
		所有用户：
		`http://acs.amazonaws.com/groups/global/AllUsers`
		所有认证用户：
		`http://acs.amazonaws.com/groups/global/AuthenticatedUsers`
			如果Type为 `Group`，则该字段必须

Permission	string	向被授权用户授予的权限，取值范围： `FULL_CONTROL` / `WRITE` / `WRITE_ACP` / `READ` / `READ_ACP`	该方式下必须
Owner	dict	Bucket 所有者	该方式下必须
ID(Owner)	string	Bucket 所有者ID	该方式下必须
GrantFullControl	string	被授权用户可以对桶进行read, write, read ACP, and write ACP 操作 以下Grant*参数，格式都是"id=xxxx"或"emailAddress=xxxx"或者"uri=xxxx"以及他们的组合（用>逗号连接）	否
GrantRead	string	被授权用户可以对桶进行读操作，即list object	否
GrantWrite	string	被授权用户可以对桶进行写操作，创建新的对象，删除或覆盖写属于自己的对象	否
GrantReadACP	string	被授权用户可以读取桶的ACL	否
GrantWriteACP	string	被授权用户可以修改桶的ACL	否

返回结果及说明


```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint\/mbucket\/ceph.conf?acl",
        "headers": {
            "x-amz-request-id": "tx000000000000000000075-00671b4eb6-d359-
default",
            "content-type": "application\/xml",
            "content-length": "0",
            "date": "Fri, 25 Oct 2024 07:54:30 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

//putBucketAcl
try {
    $result = $s3Client->putBucketAcl([
        'Bucket' => $bucket,
        'Key' => $key,
        //1.ACL设置
        'ACL' => 'public-read',
        /* //2.AccessControlPolicy设置
        'AccessControlPolicy' => [
        'Grants' => [
            [
                'Grantee' => [
                    'ID' => 'admin_test',
                    'Type' => 'CanonicalUser', // REQUIRED
                ],
                'Permission' => 'FULL_CONTROL',
            ],
            [
                'Grantee' => [
                    'ID' => 'admin_test',
                    'Type' => 'CanonicalUser', // REQUIRED
                    'URI' => 'http://acs.amazonaws.com/groups/global/AllUsers'
                ],
                'Permission' => 'READ'
            ]
        ],
        'Owner' => [
            'ID' => 'admin_test',
        ],
```

```
    ]
    //3.Grant*设置
    'GrantWrite' => 'id=admin_test'
    */
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取对象访问权限

功能介绍

获取对象的访问权限设置。

方法原型

```
function getObjectAcl(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象名	是
VersionId	string	多版本场景下，指定对象的特定版本	否

返回结果及说明

```
{
    "Owner": {
        "DisplayName": "admin_test",
        "ID": "admin_test"
    },
    "Grants": [
        {
            "Grantee": {
                "DisplayName": "admin_test",
                "EmailAddress": "",
                "ID": "admin_test",
                "Type": "CanonicalUser"
            },
            "Permission": "FULL_CONTROL"
        }
    ],
    "@metadata": {
        "statusCode": 200,
```

```
        "effectiveUri": "http:\/\/endpoint\/bucket\/key?acl",
        "headers": {
            "x-amz-request-id": "tx000000000000000000070-00671b493c-d359-
default",
            "content-type": "application\/xml",
            "content-length": "474",
            "date": "Fri, 25 Oct 2024 07:31:08 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
Owner	dict	Bucket 所有者
DisplayName(Owner)	string	Bucket 所有者的展示名
ID(Owner)	string	Bucket 所有者的用户ID
Grants	list	授权列表
Grantee	dict	被授权用户
Type	string	被授权用户类型
URI	string	被授权用户URI
ID(Grantee)	string	被授权用户ID
DisplayName(Grantee)	string	被授权用户展示名
EmailAddress	string	被授权用户邮箱
Permission	string	被授权权限
@metadata	dict	http 请求返回数据

代码示例

```php
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
```

```
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

//getObjectAcl
try {
    $result = $s3Client->getObjectAcl([
        'Bucket' => $bucket,
        'Key' => $key,
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置对象标签

功能介绍

为桶中对象设置标签。标签的最大数量限制为每个对象 10 个标签。

方法原型

```
function putObjectTagging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的key	是
Tagging	Tagging	封装标签键值对的数组	是
VersionId	string	对象的版本号	否

`Tagging` 属性表

属性名	类型	说明	是否必要
TagSet	Tag	标签集合	是

`Tag` 属性表

属性名	类型	说明	是否必要
Key	string	键	是
Value	string	值	是

返回结果及说明

```
{
    "VersionId": "",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint/bucket/key?tagging",
        "headers": {
            "x-amz-request-id": "tx000000000000000000001b-006718c427-371d-
default",
            "content-type": "application\/xml",
            "content-length": "0",
            "date": "Wed, 23 Oct 2024 09:38:47 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';
```

```php
$key = 'your_object_key';

try {
    $result = $s3Client->putObjectTagging([
        'Bucket' => $bucket,
        'Key' => $key,
        'Tagging' => [
            'TagSet' => [
                [
                    'Key' => 'Tag1',
                    'Value' => 'Value1',
                ],
                [
                    'Key' => 'Tag2',
                    'Value' => 'Value2',
                ],
            ],
        ]
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取对象标签

功能介绍

查询对象的标签信息。

方法原型

```php
function getObjectTagging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的 `key`	是
VersionId	string	对象的版本号	否

返回结果及说明


```
{
    "TagSet": [
        {
            "Key": "Tag1",
            "Value": "Value1"
        },
        {
            "Key": "Tag2",
            "Value": "Value2"
        }
    ],
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "endpoint/bucket/key?tagging",
        "headers": {
            "x-amz-request-id": "tx000000000000000000030-006718d0d2-371d-
default",
            "content-type": "application\/xml",
            "content-length": "216",
            "date": "Wed, 23 Oct 2024 10:32:50 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
TagSet	Tag	标签集合
@metadata	dict	http 请求返回数据

`Tag` 属性表

属性名	类型	说明
Key	string	键
Value	string	值

代码示例

```
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;
```

```php
//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';
$key = 'your_object_key';

try {
    $result = $s3Client->getObjectTagging([
        'Bucket' => $bucket,
        'Key' => $key
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

删除对象标签

功能介绍

删除指定对象的全部标签信息。删除时可以指定版本号参数来删除指定版本的对象的标签信息，不指定时默认操作于当前版本。

方法原型

```php
function deleteObjectTagging(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的 `key`	是
VersionId	string	对象的版本号	否

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 204,
        "effectiveUri": "endpoint/bucket/key?tagging",
        "headers": {
            "x-amz-request-id": "tx000000000000000000036-006718d7c3-371d-
default",
            "date": "Wed, 23 Oct 2024 11:02:27 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';
$key = 'your_object_key';

try {
    $result = $s3Client->deleteObjectTagging([
        'Bucket' => $bucket,
        'Key' => $key
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
```

```
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

生成预签名链接

功能介绍

生成预签名链接。

方法原型

```
function createPresignedRequest(CommandInterface $command, $expires, array
$options = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的名称	是
expires	string	链接有效期时长	是
command	string	操作类型，PutObject上传对象；GetObject获取对象。如果值为PutObject上传对象，则生成的签名链接只能用于上传操作，无法通过链接获取对象	是

返回结果及说明

调用该方法会返回一个对象，通过对象的成员方法来获取签名链接

返回结果	类型	说明
getUri()	function	通过将该方法的返回值，转化成字符串类型，来获取最终的签名链接，例：(string)$object->getUri();

代码示例

```
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;
```

```php
function put($presignedUrl,$filePath,$https=true) {
    // 使用 cURL 发送 PUT 请求上传文件
    $ch = curl_init();
    curl_setopt($ch, CURLOPT_URL, $presignedUrl);
    curl_setopt($ch, CURLOPT_PUT, true);
    curl_setopt($ch, CURLOPT_CUSTOMREQUEST, 'PUT');
    curl_setopt($ch, CURLOPT_RETURNTRANSFER, true);
    if (!$https) {  //不验证https证书
        curl_setopt($ch, CURLOPT_SSL_VERIFYPEER, false);
        curl_setopt($ch, CURLOPT_SSL_VERIFYHOST, false);
    }
    curl_setopt($ch, CURLOPT_INFILE, fopen($filePath, 'rb'));
    curl_setopt($ch, CURLOPT_INFILESIZE, filesize($filePath));

    // 执行 cURL 请求并获取响应
    $response = curl_exec($ch);
    if ($response === false) {
        throw new Exception(curl_error($ch), curl_errno($ch));
    }

    // 检查 HTTP 响应码以确认上传是否成功
    $httpCode = curl_getinfo($ch, CURLINFO_HTTP_CODE);
    if ($httpCode >= 200 && $httpCode < 300) {
        return true;
    } else {
        echo "Failed to upload file. HTTP Code: $httpCode\n";
        return false;
    }

    // 关闭 cURL 会话
    curl_close($ch);
}

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';
    $expires = '+5 minutes'; //this is an example
    $command = 'PutObject'; // or GetObject
    $file_path = 'D:\video\1.txt';

    //get the  signature address
    $presignedRequest = $s3Client->createPresignedRequest(
        $s3Client->getCommand($command, [
            'Bucket' => $bucket,
            'Key'    => $key,
        ]),
```

```
        $expires
    );
    $result = (string) $presignedRequest->getUri();
    echo $result;
    echo put($result,$file_path);
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

对象解冻

功能介绍

解冻归档存储类型的对象。

方法原型

```
function restoreObject(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的名称	是
VersionId	string	对象的版本号	否
RestoreRequest	RestoreRequest	解冻操作相关参数	是

`RestoreRequest` 属性表

属性名	类型	说明	是否必要
Days	int64	解冻后副本的保留时间，支持范围为 [1, 31]	是
GlacierJobParameters	GlacierJobParameters	解冻操作相关参数	否

`GlacierJobParameters` 属性表

属性名	类型	说明	是否必要
Tier	string	解冻操作类型，支持值为 STANDARD	是

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 202,
        "effectiveUri": "endpoint/bucket/key?restore",
        "headers": {
            "x-amz-request-id": "tx000000000000000000000d-00671b0034-5e2d-
default",
            "content-length": "0",
            "date": "Fri, 25 Oct 2024 02:19:32 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据

代码示例

```
<?php
// require '/path/to/aws-autoloader.php';
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_access_key',
            'secret' => 'your_secret_key'
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket';
$key = 'your_object_key';

try {
    $result = $s3Client->restoreObject([
        'Bucket' => $bucket,
        'Key' => $key,
        'RestoreRequest' => [
            'Days' => 1
        ]
```

```
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置对象依法保留配置

功能介绍

请求在指定对象上使用依法保留配置。

方法原型

```
function putObjectLegalHold(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的Key	是
LegalHold	dict	指定对象依法保留配置	否
Status	string	表示指定对象是否设置依法保留配置，取值范围：`OFF` / `ON`	否
VersionId	string	配置依法保留的对象的版本ID	否

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket\/key?legal-hold",
        "headers": {
            "x-amz-request-id": "tx000000000000000000000c-00671ef64a-fa69-
default",
            "content-length": "0",
            "date": "Mon, 28 Oct 2024 02:26:18 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据返回对象的内容

代码示例

```
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

//putObjectLegalHold
try {
    $result = $s3Client->putObjectLegalHold([
        'Bucket' => $bucket,
        'Key' => $key,
        'LegalHold' => [
        'Status' => 'ON',
    ],
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取对象依法保留配置

功能介绍

获取指定对象上的依法保留配置。

方法原型

```
function getObjectLegalHold(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	进行依法保留设置的对象名	是
VersionId	string	配置依法保留的对象的版本ID	否

返回结果及说明

```
{
    "LegalHold": {
        "Status": "ON"
    },
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket\/key?legal-hold",
        "headers": {
            "x-amz-request-id": "tx000000000000000000000e-00671efabe-fa69-
default",
            "content-type": "application\/xml",
            "content-length": "81",
            "date": "Mon, 28 Oct 2024 02:45:18 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
LegalHold	dict	指定对象的依法保留配置
@metadata	dict	http 请求返回数据返回对象的内容

代码示例

```
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
```

```php
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

//getObjectLegalHold
try {
    $result = $s3Client->getObjectLegalHold([
        'Bucket' => $bucket,
        'Key' => $key,
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

设置对象保留期限配置

功能介绍

请求在指定对象上设置对象保留期限配置。

方法原型

```php
function putObjectRetention(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的Key	是
Retention	dict	指定对象保留期限配置	否
Mode	string	表示指定对象的保留期限模式，取值范围：`GOVERNANCE` / `COMPLIANCE`	否
RetainUtilDate	datetime	对象锁定保留期限过期日期	否
VersionId	string	配置依法保留的对象的版本ID	否
BypassGovernanceRetention	boolean	表示这个操作是否应该绕过监管模式	否

返回结果及说明

```
{
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\bucket\/key?retention",
        "headers": {
            "x-amz-request-id": "tx00000000000000000010-00671f0289-fa69-
default",
            "content-length": "0",
            "date": "Mon, 28 Oct 2024 03:18:33 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据返回对象的内容

代码示例

```
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
```

```
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);

$bucket = 'your_bucket_name';
$key = 'your_key_name';

//putObjectRetention
try {
    $result = $s3Client->putObjectRetention([
        'Bucket' => $bucket,
        'Key' => $key,
        'Retention' => [
            'Mode' => 'GOVERNANCE',
            'RetainUntilDate' => new DateTime('2024-11-25'),
        ],
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

获取对象保留期限配置

功能介绍

获取指定对象上的保留期限配置。

方法原型

```
function getObjectRetention(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	进行保留期限设置的对象名	是
VersionId	string	配置保留期限的对象的版本ID	否

返回结果及说明


```
{
    "Retention": {
        "Mode": "GOVERNANCE",
        "RetainUntilDate": "2024-11-25T00:00:00+00:00"
    },
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "http:\/\/endpoint\/bucket\/key?retention",
        "headers": {
            "x-amz-request-id": "tx000000000000000000012-00671f2cef-fa69-
default",
            "content-type": "application\/xml",
            "content-length": "149",
            "date": "Mon, 28 Oct 2024 06:19:27 GMT",
            "connection": "Keep-Alive"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
Retention	dict	指定对象的保留期限配置
Mode	string	表示指定对象的保留期限模式
RetainUtilRetention	datetime	对象锁定保留期限过期日期
@metadata	dict	http 请求返回数据返回对象的内容

代码示例

```
<?php
require './aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

//Create a S3Client
$s3Client = new S3Client([
    'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
    'region' => 'us-east-1',
    'version' => '2006-03-01',
    'endpoint' => 'your_endpoint'
]);
```

```php
$bucket = 'your_bucket_name';
$key = 'your_key_name';

//getObjectRetention
try {
    $result = $s3Client->getObjectRetention([
        'Bucket' => $bucket,
        'Key' => $key,
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

分段上传操作

初始化分段上传

功能介绍

初始化分段上传。

方法原型

```php
function createMultipartUpload(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
ACL	string	配置上传对象的预定义 ACL 信息，如`private`、`public-read`、`public-read-write`等。不传该参数时默认为`private`	否
Bucket	string	桶的名称	是
Key	string	对象的名称	是
StorageClass	string	存储类型，可选的有`STANDARD`，`STANDARD_IA`，`GLACIER`，默认为`STANDARD`	否
Tagging	string	对象标签，例："Tag1=Valve1&Tag2=Value2"	否
Metadata	array	对象元数据	否

返回结果及说明


```
{
    "Bucket": "bucket",
    "Key": "key",
    "UploadId": "UploadId",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket\/key?uploads",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 08 Nov 2024 09:27:28 GMT",
            "content-type": "application\/xml",
            "content-length": "242",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
UploadId	string	分段上传的唯一标识，以供后续接口调用（uploadPart、completeMultipartUpload）
Bucket	string	桶的名称
Key	string	对象的名称

代码示例

```
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
```

```
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';

    //initialize shard upload
    $initiateResponse = $s3Client->createMultipartUpload([
        'Bucket' => $bucket,
        'Key'  => $key,
        //'Tagging'=>'Tag1=Valve1&Tag2=Value2',
        //'Metadata' => [
          //   'metadata1' => 'value1',
        //],
    ]);
    echo $initiateResponse;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

进行分段上传

功能介绍

分段上传。

方法原型

```
function uploadPart(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	初始化分段上传所用的对象名称	是
UploadId	string	上传ID，通过createMultipartUpload接口获取	是
PartNumber	string	分段编号，可用范围： [1, 10000]	是
Body	binary data	分段的数据，依次读取文件的分段的数据(二进制流)	是

返回结果及说明

```
{
    "ETag": "\"6f96cfdfe5ccc627cadf24b41725caa4\"",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket\/key?
uploadId=uploadId&partNumber=1",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Wed, 30 Oct 2024 01:47:35 GMT",
            "content-length": "0",
            "connection": "keep-alive",
            "etag": "\"6f96cfdfe5ccc627cadf24b41725caa4\"",
            "accept-ranges": "bytes",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
ETag	string	分段的 `Entity Tag`，以供后续接口使用（completeMultipartUpload）

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);
```

```
    $bucket = 'your_bucket';
    $key = 'your_object_key';
    $localFilePath = 'your_local_file_path';
    $uploadId = 'your_upload_id'; //ID obtained through
api(createMultipartUpload)
    $partNumberMarker = 0;
    $parts = [];

    // open local file
    $fileHandle = fopen($localFilePath, 'r');
    // Read files block by block and upload them
    while (!feof($fileHandle)) {
        $partNumberMarker++;
        $data = fread($fileHandle, 10 * 1024 * 1024); //example:slice size 10M
        if ($data === false) {
            break; // file end
        }
        //upload
        $partResponse = $s3Client->uploadPart([
            'Bucket'     => $bucket,
            'Key'        => $key,
            'UploadId'   => $uploadId,
                'PartNumber' => $partNumberMarker,
                'Body'       => $data,
        ]);
        echo $partResponse;
        //The following parameters are provided for future api
use(completeMultipartUpload)
        $parts[] = [
            'PartNumber' => $partNumberMarker,
            'ETag'       => $partResponse['ETag'],
        ];
    }
    fclose($fileHandle); //close file
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

完成分段上传

功能介绍

完成分段上传。

方法原型

```
function completeMultipartUpload(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的名称	是
UploadId	string	上传ID，通过createMultipartUpload接口获取	是
MultipartUpload	CompletedMultipartUpload	每次调用分段上传接口(uploadPart)返回的二维数组汇总，每个数组包含PartNumber和ETag	是

`CompletedMultipartUpload` 属性表

属性名	类型	说明	是否必要
Parts	CompletedPart	已完成的分段上传信息数组	是

`CompletedPart` 属性表

属性名	类型	说明
ETag	string	分段的 `Entity Tag`
PartNumber	int	分段编号

返回结果及说明


```
{
    "VersionId": "",
    "Location": "endpoint\/bucket\/key",
    "Bucket": "bucket",
    "Key": "key",
    "ETag": "bf97961e0231bc99294033496bd006ea-4",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpoint\/bucket\/ttt.mp4?uploadId=uploadId",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 08 Nov 2024 09:39:33 GMT",
            "content-type": "application\/xml",
            "content-length": "284",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
```

```
            ]
        }
    },
    "ObjectURL": "endpoint\/bucket\/key"
}
```

返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
Bucket	string	桶的名称
Key	string	初始化分段上传所用的对象名称
ETag	string	最终上传对象的 Entity Tag
Location	string	最终上传对象的具体位置路径
VersionId	string	最终上传对象的的版本号径

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';
    $uploadId = 'your_upload_id'; //ID obtained through
api(createMultipartUpload)
    $parts = [ //Summary of arrays returned through api(uploadPart)
        ['PartNumber'=>1,'ETag'=>'ETag1'],
        ['PartNumber'=>2,'ETag'=>'ETag2'],
    ];

    //complete multi part upload
    $res = $s3Client->completeMultipartUpload([
        'Bucket'     => $bucket,
```

```
        'Key'        => $key,
        'UploadId'   => $uploadId,
        'MultipartUpload' => [
            'Parts' => $parts,
        ],
    ]);
    echo $res;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

列出分段

功能介绍

列出特定分段上传任务的分段。

方法原型

```
function listParts(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	初始化分段上传所用的对象名称	是
MaxParts	int	设置返回分段数量的最大值，默认值为 1000，可选范围：[1, 1000]	否
PartNumberMarker	int	指定返回结果的分段编号起始位置，只有分段编号大于该值的分段会被返回	否
UploadId	string	分段上传的唯一标识	是

返回结果及说明

```
{
    "Bucket": "bucket",
    "Key": "key",
    "UploadId": "UploadId",
    "PartNumberMarker": 0,
    "NextPartNumberMarker": 2,
    "MaxParts": 1000,
    "IsTruncated": false,
    "Parts": [
        {
```

```
            "PartNumber": 1,
            "LastModified": "2024-11-08T09:31:17+00:00",
            "ETag": "\"733f2a7397bf0ab58d3b9bd00017614c\"",
            "Size": "2097152"
        },
        {
            "PartNumber": 2,
            "LastModified": "2024-11-08T09:31:27+00:00",
            "ETag": "\"a203f7d532cb6282281b6451ca18724a\"",
            "Size": "2097152"
        }
    ],
    "Owner": {
        "DisplayName": "",
        "ID": ""
    },
    "StorageClass": "STANDARD",
    "@metadata": {
        "statusCode": 200,
        "effectiveUri": "https:\/\/endpint\/bucket\/ttt.mp4?uploadId=UploadId",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Fri, 08 Nov 2024 09:31:59 GMT",
            "content-type": "application\/xml",
            "transfer-encoding": "chunked",
            "connection": "keep-alive",
            "vary": "Accept-Encoding",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```

属性名	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，200代表请求成功
Bucket	string	桶的名称
IsTruncated	bool	表示本次响应是否被截断
Key	string	初始化分段上传所用的对象名称
MaxParts	int	本次响应可返回分段数量的最大值
NextPartNumberMarker	int	当响应被截断时返回，可作为下次请求的 `PartNumberMarker` 参数以进行连续查询
Owner	Owner	分段上传所属的用户
PartNumberMarker	int64	当前响应中分段的起始编号
Parts	Part	分段信息数组

`Owner` 属性表

属性名	类型	说明
DisplayName	string	用户名称
ID	string	用户 ID

`Part` 属性表

属性名	类型	说明
ETag	string	分段的 `Entity Tag`
LastModified	Time	该分段完成上传的时间点
PartNumber	int	分段编号
Size	int	分段大小，单位为字节

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
```

```
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';
    $uploadId = 'your_upload_id'; //ID obtained through
api(createMultipartUpload)

    //List the segments of specific shard upload tasks
    $result = $s3Client->listParts([
        'Bucket' => $bucket,
        'Key' => $key,
        'UploadId' => $uploadId
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```

终止分段上传

功能介绍

中止分段上传。将弃用对应的 `UploadId` 并舍弃之前在该分段上传中上传的所有分段。

方法原型

```
function abortMultipartUpload(array $args = [])
```

参数说明

属性名	类型	说明	是否必要
Bucket	string	桶的名称	是
Key	string	对象的名称	是
UploadId	string	上传ID，通过createMultipartUpload接口获取	是

返回结果及说明


```
{
    "@metadata": {
        "statusCode": 204,
        "effectiveUri": "https:\/\/endpoint\/bucket\/key?uploadId=uploadId",
        "headers": {
            "server": "ct-zos\/1.22.2",
            "date": "Tue, 29 Oct 2024 01:00:31 GMT",
            "connection": "keep-alive",
            "x-amz-request-id": "request-id"
        },
        "transferStats": {
            "http": [
                []
            ]
        }
    }
}
```


返回结果	类型	说明
@metadata	dict	http 请求返回数据
statusCode	int	http 请求返回状态码，204代表请求成功

代码示例

```php
<?php

require __DIR__.'/aws/aws-autoloader.php';

use Aws\S3\S3Client;
use Aws\Exception\AwsException;
use Aws\S3\Exception\S3Exception;

try {
    //Create a S3Client
    $s3Client = new S3Client([
        'credentials' => [
            'key'    => 'your_ak',
            'secret' => 'your_sk',
        ],
        'region' => 'us-east-1',
        'version' => '2006-03-01',
        'endpoint' => 'your_endpoint'
    ]);

    $bucket = 'your_bucket';
    $key = 'your_object_key';
    $uploadId = 'your_upload_id'; //ID obtained through
api(createMultipartUpload)

    //abort multipart upload
    $result = $s3Client->abortMultipartUpload([
```

```
        'Bucket' => $bucket,
        'Key' => $key,
        'UploadId' => $uploadId
    ]);
    echo $result;
} catch (S3Exception $e) {
    echo $e->getMessage();
} catch (AwsException $e) {
    echo $e->getAwsRequestId()."\n";
    echo $e->getAwsErrorType()."\n";
    echo $e->getAwsErrorCode()."\n";
    var_dump($e->toArray());
}
?>
```