对象存储服务

BrowserJS SDK 开发指南

文档版本 01

发布日期 2024-12-03

华为技术有限公司

版权所有 © 华为技术有限公司 2024。 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


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。

安全声明

漏洞处理流程

华为公司对产品漏洞管理的规定以“漏洞处理流程”为准，该流程的详细内容请参见如下网址：
https://www.huawei.com/cn/psirt/vul-response-process
如企业客户须获取漏洞信息，请参见如下网址：
https://securitybulletin.huawei.com/enterprise/cn/security-advisory

 版权所有 © 华为技术有限公司

目 录

 版权所有 © 华为技术有限公司

 版权所有 © 华为技术有限公司

 版权所有 © 华为技术有限公司

 版权所有 © 华为技术有限公司

1 使用前须知

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

- 使用OBS BrowserJS SDK访问桶前，必须**配置桶的CORS**。
- 请确认您已经熟悉OBS的基本概念，如**桶（Bucket）**、**对象（Object）**、**访问密钥（AK和SK）**等。
- 您可以先参考**OBS客户端通用示例**，了解OBS BrowserJS SDK接口调用的通用方式。
- OBS客户端支持回调函数和Promise对象两种方式返回调用结果。
- 当前各区域特性开放不一致，部分特性只在部分区域开放，使用过程中如果接口HTTP状态码为405，请确认该区域是否支持该功能特性。

版权所有 © 华为技术有限公司

2 SDK 下载

SDK 源码和 API 文档

- OBS BrowserJS SDK最新版本源码：**最新源码下载**
- OBS BrowserJS SDK历史版本下载地址：**历史版本下载**
- OBS BrowserJS SDK API文档：**OBS BrowserJS SDK API参考**

兼容性

- 版本修订记录信息：**ChangeLog**。
- 推荐的浏览器版本：
  - 完全支持HTML5的浏览器。
- 功能限制：不支持创建桶、列举桶、设置桶CORS配置。
- 接口函数：与旧版本（2.1.x）**不完全兼容，接口变化如下表**：

接口函数	变化说明
ObsClient.setBucketAcl	请求参数中Grants字段变为数组，去掉Grants.Grant字段。
ObsClient.getBucketAcl	响应结果中InterfaceResult.Grants字段变为数组，去掉InterfaceResult.Grants.Grant字段。
ObsClient.setObjectAcl	请求参数中Grants字段变为数组，去掉Grants.Grant字段。
ObsClient.getObjectAcl	响应结果中InterfaceResult.Grants字段变为数组，去掉InterfaceResult.Grants.Grant字段。
ObsClient.setBucketLogging	请求参数中LoggingEnabled.TargetGrants字段变为数组，去掉LoggingEnabled.TargetGrants.Grant字段。
ObsClient.getBucketLogging	响应结果中InterfaceResult.LoggingEnabled.TargetGrants字段变为数组，去掉InterfaceResult.LoggingEnabled.TargetGrants.Grant字段。

版权所有 © 华为技术有限公司

接口函数	变化说明
ObsClient.setBucketWebsite	请求参数中RoutingRules字段变为数组，去掉RoutingRules.RoutingRule字段。
ObsClient.getBucketWebsite	响应结果中InterfaceResult.RoutingRules字段变为数组，去掉InterfaceResult.RoutingRules.RoutingRule字段。
ObsClient.getBucketCors	响应结果中InterfaceResult.CorsRule字段改名为InterfaceResult.CorsRules。
ObsClient.setBucketTagging	请求参数中TagSet字段改名为Tags并变为数组，去掉TagSet.Tag字段。
ObsClient.getBucketTagging	响应结果中InterfaceResult.TagSet字段改名为Tags并变为数组，去掉InterfaceResult.TagSet.Tag字段。

版权所有 © 华为技术有限公司

3 示例程序

OBS BrowserJS SDK提供了丰富的示例程序，方便用户参考或直接使用。

您可以从OBS BrowserJS SDK开发包中获取示例程序。

下载eSDK_Storage_OBS_<VersionId>_BrowserJS.zip，解压后eSDK_Storage_OBS_<VersionId>_BrowserJS/examples为示例程序。

您也可以从下面表格中直接下载示例程序。

示例包括以下内容：

示例代码	说明
bucket-operations-sample	展示了桶相关接口的用法
object-operations-sample	展示了对象相关接口的用法
upload-download-sample	展示了上传和下载对象的用法
create-folder-sample	展示了创建文件夹的用法
delete-objects-sample	展示了批量删除对象的用法
list-objects-sample	展示了列举对象的用法
list-versions-sample	展示了列举多版本对象的用法
simple-multipart-upload-sample	展示了分段上传的基本用法
post-object-sample	展示了表单上传对象的用法
temporary-signature-sample	展示了使用URL进行授权访问的用法

版权所有 © 华为技术有限公司

4 快速入门

4.1 OBS 服务环境搭建

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

步骤1 注册云服务账号

使用OBS之前必须要有一个云服务账号。

1. 打开浏览器。
2. 登录**公有云网站**。
3. 在页面右上角单击“注册”。
4. 按需填写注册信息并单击“同意协议并注册”。

步骤2 开通OBS服务

使用OBS服务之前必须先充值，才能正常使用OBS服务。

1. 登录**管理控制台**。
2. 单击页面右上角的“费用和成本”进入费用中心页面。
3. 选择“资金管理 > 充值”，系统自动跳转到充值窗口。
4. 根据界面提示信息，对账户进行充值。
5. 充值成功后，关闭充值窗口，返回管理控制台首页。
6. 在服务列表中选择“对象存储服务 OBS”，开通并进入OBS管理控制台。

步骤3 创建访问密钥

OBS通过用户账户中的AK和SK进行签名验证，确保通过授权的账户才能访问指定的OBS资源。以下是对AK和SK的解释说明：

- AK：Access Key ID，接入键标识，用户在对象存储服务系统中的接入键标识，一个接入键标识唯一对应一个用户，一个用户可以同时拥有多个接入键标识。对象存储服务系统通过接入键标识识别访问系统的用户。

版权所有 © 华为技术有限公司

- SK：Secret Access Key，安全接入键，用户在对象存储服务系统中的安全接入键，是用户访问对象存储服务系统的密钥，用户根据安全接入键和请求头域生成鉴权信息。安全接入键和接入键标识一一对应。

访问密钥分永久访问密钥（AK/SK）和临时访问密钥（AK/SK和SecurityToken）两种。每个用户最多可创建两个有效的永久访问密钥。临时访问密钥只在设置的有效期内能够访问OBS，过期后需要重新获取。出于安全性考虑，建议您使用临时访问密钥访问OBS，或使用永久访问密钥访问OBS时，定期更新您的访问密钥（AK/SK）。两种密钥的获取方式如下所示。

- 永久访问密钥：
  a. 登录**管理控制台**。
  b. 单击页面右上角的用户名，并选择“我的凭证”。
  c. 在“我的凭证”页面，单击左侧导航栏的“访问密钥”。
  d. 在“访问密钥”页面，单击“新增访问密钥”。

说明

每个用户最多可创建两个有效的访问密钥。

  e. 在弹出的“新增访问密钥”对话框中，输入描述内容（建议），单击“确定”。

  f. （可选）在弹出的“身份验证”对话框中，选择合适的验证方式进行验证，单击“确定”。

版权所有 © 华为技术有限公司

g. 在弹出的“创建成功”提示框中，单击“立即下载”后，密钥会直接保存到浏览器默认的下载文件夹中。

h. 打开下载下来的“credentials.csv”文件即可获取到访问密钥（AK和SK）。

说明

- 在密钥文件中，Access Key ID列对应的值即AK，Secret Access Key列对应的值即SK。
- 为防止访问密钥泄露，建议您将其保存到安全的位置。如果用户在此提示框中单击“取消”，则不会下载密钥，后续也将无法重新下载。如果需要使用访问密钥，可以重新创建新的访问密钥。

- 临时访问密钥：

  临时AK/SK和SecurityToken是系统颁发给用户的临时访问令牌，通过接口设置有效期，范围为15分钟至24小时，过期后需要重新获取。临时AK/SK和SecurityToken遵循权限最小化原则。使用临时AK/SK鉴权时，临时AK/SK和SecurityToken必须同时使用。

  获取临时访问密钥的接口请参考**获取临时AK/SK和securitytoken**。

须知

OBS属于全局级服务，所以在获取临时访问密钥时，需要设置Token的使用范围取值为domain，表示获取的Token可以作用于全局服务，全局服务不区分项目或者区域。

----结束

版权所有 © 华为技术有限公司

4.2 开发环境准备

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

- 从**Eclipse官网**下载并安装Eclipse IDE for JavaScript and Web Developer最新版本。
- 下载并安装支持HTML5标准的浏览器，例如Chrome 64。

4.3 安装 SDK

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

如**表1**所示，BrowserJS SDK有2种安装方式。

表 4-1 BrowserJS SDK 安装方式概览

序号	方式
1	手动下载源码开发包安装
2	使用npm命令下载安装

手动下载源码开发包安装

以安装OBS BrowserJS SDK最新版本为例，步骤如下：

步骤1 **下载**OBS BrowserJS SDK开发包。

步骤2 解压该开发包，可以看到其中包含examples文件夹（示例代码），dist文件夹（SDK库文件）和README.txt（SDK版本特性描述文件）。

步骤3 浏览器页面引入SDK库文件：

```
<script src="./esdk-obs-browserjs-without-polyfill-x.x.x.min.js"></script>
```

----结束

版权所有 © 华为技术有限公司

须知

- SDK内部集成了axios作为HTTP库，该库依赖于promise特性。
- dist文件夹中包含两个版本的SDK库文件，如果使用的浏览器不支持promise特性（例如IE10、IE11），请引入库文件：esdk-obs-browserjs-x.x.x.min.js。

使用 npm 命令下载安装

步骤1 运行**npm -v**命令查看npm版本并确保npm已安装。

步骤2 运行**npm install esdk-obs-browserjs**命令执行安装。

----结束

说明

- 如果您使用的是Windows操作系统，当运行npm命令时提示“不是内部或外部命令”，请在Path环境变量中增加npm的安装目录（一般为Node.js的安装目录）。
- 您可能需要重启电脑使环境变量生效。
- 如果您使用npm安装依赖时出现网络错误，请使用代理。

4.4 获取服务地址

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

- 您可以从**这里**查看OBS当前开通的服务地址和区域信息。

须知

SDK支持带协议名和不带协议名两种方式传入服务地址，例如获取到的服务地址为“your-endpoint”，则初始化OBS客户端时传入的服务地址可以为“http://your-endpoint”、“https://your-endpoint”和“your-endpoint”三种形式。

4.5 配置桶允许跨域请求

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

要使用OBS BrowserJS SDK访问OBS服务的桶，必须配置该桶允许跨域请求（桶的CORS），推荐为桶的CORS配置的规则如下：

版权所有 © 华为技术有限公司

配置项	配置值	说明
AllowedOrigin	*	允许任意请求来源。 **说明** 也可以配置具体的域名或IP。
AllowedMethod	PUT、GET、POST、DELETE、HEAD	允许所有的HTTP方法。
AllowedHeader	*	允许请求中携带任意头域。
ExposeHeader	• ETag • x-obs-request-id • x-obs-api • Content-Type • Content-Length • Cache-Control • Content-Disposition • Content-Encoding • Content-Language • Expires • x-obs-id-2 • x-reserved-indicator • x-obs-version-id • x-obs-copy-source-version-id • x-obs-storage-class • x-obs-delete-marker • x-obs-expiration • x-obs-website-redirect-location • x-obs-restore • x-obs-version • x-obs-object-type • x-obs-next-append-position	允许响应中返回指定的附加头域。 **须知** 附加头域：指定浏览器可以暴露给客户端的响应消息头。 比如在浏览器环境中，需要获取ETag值，由于ETag不属于标准响应头，就需要添加到扩展头域。

您可以从这里参考各客户端工具配置CORS的详细步骤。

 版权所有 © 华为技术有限公司

4.6 初始化 OBS 客户端

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

向OBS发送任一HTTP/HTTPS请求之前，必须先创建一个ObsClient实例：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用访问OBS
```

说明

- 更多关于OBS客户端初始化的内容请参考“初始化”章节。
- 日志配置详见**配置SDK日志**。

4.7 上传对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

本示例用于上传字符串“Hello OBS”到桶名为“bucketname”里，名称为“objectname”。

代码示例如下所示：

```
obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Body : 'Hello OBS'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```


版权所有 © 华为技术有限公司

说明

- 更多上传对象的信息，请参见**上传对象**。

4.8 下载对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

本示例用于下载桶名为“bucketname”里，名称为“objectname”的对象。

代码示例如下所示：

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

说明

- 更多下载对象的信息，请参见**下载对象**。

4.9 列举对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

当完成一系列上传对象操作后，可能需要查看桶中包含哪些对象。以下代码展示如何列举指定桶中的对象：

```
obsClient.listObjects({
    Bucket : 'bucketname'
}, function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
        if(result.CommonMsg.Status < 300 && result.InterfaceResult){
            for(var j in result.InterfaceResult.Contents){
                console.log('Contents[' + j +  ']:');
                console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                console.log('LastModified-->' + result.InterfaceResult.Contents[j]['LastModified']);
                console.log('Size-->' + result.InterfaceResult.Contents[j]['Size']);
```

 版权所有 © 华为技术有限公司

```
            }
        }
    }
});
```

说明

- 上面的代码默认列举1000个对象（Object）。
- 更丰富的列举功能，请参见**列举对象**。

4.10 删除对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

本示例用于删除桶名为“bucketname”里，名称为“objectname”的对象。

代码示例如下所示：

```
obsClient.deleteObject({
    Bucket : 'bucketname',
    Key : 'objectname'
}, function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

- 本示例仅用于删除单个对象，OBS批量删除对象，需自行遍历构建待批量删除对象的列表。
- 更丰富的删除功能，请参见**删除对象**。

4.11 OBS 客户端通用示例

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

回调函数返回结果：

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

```
// 创建ObsClient实例
var obsClient = new ObsClient({
```

 版权所有 © 华为技术有限公司

```
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 构造桶操作请求参数
var requestParam1 = {
        Bucket : 'bucketname'
        // 其他字段
};

var callback1 = function (err, result){
        // 处理桶操作调用结果
};

// 调用桶操作接口，如获取桶区域位置
obsClient.getBucketLocation(requestParam1, callback1);

//构造对象操作请求参数
var requestParam2 = {
        Bucket : 'bucketname',
        Key : 'objectname'
        // 其他字段
};

var callback2 = function (err, result){
        // 处理对象操作调用结果
};

// 调用对象操作接口，如下载对象
obsClient.getObject(requestParam2, callback2);
```

说明

对于桶操作接口，请求对象中固定包含Bucket字段用于指定桶名；对于对象操作接口，请求对象中固定包含Bucket字段和Key字段分别用于指定桶名与对象名。

以下代码展示了使用回调函数返回调用结果的通用示例：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 调用接口进行操作，例如上传对象
obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Body : 'Hello OBS'
}, function (err, result) {
        // 异常信息不为空，表明接口调用异常
        if(err){
            console.log('Error-->' + err);
        }else{
```

```
            // 异常信息为空，表明接口调用完成，应进一步判断HTTP状态码
            if(result.CommonMsg.Status < 300){// 操作成功
                if(result.InterfaceResult){
                    // 处理操作成功后业务逻辑
                }
            }else{// 操作失败，获取详细异常信息
                console.log('Code-->' + result.CommonMsg.Code);
                console.log('Message-->' + result.CommonMsg.Message);
                console.log('HostId-->' + result.CommonMsg.HostId);
                console.log('RequestId-->' + result.CommonMsg.RequestId);
            }
        }

});
```

Promise 对象返回结果：

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

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 构造桶操作请求参数
var requestParam1 = {
        Bucket : 'bucketname'
        // 其他字段
};

// 调用桶操作接口，如获取桶区域位置
var promise1 = obsClient.getBucketLocation(requestParam1);
promise1.then((result) => {
    // 处理桶操作调用结果
}).catch((err)=>{
    // 处理错误
});

//构造对象操作请求参数
var requestParam2 = {
        Bucket : 'bucketname',
        Key : 'objectname'
        // 其他字段
};

// 调用对象操作接口，如下载对象
var promise2 = obsClient.getObject(requestParam2, callback2);
promise2.then((result) => {
    // 处理对象操作调用结果
}).catch((err)=>{
    // 处理错误
});
```

说明

对于桶操作接口，请求对象中固定包含Bucket字段用于指定桶名；对于对象操作接口，请求对象中固定包含Bucket字段和Key字段分别用于指定桶名与对象名。

 版权所有 © 华为技术有限公司

以下代码展示了使用Promise对象返回调用结果的通用示例：

```
// 引入obs库
var ObsClient = require('./lib/obs');

// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 调用接口进行操作，例如上传对象
obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Body : 'Hello OBS'
}).then((result) => {
    // 接口调用完成，应进一步判断HTTP状态码
    if(result.CommonMsg.Status < 300){// 操作成功
        if(result.InterfaceResult){
            // 处理操作成功后业务逻辑
        }
    }else{// 操作失败，获取详细异常信息
        console.log('Code-->' + result.CommonMsg.Code);
        console.log('Message-->' + result.CommonMsg.Message);
        console.log('HostId-->' + result.CommonMsg.HostId);
        console.log('RequestId-->' + result.CommonMsg.RequestId);
    }
}).catch((err) => {
    // 接口调用发生异常
    console.error('Error-->' + err);
});
```

4.12 预定义常量

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS BrowserJS SDK提供了一组预定义常量，方便用户直接使用。

您可以通过ObsClient.enums获取预定义常量对象。

更多关于预定义常量的介绍详见《 对象存储服务BrowserJS SDK API参考 》。

 版权所有 © 华为技术有限公司

5 初始化

5.1 配置桶的 CORS

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

由于OBS以桶为单位提供基于HTTP/HTTPS协议的分布式存储服务，而浏览器默认不允许Ajax跨域请求，因此使用OBS BrowserJS SDK访问桶前必须配置该桶的CORS。您可以通过OBS Console、OBS Browser或者除OBS BrowserJS SDK外的其他OBS SDK三种途径配置桶的CORS，推荐为桶的CORS配置的规则如下：

配置项	配置值	说明
AllowedOrigin	*	允许任意请求来源。 **说明** 也可以配置具体的域名或IP。
AllowedMethod	PUT、GET、POST、DELETE、HEAD	允许所有的HTTP方法。
AllowedHeader	*	允许请求中携带任意头域。

版权所有 © 华为技术有限公司

配置项	配置值	说明
ExposeHeader	• ETag • x-obs-request-id • x-obs-api • Content-Type • Content-Length • Cache-Control • Content-Disposition • Content-Encoding • Content-Language • Expires • x-obs-id-2 • x-reserved-indicator • x-obs-version-id • x-obs-copy-source-version-id • x-obs-storage-class • x-obs-delete-marker • x-obs-expiration • x-obs-website-redirect-location • x-obs-restore • x-obs-version • x-obs-object-type • x-obs-next-append-position	允许响应中返回指定的附加头域。 **须知** 附加头域：指定浏览器可以暴露给客户端的响应消息头。 比如在浏览器环境中，需要获取ETag值，由于ETag不属于标准响应头，就需要添加到扩展头域。

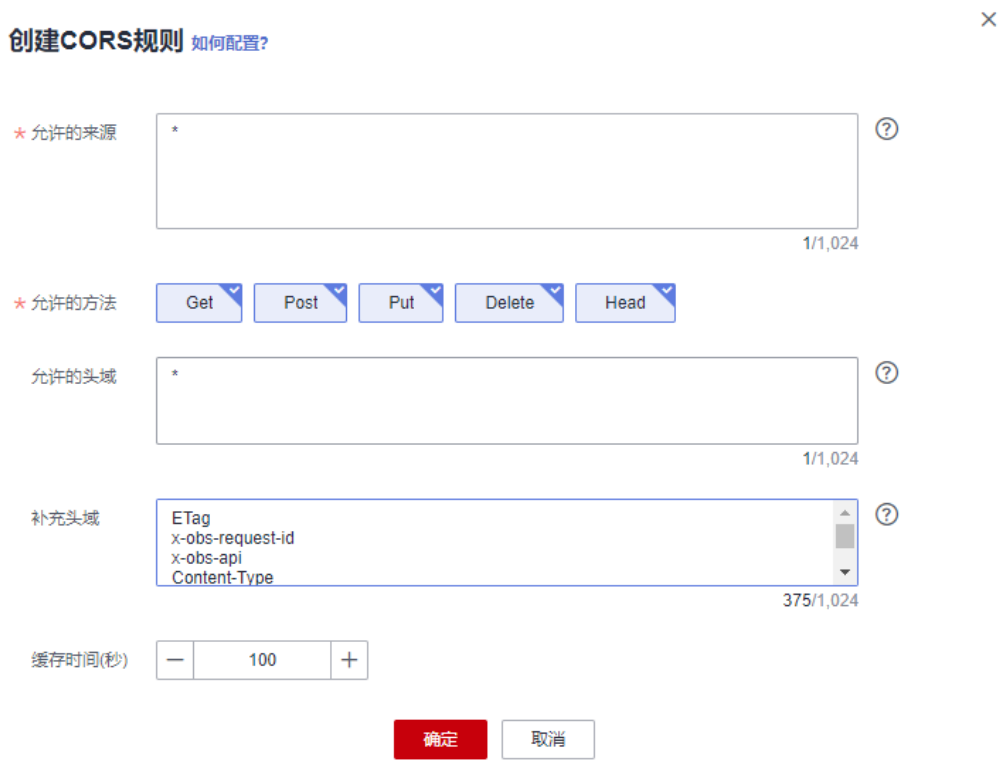
通过 OBS Console 配置桶的 CORS

步骤1 登录OBS Console后在桶列表中，单击待操作的桶，进入“概览”页面；如下图所示

步骤2 在“基础配置”下，单击“CORS规则”卡片，进入“CORS规则”界面。

版权所有 © 华为技术有限公司

步骤3 在“CORS规则”界面，单击“创建”，系统弹出“创建CORS规则”对话框，在该对话框中按照上表的参数进行配置，如下图所示：



步骤4 单击“确定”，并在“CORS规则”界面查看已配置好的规则。

----结束

说明

桶的CORS配置会在两分钟内生效，生效后才能使用OBS BrowserJS SDK访问桶。

通过 OBS Java SDK 配置桶的 CORS

您可以使用OBS Java SDK调用ObsClient.setBucketCors接口配置桶的CORS，示例代码如下：

```
// Endpoint以中国-香港为例，其他地区请按实际情况填写。
String endPoint = "https://obs.ap-southeast-1.myhuaweicloud.com";
String ak = System.getenv("ACCESS_KEY_ID");
String sk = System.getenv("SECRET_ACCESS_KEY_ID");
// 创建ObsClient实例
ObsClient obsClient = new ObsClient(ak, sk, endPoint);

BucketCors cors = new BucketCors();

List<BucketCorsRule> rules = new ArrayList<BucketCorsRule>();
BucketCorsRule rule = new BucketCorsRule();

// 指定允许跨域请求的来源
ArrayList<String> allowedOrigin = new ArrayList<String>();
allowedOrigin.add( "*");
rule.setAllowedOrigin(allowedOrigin);

// 指定允许跨域请求的方法
```

 版权所有 © 华为技术有限公司

```
ArrayList<String> allowedMethod = new ArrayList<String>();
allowedMethod.add("GET");
allowedMethod.add("POST");
allowedMethod.add("PUT");
allowedMethod.add("DELETE");
allowedMethod.add("HEAD");
rule.setAllowedMethod(allowedMethod);

// 指定允许跨域请求的头域
ArrayList<String> allowedHeader = new ArrayList<String>();
allowedHeader.add("*");
rule.setAllowedHeader(allowedHeader);

// 指定允许跨域请求的响应中的附加头域
ArrayList<String> exposeHeader = new ArrayList<String>();
exposeHeader.add("ETag");
exposeHeader.add("Content-Type");
exposeHeader.add("Content-Length");
exposeHeader.add("Cache-Control");
exposeHeader.add("Content-Disposition");
exposeHeader.add("Content-Encoding");
exposeHeader.add("Content-Language");
exposeHeader.add("Expires");
exposeHeader.add("x-obs-request-id");
exposeHeader.add("x-obs-id-2");
exposeHeader.add("x-reserved-indicator");
exposeHeader.add("x-obs-api");
exposeHeader.add("x-obs-version-id");
exposeHeader.add("x-obs-copy-source-version-id");
exposeHeader.add("x-obs-storage-class");
exposeHeader.add("x-obs-delete-marker");
exposeHeader.add("x-obs-expiration");
exposeHeader.add("x-obs-website-redirect-location");
exposeHeader.add("x-obs-restore");
exposeHeader.add("x-obs-version");
exposeHeader.add("x-obs-object-type");
exposeHeader.add("x-obs-next-append-position");
rule.setExposeHeader(exposeHeader);

rule.setMaxAgeSecond(100);
rules.add(rule);
cors.setRules(rules);

try
{
    obsClient.setBucketCors("bucketname", cors);
catch (ObsException e)
{
    System.out.println("HTTP Code: " + e.getResponseCode());
    System.out.println("Error Code:" + e.getErrorCode());
    System.out.println("Error Message: " + e.getErrorMessage());

    System.out.println("Request ID:" + e.getErrorRequestId());
    System.out.println("Host ID:" + e.getErrorHostId());
}
}
```

通过 OBS Python SDK 配置桶的 CORS

您可以使用OBS Python SDK调用ObsClient.setBucketCors接口配置桶的CORS，示例代码如下：

```
# 引入模块
from obs import 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。
ak = os.getenv("AccessKeyID")
```

 版权所有 © 华为技术有限公司

```
sk = os.getenv("SecretAccessKey")
# 【可选】如果使用临时AKSK和SecurityToken访问OBS，则同样推荐通过环境变量获取
security_token = os.getenv("SecurityToken")
# server填写Bucket对应的Endpoint, 这里以中国-香港为例，其他地区请按实际情况填写。
server = "https://obs.ap-southeast-1.myhuaweicloud.com"
# 创建obsClient实例
# 如果使用临时AKSK和SecurityToken访问OBS，需要在创建实例时通过security_token参数指定securityToken值
obsClient = ObsClient(access_key_id=ak, secret_access_key=sk, server=server)

from obs import CorsRule

# 指定允许跨域请求的来源
allowedOrigin = ['*']
# 指定允许跨域请求的方法
allowedMethod = ['PUT', 'POST', 'GET', 'DELETE', 'HEAD']
# 指定允许跨域请求的头域
allowedHeader = ['*']

# 指定允许跨域请求的响应中的附加头域
exposeHeader = ['ETag', 'Content-Type', 'Content-Length', 'Cache-Control', 'Content-Disposition', 'Content-Encoding', 'Content-Language', 'Expires', 'x-obs-request-id', 'x-obs-id-2', 'x-reserved-indicator', 'x-obs-api', 'x-obs-version-id', 'x-obs-copy-source-version-id', 'x-obs-storage-class', 'x-obs-delete-marker', 'x-obs-expiration', 'x-obs-website-redirect-location', 'x-obs-restore', 'x-obs-version', 'x-obs-object-type', 'x-obs-next-append-position']

maxAgeSecond = 100
cors = CorsRule(id='rule1', allowedMethod=allowedMethod,
                allowedOrigin=allowedOrigin, allowedHeader=allowedHeader,
                maxAgeSecond=maxAgeSecond, exposeHeader=exposeHeader)

resp = obsClient.setBucketCors('bucketname', corsList=[cors])
if resp.status < 300:
    print('requestId:', resp.requestId)
else:
    print('errorCode:', resp.errorCode)
    print('errorMessage:', resp.errorMessage)
```

说明

- 除OBS BrowserJS SDK外，其他语言的OBS SDK均支持配置桶的CORS；
- 您也可以参考**OBS Console配置跨域资源共享**或者**各SDK语言开发指南**的配置CORS规则章节，完成桶的CORS配置。

5.2 配置密钥

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

要接入OBS服务，您需要拥有一组有效的访问密钥（AK和SK）用来进行签名认证。具体可参考**OBS服务环境搭建**。

获取AK和SK之后，您便可以按照以下步骤进行初始化。

具体可参考**OBS服务环境搭建**。

获取AK和SK之后，您便可以按照以下步骤进行初始化。

- **创建OBS客户端**

 版权所有 © 华为技术有限公司

- **配置OBS客户端**
- **配置SDK日志**

5.3 创建 OBS 客户端

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS客户端（ObsClient）是访问OBS服务的BrowserJS客户端，它为调用者提供一系列与OBS服务进行交互的接口，用于管理、操作桶（Bucket）和对象（Object）等OBS服务上的资源。使用OBS BrowserJS SDK发起OBS请求，您需要初始化一个ObsClient实例，并根据需要修改客户端初始化配置参数。

直接创建

- 永久访问密钥（AK/SK）创建OBS客户端代码如下：

```
// 未引入AMD，直接通过构造函数创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用访问OBS
```

- 临时访问密钥（AK/SK/SecurityToken）创建OBS客户端代码如下：

```
// 未引入AMD，直接通过构造函数创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID、SecretAccessKey和SecurityToken。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    security_token: process.env.SecurityToken,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用访问OBS
```

AMD 规范创建

- 永久访问密钥（AK/SK）创建OBS客户端代码如下：

```
// 引入AMD，通过依赖注入的构造函数创建ObsClient实例
var obsClient;
define(['ObsClient'], function(ObsClient){
    obsClient = new ObsClient({
```

 版权所有 © 华为技术有限公司

```
        // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
        // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
        // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
        access_key_id: process.env.AccessKeyID,
        secret_access_key: process.env.SecretAccessKey,
        // 这里以中国-香港为例，其他地区请按实际情况填写
        server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
    });

    // 使用访问OBS
});
```

- 临时访问密钥（AK/SK/SecurityToken）创建OBS客户端代码如下：

```
// 引入AMD，通过依赖注入的构造函数创建ObsClient实例
var obsClient;
define(['ObsClient'], function(ObsClient){
    obsClient = new ObsClient({
        // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID、SecretAccessKey和SecurityToken
        // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
        // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
        access_key_id: process.env.AccessKeyID,
        secret_access_key: process.env.SecretAccessKey,
        security_token: process.env.SecurityToken,
        // 这里以中国-香港为例，其他地区请按实际情况填写
        server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
    });

    // 使用访问OBS
});
```

5.4 配置 OBS 客户端

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可通过初始化参数对ObsClient进行配置，可以配置的参数见下表：

参数	描述	建议值
access_key_id	访问密钥中的AK。	N/A
secret_access_key	访问密钥中的SK。	N/A
server	连接OBS的服务地址。可包含协议类型、域名、端口号。示例：https://your-endpoint:443。（出于安全性考虑，建议使用https协议）	N/A
timeout	HTTP/HTTPS请求的总超时时间（单位：秒）。默认为300秒。	[10，300]

 版权所有 © 华为技术有限公司

参数	描述	建议值
is_cname	是否通过自定义域名访问OBS服务。默认为false。	N/A
useRawXhr	是否使用原生XHR发送Ajax请求。默认为false。	N/A

说明

- 建议值为N/A的表示需要根据实际情况进行设置。
- 如网络状况不佳或者上传文件较大，建议增大timeout的值。

5.5 配置 SDK 日志

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.initLog开启日志功能并进行配置。示例代码如下：

```
obsClient.initLog({
    level:'warn', // 配置日志级别
});
```

说明

- SDK打印的日志均会显示在浏览器提供的开发者工具的控制台上。
- 日志功能默认是关闭的，需要主动开启。
- 您可以从**日志分析**章节获取更多关于SDK日志的信息。

版权所有 © 华为技术有限公司

6 问题定位

6.1 问题定位方法

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

使用OBS BrowserJS SDK对接OBS服务可能会遇到许多问题，您可以通过下面介绍的步骤进行问题分析和定位：

步骤1 确保使用的是OBS BrowserJS SDK的最新版本，您可以从**这里**下载最新版本；

步骤2 确保使用OBS BrowserJS SDK的程序代码遵照**OBS客户端通用示例**编写，所有ObsClient的接口调用都通过回调函数进行了异常判断，例如上传对象的示例如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Body : 'Hello OBS'
}, function (err, result) {
        if(err){
            // 异常信息不为空，表明调用接口错误，常见场景是网络异常
            console.error('Error-->' + err);
        }else{
            // 异常信息为空，表明接口调用完成，通过SDK公共结果对象进一步判断HTTP状态码
            if(result.CommonMsg.Status < 300){// HTTP状态码小于300，调用成功
                if(result.InterfaceResult){
                    // 处理调用成功后业务逻辑
```

 版权所有 © 华为技术有限公司

```
                // 可选：调用成功后，记录调用成功的HTTP状态码和服务端请求ID
                console.log('Status-->' + result.CommonMsg.Status);
                console.log('RequestId-->' + result.CommonMsg.RequestId);
            }
        }else{
            // 推荐：调用失败后，记录失败的HTTP状态码、服务端错误码、服务端请求ID等
            console.log('Status-->' + result.CommonMsg.Status);
            console.log('Code-->' + result.CommonMsg.Code); ;
            console.log('RequestId-->' + result.CommonMsg.RequestId);
        }
    }
});
```

说明

您可以从**这里**查看关于SDK公共结果对象的详细说明。

步骤3 当异常信息不为空时，首先排查客户端到OBS服务端的网络健康状况，如果网络良好，收集异常信息并联系OBS客户端运维团队定位异常原因；

步骤4 当调用ObsClient的接口失败时，从SDK公共结果对象中获取**HTTP状态码**、**OBS服务端错误码**后进行对照，排查失败原因；

步骤5 如果通过步骤4未能排查到异常原因，可从SDK公共结果对象中获取OBS服务端请求ID后联系OBS服务端运维团队定位失败原因；

步骤6 如果无法从SDK公共结果对象中获取OBS服务端请求ID，请联系OBS客户端运维团队定位失败原因。

----结束

6.2 高频问题汇总

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

SignatureDoesNotMatch 签名不匹配

```
Status-->403
Code-->SignatureDoesNotMatch
```

此错误一般是初始化ObsClient时传入的SK有误，解决方法：检查SK，确保正确。

MethodNotAllowed 方法不允许

```
Status-->405
Code-->MethodNotAllowed
```

此错误一般是请求的OBS服务端未上线ObsClient接口依赖的特性，请联系OBS运维团队进行进一步确认。

网络连接错误

```
Error: Network Error
```

此类错误一般有三种原因：

 版权所有 © 华为技术有限公司

1. 初始化ObsClient时传入的Endpoint有误，解决方法：检查Endpoint，确保正确；
2. 客户端到OBS服务端的网络异常，导致无法访问，解决方法：检查客户端到OBS服务端的网络健康状况；
3. DNS解析出的OBS服务域名无法访问，解决方法：联系OBS运维团队；
4. SDK依赖底层库Axios的兼容性问题，解决方法：使用**浏览器表单上传**，或联系OBS运维团队。

请求超时

```
timeout of xxx exceeded
```

此类错误一般有两种原因：

1. 客户端到OBS服务端的网络时延过大，解决方法：检查客户端到OBS服务端的网络健康状况；
2. 客户端到OBS服务端的网络异常，导致无法访问，解决方法：检查客户端到OBS服务端的网络健康状况。

跨域请求被拦截

```
Access to XMLHttpRequest at 'xxx' from origin 'xxx' has been blocked by CORS policy: Response to preflight request doesn't pass access control check: No 'Access-Control-Allow-Origin' header is present on the requested resource.
```

此错误均是由于未配置桶的CORS或配置的CORS有误导致，解决方法：请参考**配置桶的CORS**章节重新配置桶的CORS。

请求成功但返回结果缺少某些字段

此类错误一般有两种原因：

1. 在桶的CORS配置中ExposeHeader配置不完整，例如未配置ETag（导致上传对象成功后无法获取ETag值）、未配置x-obs-request-id（导致请求完成后无法获取OBS服务端请求ID）等，解决方法：请参考**配置桶的CORS**章节重新配置桶的CORS；
2. 使用了旧版本的SDK，解决方法：升级到最新版本的SDK，可以从**这里**下载最新版本。

IE 浏览器集成 SDK 后加载报错

```
SCRIPT5009: 'Promise'未定义
```

此错误的原因是IE浏览器不支持Promise对象（ES6规范）导致，解决方法有三种：

1. 程序中使用**esdk-obs-browserjs-x.x.x.min.js**作为SDK库文件，而不使用**esdk-obs-browserjs-without-polyfill-x.x.x.min.js**；
2. 对IE浏览器引入第三方库，补齐ES6规范，例如，引入**babel-polyfill**库；
3. 使用Chrome、Firefox等完成支持ES6规范的浏览器。

无法获取上传后的 ETag 值

使用ObsClient.putObject和ObsClient.uploadPart上传文件成功后返回结果中无ETag值，此类错误一般有两种原因：

1. 桶的CORS配置中ExposeHeader不包含ETag头域，解决方法：按照文档**配置桶的CORS**为桶配置完整的CORS配置；

 版权所有 © 华为技术有限公司

2. 桶的CORS配置中ExposeHeader包含ETag头域，但浏览器的返回结果屏蔽了ETag头域（一般发生在低版本的浏览器），解决方法：升级到高版本且完全支持HTML5的浏览器。

ObsClient 未定义

```
Uncaught ReferenceError: ObsClient is not defined
```

此类错误一般有三种原因：

1. 程序中未能正确引入SDK，解决方法：排查程序引入SDK的方式，确认是否正确引入了**esdk-obs-browserjs-x.x.x.min.js**或**esdk-obs-browserjs-without-polyfill-x.x.x.min.js**；
2. 程序中引入了AMD规范的模块化组件，例如require.js等，解决方法：使用**AMD规范创建ObsClient**；
3. 程序中引入的组件与SDK的依赖库冲突（这种场景发生概率较小），解决方法：联系OBS运维团队。

不支持 window.File 的浏览器上传文件问题

```
Error: source file must be an instance of window.File or window.Blob
```

此错误的原因是SDK依赖H5标准提供的window.File实现文件上传，如果使用不支持window.File的浏览器，例如IE8、IE9等，则无法使用ObsClient.putObject和ObsClient.uploadFile上传文件，解决方法：基于表单实现文件上传，具体步骤如下。

步骤1 判断浏览器是否支持window.File，示例代码如下：

```
function getBrowserInfo() {
    var agent = navigator.userAgent.toLowerCase();
    var regStr_ie = /msie [\d.]+;/gi;
    var regStr_ff = /firefox\/[\d.]+/gi
    var regStr_chrome = /chrome\/[\d.]+/gi;
    var regStr_saf = /safari\/[\d.]+/gi;
    var isIE = agent.indexOf('compatible') > -1 && agent.indexOf('msie' > -1);
    var isEdge = agent.indexOf('edge') > -1 && !isIE;
    var isIE11 = agent.indexOf('trident') > -1 && agent.indexOf('rv:11.0') > -1;
    if (isIE) {
        var reIE = new RegExp('msie (\\d+\\.\\d+);');
        reIE.test(agent);
        var fIEVersion = parseFloat(RegExp['$1']);
        if (fIEVersion == 7) {
            return 'IE/7';
        } else if (fIEVersion == 8) {
            return 'IE/8';
        } else if (fIEVersion == 9) {
            return 'IE/9';
        } else if (fIEVersion == 10) {
            return 'IE/10';
        }
    }
    // isIE end
    if (isIE11) {
        return 'IE/11';
    }
    // Firefox
    if (agent.indexOf('firefox') > 0) {
        return agent.match(regStr_ff);
    }
    // Safari
    if (agent.indexOf('safari') > 0 && agent.indexOf('chrome') < 0) {
        return agent.match(regStr_saf);
    }
    // Chrome
    if (agent.indexOf('chrome') > 0) {
```

 版权所有 © 华为技术有限公司

```
            return agent.match(regStr_chrome);
        }
        return '';
}

var browserInfo = getBrowserInfo();
// 判断浏览器是否支持window.File
var isSupportFileApi = browserInfo !== 'IE/7' && browserInfo !== 'IE/8' && browserInfo !== 'IE/9' &&
window.File;
```

步骤2 根据步骤1的判断结果，选择合适的上传方式，示例代码如下：

```
function postObject(){
  // 使用JS代码提交表单，实现表单上传。
}
if(isSupportFileApi){
      // 使用表单上传文件
      return postObject();
}
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
// 使用SDK的断点续传接口上传文件
obsClient.uploadFile({
  // 传入请求参数
}, function (err, result) {
  // 处理回调函数
} ) ;
```

----结束

说明

基于表单上传的文件最大不能超过5GB。

引入 CommonJS 规范导致未定义

```
Uncaught (in promise) TypeError: Cannot read property 'CancelToken' of undefined
```

此错误的原因是程序中引入了CommonJS规范的模块化组件，例如webpack等，解决方法：升级到3.19.5及以上版本SDK。

引入 Mock.js 导致未定义

```
Uncaught TypeError: request.upload.addEventListener in not a function
```

此错误的原因是程序中引入了Mock.js组件对XHR打桩导致，解决方法有两种：

1. 规避：使用SDK进行上传、下载、断点续传上传时不启用进度条功能；
2. 替换：替换Mock.js组件，使用能够全部模拟XHR接口的组件；
3. 扩展：扩展Mock.js组件，补齐Mock.js对XHR不支持的接口。

 版权所有 © 华为技术有限公司

7 管理桶

7.1 获取桶元数据

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketMetadata接口获取桶元数据。

本示例用于获取桶名为“bucketname”的元数据信息。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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


版权所有 © 华为技术有限公司

说明

- 获取桶元数据过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

7.2 判断桶是否存在

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.headBucket接口判断该桶是否已存在。

本示例用于判断桶名为“bucketname”是否存在。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

说明

- 如果返回的HTTP状态码为404，表明桶不存在。

7.3 删除桶

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

 版权所有 © 华为技术有限公司

您可以通过ObsClient.deleteBucket删除桶。以下代码展示如何删除一个桶：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 删除桶
obsClient.deleteBucket({
      Bucket: 'bucketname'
}, function(err, result) {
      if(err){
            console.error('Error-->' + err);
      }else{
            console.log('Status-->' + result.CommonMsg.Status);
      }
});
```

说明

- 如果桶不为空（包含对象或分段上传碎片），则该桶无法删除。
- 删除桶非幂等操作，删除不存在的桶会失败。

7.4 管理桶访问权限

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

桶访问权限（**ACL**）可以通过两种方式设置：

1. 调用ObsClient.setBucketAcl指定预定义访问策略。
2. 调用ObsClient.setBucketAcl直接设置。

OBS支持的桶或对象权限包含五类，见下表：

权限	描述	OBS BrowserJS SDK对应值
读权限	如果有桶的读权限，则可以获取该桶内对象列表和桶的元数据。 如果有对象的读权限，则可以获取该对象内容和元数据。	ObsClient.enums.PermissionRead

 版权所有 © 华为技术有限公司

权限	描述	OBS BrowserJS SDK对应值
写权限	如果有桶的写权限，则可以上传、覆盖和删除该桶内任何对象。 此权限在对象上不适用。	ObsClient.enums.PermissionWrite
读ACP权限	如果有读ACP的权限，则可以获取对应的桶或对象的权限控制列表（ACL）。 桶或对象的所有者永远拥有读对应桶或对象ACP的权限。	ObsClient.enums.PermissionReadAcp
写ACP权限	如果有写ACP的权限，则可以更新对应桶或对象的权限控制列表（ACL）。 桶或对象的所有者永远拥有写对应桶或对象的ACP的权限。 拥有了写ACP的权限，由于可以更改权限控制策略，实际上意味着拥有了完全访问的权限。	ObsClient.enums.PermissionWriteAcp
完全控制权限	如果有桶的完全控制权限意味着拥有读权限、写权限、读ACP权限和写ACP权限的权限。 如果有对象的完全控制权限意味着拥有读权限、读ACP权限和写ACP权限的权限	ObsClient.enums.PermissionFullControl

OBS预定义的访问策略包含五类，见下表：

权限	描述	OBS BrowserJS SDK对应值
私有读写	桶或对象的所有者拥有完全控制的权限，其他任何人都没有访问权限。	ObsClient.enums.AclPrivate
公共读	设在桶上，所有人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。 设在对象上，所有人可以获取该对象内容和元数据。	ObsClient.enums.AclPublicRead

版权所有 © 华为技术有限公司

权限	描述	OBS BrowserJS SDK对应值
公共读写	设在桶上，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据、上传对象删除对象、初始化段任务、上传段、合并段、拷贝段、取消多段上传任务。 设在对象上，所有人可以获取该对象内容和元数据。	ObsClient.enums.AclPublicReadWrite
桶公共读，桶内对象公共读	设在桶上，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据，可以获取该桶内对象的内容和元数据。 不能应用于对象。	ObsClient.enums.AclPublicReadDelivered
桶公共读写，桶内对象公共读写	设在桶上，所有人可以获取该桶内对象列表、桶内多段任务、桶的元数据、上传对象、删除对象、初始化段任务、上传段、合并段、拷贝段、取消多段上传任务，可以获取该桶内对象的内容和元数据。 不能应用于对象。	ObsClient.enums.AclPublicReadWriteDelivered

为桶设置预定义访问策略

以下代码展示如何为桶设置预定义访问策略：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 用预定义访问策略设置桶权限
obsClient.setBucketAcl({
        Bucket: 'bucketname',
        // 设置桶访问权限为私有读写
        ACL: obsClient.enums.AclPrivate
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

 版权所有 © 华为技术有限公司

说明

使用**ACL参数**指定桶的访问权限。

直接设置桶访问权限

以下代码展示如何直接设置桶访问权限：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 直接设置桶访问权限
obsClient.setBucketAcl({
        Bucket: 'bucketname',
        // 设置桶所有者
        Owner: {'ID': 'ownerid'},
        Grants: [
            // 为指定用户设置完全控制权限
            { Grantee : {Type : 'CanonicalUser',ID : 'userid'}, Permission : obsClient.enums.PermissionFullControl},
            // 为所有用户设置读权限
            { Grantee : {Type : 'Group',URI : obsClient.enums.GroupAuthenticatedUsers}, Permission : obsClient.enums.AclPublicRead}
        ]
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**Owner参数**指定桶的所有者信息；使用**Grants参数**指定被授权的用户信息。
- ACL中需要填写的所有者（Owner）或者被授权用户（Grantee）的ID，是指用户的账户ID，可通过OBS控制台“我的凭证”页面查看。
- 当前OBS桶支持的可被授权的用户组为：
  - 所有用户：ObsClient.enums.GroupAllUsers

获取桶访问权限

您可以通过ObsClient.getBucketAcl获取桶的访问权限。以下代码展示如何获取桶访问权限：

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

```
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

7.5 管理桶策略

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

除了桶访问权限外，桶的拥有者还可以通过桶策略，提供对桶和桶内对象的集中访问控制。

更多关于桶策略的内容请参考**桶策略**。

设置桶策略

您可以通过ObsClient.setBucketPolicy设置桶策略。示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
// 桶名
const bucketName = 'bucketname';
// 桶策略
const policy =  "{\"Statement\":[{\"Principal\":\"*\",\"Effect\":\"Allow\",\"Action\":\"ListBucket\",\"Resource\":\""+bucketName+"\"}]}";
// 设置桶策略
```

 版权所有 © 华为技术有限公司

```
obsClient.setBucketPolicy({
      Bucket: bucketName,
      Policy: policy
}, function(err, result) {
      if(err){
            console.error('Error-->' + err);
      }else{
            console.log('Status-->' + result.CommonMsg.Status);
      }
});
```

说明

桶策略内容的具体格式（JSON格式字符串）请参考《对象存储服务API参考》。

获取桶策略

您可以通过ObsClient.getBucketPolicy获取桶策略。示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 获取桶策略
obsClient.getBucketPolicy({
      Bucket: 'bucketname',
}, function(err, result) {
      if(err){
            console.error('Error-->' + err);
      }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                  console.log('Policy-->' + result.InterfaceResult.Policy);
            }
      }
});
```

删除桶策略

您可以通过ObsClient.deleteBucketPolicy删除桶策略。示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 删除桶策略
obsClient.deleteBucketPolicy({
```

```
    Bucket: 'bucketname'
}, function(err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

7.6 获取桶区域位置

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketLocation获取桶的区域位置。

本示例用于获取桶名为“bucketname”的区域位置信息。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

说明

- 获取桶元区域位置过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**。

版权所有 © 华为技术有限公司

7.7 获取桶存量信息

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

桶存量信息包括桶的空间大小以及桶包含的对象个数。

您可以通过ObsClient.getBucketStorageInfo获取桶的存量信息。

本示例用于获取桶名为“bucketname”的存量信息。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

说明

- 获取桶存量信息过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

7.8 桶配额

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

版权所有 © 华为技术有限公司

设置桶配额

您可以通过ObsClient.setBucketQuota设置桶配额。以下代码展示如何设置桶配额：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

//设置桶配额为100MB
obsClient.setBucketQuota({
        Bucket: 'bucketname',
        StorageQuota: 1024 * 1024 * 100
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**StorageQuota参数**指定桶的配额大小。
- 桶配额值必须为非负整数，单位为字节，支持的最大值为2^{63} - 1。

获取桶配额

您可以通过ObsClient.getBucketQuota获取桶配额。以下代码展示如何获取桶配额：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

 版权所有 © 华为技术有限公司

7.9 桶存储类型

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS允许您对桶配置不同的存储类型，桶中对象的存储类型默认将与桶的存储类型保持一致。不同的存储类型可以满足客户业务对存储性能、成本的不同诉求。桶的存储类型分为三类，见下表：

类型	说明	OBS BrowserJS SDK对应值
标准存储	标准存储拥有低访问时延和较高的吞吐量，适用于有大量热点对象（平均一个月多次）或小对象（<1MB），且需要频繁访问数据的业务场景。	ObsClient.enums.StorageClassStandard
低频访问存储	低频访问存储适用于不频繁访问（平均一年少于12次）但在需要时也要求能够快速访问数据的业务场景。	ObsClient.enums.StorageClassWarm
归档存储	归档存储适用于很少访问（平均一年访问一次）数据的业务场景。	ObsClient.enums.StorageClassCold

更多关于桶存储类型的内容请参考**桶的存储类别**。

设置桶存储类型

您可以通过ObsClient.setBucketStoragePolicy设置桶存储类型。以下代码展示如何设置桶存储类型：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketStoragePolicy({
        Bucket: 'bucketname',
        StorageClass: obsClient.enums.StorageClassWarm
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
```


版权所有 © 华为技术有限公司

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

说明

使用**StorageClass参数**指定桶的存储类型。

获取桶存储类型

您可以通过ObsClient.getBucketStoragePolicy获取桶存储类型。以下代码展示如何获取桶存储类型：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

8 上传对象

8.1 对象上传简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

在OBS中，用户操作的基本数据单元是对象。OBS BrowserJS SDK提供了丰富的对象上传接口，可以通过以下方式上传对象：

- **文本上传**
- **文件上传**
- **分段上传**
- **追加上传**
- **基于表单上传**

SDK支持上传0KB~5GB的对象。文件上传的内容大小不能超过5GB；当上传较大文件时，请使用分段上传，分段上传每段内容大小不能超过5GB；基于表单上传提供了基于浏览器表单上传对象的方式。

如果上传的对象权限设置为匿名用户读取权限，对象上传成功后，匿名用户可通过链接地址访问该对象数据。对象链接地址格式为：https://*桶名.域名/文件夹目录层级/对象名*。如果该对象存在于桶的根目录下，则链接地址将不需要有文件夹目录层级。

8.2 文本上传

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

版权所有 © 华为技术有限公司

文本上传用于直接上传字符串。您可以通过ObsClient.putObject直接上传字符串到OBS。

本示例用于上传字符串“Hello OBS”到桶名为“bucketname”里，名称为“objectname”。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket: 'bucketname',
        // 对象名，是对象在桶中的完整路径，路径中不包含桶名。
        Key: 'objectname',
        // 待上传对象的内容。
        Body: 'Hello OBS'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**Body参数**指定待上传的字符串。

8.3 文件上传

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档的“上传对象”章节**详细介绍了参数和使用方法。

文件上传使用本地文件作为对象的数据源。以下代码展示了如何进行文件上传：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
```

 版权所有 © 华为技术有限公司

```
obsClient.putObject({
    Bucket: 'bucketname',
    Key: 'objectname',
    SourceFile: document.getElementById('input-file').files[0]
}, function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

- 使用**SourceFile参数**指定待上传的文件，其必须是File对象或者Blob对象，例如在HTML页面中使用类型为“file”的input标签指定待上传的文件：<input type="file" id="input-file"/>。
- **SourceFile参数**和**Body参数**不能同时使用。
- 上传的内容大小不能超过5GB。

须知

- 浏览器必须支持window.File特性，否则无法正常使用文件上传功能；
- 请参考**不支持window.File的浏览器上传文件问题**，解决不支持window.File的浏览器上传文件的问题；

8.4 获取上传进度

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过设置回调函数来获取上传的进度。

本示例用于上传文件到桶名为“bucketname”里，名称为“objectname”的对象并通过ProgressCallback获取上传进度。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var callback = function(transferredAmount, totalAmount, totalSeconds){
    // 获取上传平均速率（KB/S）
```

```
    console.log(transferredAmount * 1.0 / totalSeconds / 1024);
    // 获取上传进度百分比
    console.log(transferredAmount * 100.0 / totalAmount);
};

obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'objectname',
        SourceFile: document.getElementById('input-file').files[0],
        ProgressCallback: callback
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 支持获取上传进度的接口包括：文本上传、文件上传、上传段、追加上传和断点续传上传。

8.5 创建文件夹

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS本身是没有文件夹的概念的，桶中存储的元素只有对象。创建文件夹实际上是创建了一个大小为0且对象名以“/”结尾的对象，这类对象与其他对象无任何差异，可以进行下载、删除等操作，只是OBS控制台会将这类以“/”结尾的对象以文件夹的方式展示。

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'parent_directory/'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
// 在文件夹下创建对象
obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'parent_directory/objectname'
```

```
}, function (err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

- 创建文件夹本质上来说是创建了一个大小为0且对象名以“/”结尾的对象。
- 多级文件夹创建最后一级即可，比如src1/src2/src3/，创建src1/src2/src3/即可，无需创建src1/、src1/src2/。

8.6 设置对象属性

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以在上传对象时设置对象属性。对象属性包含对象长度、对象MIME类型、对象MD5值（用于校验）、对象存储类型、对象自定义元数据。对象属性可以在多种上传方式下（文本上传、文件上传、分段上传等），或**复制对象**时进行设置。

对象属性详细说明见下表：

名称	描述	默认值
对象长度（Content-Length）	上传对象的长度，超过文件的长度会截断。	文件实际长度
对象MIME类型（Content-Type）	对象的MIME类型，定义对象的类型及网页编码，决定浏览器将以什么形式、什么编码读取对象。	binary/octet-stream
对象MD5值（Content-MD5）	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。	无
对象存储类型	对象的存储类型，不同的存储类型可以满足客户业务对存储性能、成本的不同诉求。默认与桶的存储类型保持一致，可以设置为与桶的存储类型不同。	无
对象自定义元数据	用户对上传到桶中对象的自定义属性描述，以便对对象进行自定义管理。	无

版权所有 © 华为技术有限公司

设置对象长度

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'objectname',
        SourceFile: document.getElementById('input-file').files[0],
        ContentLength: 1024 * 1024 // 1MB
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

使用**ContentLength参数**指定对象长度。

设置对象 MIME 类型

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 上传图片
obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'objectname.jpg',
        SourceFile: document.getElementById('input-file').files[0],
        ContentType: 'image/jpeg'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**ContentType参数**指定对象MIME类型。
- 如果不设置对象MIME类型，SDK会根据上传对象的后缀名自动判断对象MIME类型，如.xml判断为application/xml文件；.html判断为text/html文件。

 版权所有 © 华为技术有限公司

设置对象 MD5 值

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        SourceFile : document.getElementById('input-file').files[0],
        ContentMD5 : 'your md5 which should be encoded by base64'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**ContentMD5参数**指定对象MD5值。
- 对象数据的MD5值必须经过Base64编码。
- OBS服务端会将该MD5值与对象数据计算出的MD5值进行对比，如果不匹配则上传失败，返回HTTP 400错误。
- 如果不设置对象的MD5值，OBS服务端会忽略对对象数据的MD5值校验。

设置对象存储类型

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        SourceFile : document.getElementById('input-file').files[0],
        // 设置对象存储类型为归档存储
        StorageClass : ObsClient.enums.StorageClassCold
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

 版权所有 © 华为技术有限公司

说明

- 使用**StorageClass参数**指定对象的存储类型。
- 如果不设置，对象的存储类型默认与桶的存储类型保持一致。
- 对象的存储类型分为三类，其含义与**桶存储类型**一致。
- 下载归档存储类型的对象前必须将其恢复。

设置对象自定义元数据

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        SourceFile : document.getElementById('input-file').files[0],
        Metadata : {'property1':'property-value1', 'property2' : 'property-value2'},
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**Metadata参数**指定对象自定义元数据。
- 在上面设置对象自定义元数据示例代码中，用户自定义了一个名称为“property1”，值为“property-value1”的元数据和一个名称为“property2”，值为“property-value2”的元数据。
- 一个对象可以有多个元数据，总大小不能超过8KB。
- 对象的自定义元数据可以通过ObsClient.getObjectMetadata获取，参见**获取对象元数据**。
- 使用ObsClient.getObject下载对象时，对象的自定义元数据也会同时下载。

8.7 分段上传

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

对于较大文件上传，可以切分成段上传。用户可以在如下的应用场景内（但不仅限于此），使用分段上传的模式：

 版权所有 © 华为技术有限公司

- 上传超过100MB大小的文件。
- 网络条件较差，和OBS服务端之间的链接经常断开。
- 上传前无法确定将要上传文件的大小。

分段上传分为如下3个步骤：

步骤1 初始化分段上传任务（ObsClient.initiateMultipartUpload）。

步骤2 逐个或并行上传段（ObsClient.uploadPart）。

步骤3 合并段（ObsClient.completeMultipartUpload）或取消分段上传任务(ObsClient.abortMultipartUpload)。

----结束

说明

您也可以直接使用SDK提供的**断点续传上传**接口（对分段上传的封装和加强）实现分段上传。

初始化分段上传任务

使用分段上传方式传输数据前，必须先通知OBS初始化一个分段上传任务。该操作会返回一个OBS服务端创建的全局唯一标识（Upload ID），用于标识本次分段上传任务。您可以根据这个唯一标识来发起相关的操作，如取消分段上传任务、列举分段上传任务、列举已上传的段等。

您可以通过ObsClient.initiateMultipartUpload初始化一个分段上传任务：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

说明

- 初始化分段上传任务时，除了指定上传对象的名称和所属桶外，您还可以使用**ContentType参数**和**Metadata参数**分别指定对象MIME类型和对象自定义元数据。
- 调用初始化分段上传任务接口成功后，会返回分段上传任务的全局唯一标识（Upload ID），在后面的操作中将用到它。

上传段

初始化一个分段上传任务之后，可以根据指定的对象名和Upload ID来分段上传数据。每一个上传的段都有一个标识它的号码——分段号（Part Number，范围是1~10000）。对于同一个Upload ID，该分段号不但唯一标识这一段数据，也标识了这段数据在整个对象内的相对位置。如果您用同一个分段号上传了新的数据，那么OBS上已有的这个段号的数据将被覆盖。**除了最后一段以外，其他段的大小范围100KB~5GB**；最后段大小范围是0~5GB。每个段不需要按顺序上传，甚至可以在不同进程、不同机器上上传，OBS会按照分段号排序组成最终对象。

您可以通过ObsClient.uploadPart上传段：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

const bucketname = 'examplebucket';
const objectname = 'exampleobject';
const PartSize = 5 * 1024 * 1024;
const Uploadid = 'upload id from initiateMultipartUpload';
const file = document.getElementById('input-file').files[0];
const lastPartSize = file.size % PartSize;
// 段数量
const count = Math.ceil(file.size / PartSize);
// 上传第n段
const uploadPart = (n) => {
    obsClient.uploadPart({
        Bucket: bucketname,
        Key: objectname,
        // 设置分段号，范围是1~10000
        PartNumber: n,
        // 设置Upload ID
        UploadId,
        // 设置将要上传的大文件
        SourceFile: file,
        // 设置分段大小
        PartSize: count === n ? lastPartSize : PartSize,
        // 设置分段的起始偏移大小
        Offset: (n-1) * PartSize
    }, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                console.log('ETag-->' + result.InterfaceResult.ETag);
            }
        }
```

 版权所有 © 华为技术有限公司

```
    });
}

// 上传第1段
uploadPart(1);
```

⚠ 注意

如果ETag值获取是undefined，则需要配置CORS规则，将ETag添加到附加头域中即可，参考**•ETag**。

说明

- 使用**PartNumber参数**指定分段号；使用**UploadId参数**指定分段上传任务的全局唯一标识；使用**SourceFile参数**指定待上传的文件；使用**PartSize参数**指定分段大小；使用**Offset参数**指定待上传文件的起始偏移大小。
- **SourceFile参数**必须是File对象或者Blob对象，例如在HTML页面中使用类型为“file”的input标签指定待上传的文件：<input type="file" id="input-file"/>。
- 上传段接口要求除最后一段以外，其他的段大小都要大于100KB。但是上传段接口并不会立即校验上传段的大小（因为不知道是否为最后一块）；只有调用合并段接口时才会校验。
- OBS会将服务端收到段数据的ETag值（段数据的MD5值）返回给用户。
- 可以通过**ContentMD5参数**设置上传数据的MD5值，提供给OBS服务端用于校验数据完整性。
- 分段号的范围是1~10000。如果超出这个范围，OBS将返回400 Bad Request错误。
- OBS 3.0的桶支持最小段的大小为100KB，OBS 2.0的桶支持最小段的大小为5MB。请在OBS 3.0的桶上执行分段上传操作。

合并段

所有分段上传完成后，需要调用合并段接口来在OBS服务端生成最终对象。在执行该操作时，需要提供所有有效的分段列表（包括分段号和分段ETag值）；OBS收到提交的分段列表后，会逐一验证每个段的有效性。当所有段验证通过后，OBS将把这些分段组合成最终的对象。

您可以通过ObsClient.completeMultipartUpload合并段：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.completeMultipartUpload({
        Bucket:'bucketname',
        Key:'objectname',
        // 设置Upload ID
        UploadId:'upload id from initiateMultipartUpload',
        Parts: [{'PartNumber':1,'ETag':'etag value from uploadPart'}]
}, function (err, result) {
        if(err){
```

```
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

⚠ 注意

- 如果最后一个段之外的其它段尺寸过小（小于100KB），OBS返回400 Bad Request。

说明

- 使用**UploadId参数**指定分段上传任务的全局唯一标识；使用**Parts参数**指定分段号与分段ETag值的列表，该列表必须按分段号升序排列。
- 分段可以是不连续的。

取消分段上传任务

分段上传任务可以被取消，当一个分段上传任务被取消后，就不能再使用其Upload ID做任何操作，已经上传段也会被OBS删除。

采用分段上传方式上传对象过程中或上传对象失败后会在桶内产生段，这些段会占用您的存储空间，您可以通过取消该分段上传任务来清理掉不需要的段，节约存储空间。

您可以通过ObsClient.abortMultipartUpload取消分段上传任务：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.abortMultipartUpload({
        Bucket:'bucketname',
        Key:'objectname',
        // 设置Upload ID
        UploadId:'upload id from initiateMultipartUpload',
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

列举已上传的段

您可使用ObsClient.listParts列举出某一分段上传任务所有已经上传成功的段。

该接口可设置的参数如下：

 版权所有 © 华为技术有限公司

参数	作用
UploadId	分段上传任务全局唯一标识，从ObsClient.initiateMultipartUpload返回的结果获取。
MaxParts	表示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	表示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。

- 简单列举

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 列举已上传的段，其中uploadId来自于initiateMultipartUpload
obsClient.listParts({
        Bucket : 'bucketname',
        Key: 'objectname',
        UploadId : 'upload id from initiateMultipartUpload'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                for(var i in result.InterfaceResult.Parts){
                    console.log('Part['+ i +']:');
                    // 分段号，上传时候指定
                    console.log('PartNumber-->' + result.InterfaceResult.Parts[i]['PartNumber']);
                    // 段的最后上传时间
                    console.log('LastModified-->' + result.InterfaceResult.Parts[i]['LastModified']);
                    // 分段的ETag值
                    console.log('ETag-->' + result.InterfaceResult.Parts[i]['ETag']);
                    // 段数据大小
                    console.log('Size-->' + result.InterfaceResult.Parts[i]['Size']);
                }
            }
        }
});
```

说明

- 列举段至多返回1000个段信息，如果指定的Upload ID包含的段数量大于1000，则返回结果中InterfaceResult.IsTruncated为true表明本次没有返回全部段，并可通过InterfaceResult.NextPartNumberMarker获取下次列举的起始位置。
- 如果想获取指定Upload ID包含的所有分段，可以采用分页列举的方式。

- 列举所有段

由于ObsClient.listParts只能列举至多1000个段，如果段数量大于1000，列举所有分段请参考如下示例：

 版权所有 © 华为技术有限公司

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var listAll = function (partNumberMarker) {
      // 列举已上传的段，其中uploadId来自于initiateMultipartUpload
      obsClient.listParts({
            Bucket : 'bucketname',
            Key: 'objectname',
            UploadId : 'upload id from initiateMultipartUpload',
            PartNumberMarker : partNumberMarker
      }, function (err, result) {
            if(err){
                  console.log('Error-->' + err);
            }else{
                  console.log('Status-->' + result.CommonMsg.Status);
                  if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                        for(var i in result.InterfaceResult.Parts){
                              console.log('Part['+ i +']:');
                              // 分段号，上传时候指定
                              console.log('PartNumber-->' + result.InterfaceResult.Parts[i]['PartNumber']);
                              // 段的最后上传时间
                              console.log('LastModified-->' + result.InterfaceResult.Parts[i]['LastModified']);
                              // 分段的ETag值
                              console.log('ETag-->' + result.InterfaceResult.Parts[i]['ETag']);
                              // 段数据大小
                              console.log('Size-->' + result.InterfaceResult.Parts[i]['Size']);
                        }
                        if(result.InterfaceResult.IsTruncated === 'true'){
                              listAll(result.InterfaceResult.NextPartNumberMarker);
                        }
                  }
            }
      });
};

listAll();
```

列举分段上传任务

您可以通过ObsClient.listMultipartUploads列举分段上传任务。列举分段上传任务可设置的参数如下：

参数	作用
Prefix	限定返回的分段上传任务中的对象名必须带有Prefix前缀。
Delimiter	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。
MaxUploads	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。

 版权所有 © 华为技术有限公司

参数	作用
KeyMarker	表示列举时返回指定的KeyMarker之后的分段上传任务。
UploadIdMarker	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。

- 简单列举分段上传任务

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listMultipartUploads({
        Bucket : 'bucketname'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                for(var i in result.InterfaceResult.Uploads){
                    console.log('Uploads[' + i + ']');
                    console.log('UploadId-->' + result.InterfaceResult.Uploads[i]['UploadId']);
                    console.log('Key-->' + result.InterfaceResult.Uploads[i]['Key']);
                    console.log('Initiated-->' + result.InterfaceResult.Uploads[i]['Initiated']);
                }
            }
        }
});
```

说明

- 列举分段上传任务至多返回1000个任务信息，如果指定的桶包含的分段上传任务数量大于1000，则返回结果中InterfaceResult.IsTruncated为true表明本次没有返回全部结果，并可通过InterfaceResult.NextKeyMarker和InterfaceResult.NextUploadIdMarker获取下次列举的起点。
- 如果想获取指定桶包含的所有分段上传任务，可以采用分页列举的方式。

- 列举全部分段上传任务

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
```

 版权所有 © 华为技术有限公司

```
var listAll = function (keyMarker, uploadIdMarker) {
    obsClient.listMultipartUploads({
        Bucket : 'bucketname',
        KeyMarker : keyMarker,
        UploadIdMarker : uploadIdMarker
    }, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                for(var i in result.InterfaceResult.Uploads){
                    console.log('Uploads[' + i + ']');
                    console.log('UploadId-->' + result.InterfaceResult.Uploads[i]['UploadId']);
                    console.log('Key-->' + result.InterfaceResult.Uploads[i]['Key']);
                    console.log('Initiated-->' + result.InterfaceResult.Uploads[i]['Initiated']);
                }

                if(result.InterfaceResult.IsTruncated === 'true'){
                    listAll(result.InterfaceResult.NextKeyMarker,
result.InterfaceResult.NextUploadIdMarker);
                }
            }
        }
    });
}

listAll();
```

8.8 设置对象生命周期

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

上传对象或者初始化分段上传任务时，您可以直接指定对象的过期时间。示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 上传对象时，设置对象30天后过期
obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Body : 'Hello OBS',
        Expires : 30
}, function(err, result){
        if(err){
            console.error('Error-->' + err);
```

 版权所有 © 华为技术有限公司

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

// 初始化分段上传任务时，设置合并段后生成的对象60天后过期
obsClient.initiateMultipartUpload({
    Bucket : 'bucketname',
    Key : 'objectname',
    ContentType : 'text/plain',
    Expires : 60
}, function(err, result) {
    if(err){
        console.error('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
        if(result.CommonMsg.Status < 300 && result.InterfaceResult){
            console.log('UploadId-->' + result.InterfaceResult.UploadId);
        }
    }
});
```

说明

- 使用**Expires参数**指定对象的过期时间。
- 上述方式仅支持设置以天为单位的对象过期时间，过期后的对象会被OBS服务端自动清理。
- 上述方式设置的对象过期时间，其优先级高于桶生命周期规则。

8.9 追加上传

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

追加上传可实现对同一个对象追加数据内容的功能。您可以通过ObsClient.appendObject进行追加上传。示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 第一次追加上传，起始位置必须为0
obsClient.appendObject({
    Bucket:'bucketname',
    Key:'objectname',
    Position : 0,
    Body : 'Hello OBS'
}).then(function(result){
    console.log('Status-->' + result.CommonMsg.Status);
    if(result.CommonMsg.Status < 300 && result.InterfaceResult){
        console.log('NextPosition-->' + result.InterfaceResult.NextPosition);
```

 版权所有 © 华为技术有限公司

```
    }
    // 第二次追加上传
    obsClient.appendObject({
        Bucket:'bucketname',
        Key:'objectname',
        Position : result.InterfaceResult.NextPosition,
        Body : 'Hello OBS Again'
    }, function(err, result2){
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result2.CommonMsg.Status);
            if(result2.CommonMsg.Status < 300 && result2.InterfaceResult){
                console.log('NextPosition-->' + result2.InterfaceResult.NextPosition);
            }
        }
    });

    // 通过获取对象属性接口获取下次追加上传的位置
    obsClient.getObjectMetadata({
        Bucket:'bucketname',
        Key:'objectname',
    }).then(function(result3){
        console.log('Status-->' + result3.CommonMsg.Status);
        if(result3.CommonMsg.Status < 300 && result3.InterfaceResult){
            console.log('RequestId-->' + result3.InterfaceResult.RequestId);
            console.log('NextPosition-->' + result3.InterfaceResult.NextPosition);
        }
    }).catch(function(err){
        console.error('err:' + err);
    });

}).catch(function(err){
    console.error('err:' + err);
});
```

说明

- 使用**Position参数**指定追加上传的位置，且第一次追加上传的位置必须为0。
- ObsClient.putObject上传的对象可覆盖ObsClient.appendObject上传的对象，覆盖后对象变为普通对象，不可再进行追加上传。
- 第一次调用追加上传时，如果已存在同名的普通对象，则会抛出异常（HTTP状态码为409）。
- 追加上传返回的ETag是当次追加数据内容的ETag，不是完整对象的ETag。
- 单次追加上传的内容不能超过5GB，且最多支持10000次追加上传。
- 追加上传成功后，可通过返回结果中InterfaceResult.NextPosition获取下次追加上传的位置；或者通过ObsClient.getObjectMetadata接口获取下次追加上传的位置。

8.10 分段复制

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

分段复制是分段上传的一种特殊情况，即分段上传任务中的段通过复制OBS指定桶中现有对象（或对象的一部分）来实现。

您可以通过ObsClient.copyPart来复制段。

 版权所有 © 华为技术有限公司

本示例用于分段复制桶名为“sourcebucketname”里的“sourceobjectname”对象到桶名为“destbucketname”里的“destobjectname”对象。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var destBucketName = 'destbucketname';
var destObjectKey = 'destobjectname';
var sourceBucketName = 'sourcebucketname';
var sourceObjectKey = 'sourceobjectname';

obsClient.CopyPart({
        Bucket : destBucketName,
        Key : destObjectKey,
        // 设置分段号，范围是1~10000
        PartNumber : 1,
        // 设置Upload ID
        UploadId : 'upload id from initiateMultipartUpload',
        // 设置待复制的源对象
        CopySource : sourceBucketName + '/' + sourceObjectKey,
        // 设置待复制的源对象的字节范围
        CopySourceRange : 'bytes=0-100'
}, function(err, result) {
        if (err) {
            console.log('Error-->' + err);
        } else {
            console.log('Status-->' + result.CommonMsg.Status);
            if (result.CommonMsg.Status < 300 && result.InterfaceResult) {
                console.log('RequestId-->' + result.InterfaceResult.RequestId);
                console.log('LastModified-->' + result.InterfaceResult.LastModified);
                console.log('ETag-->' + result.InterfaceResult.ETag);
            }
        }
});
```

说明

- 复制段时，使用**PartNumber参数**指定分段号；使用**UploadId参数**指定分段上传任务的全局唯一标识；使用**CopySource参数**指定复制时的源对象信息；使用**CopySourceRange参数**指定待复制的源对象的字节范围。

8.11 断点续传上传

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

断点续传上传的原理是将待上传的文件分成若干个分段分别上传，并实时地将每段上传结果统一记录在断点续传记录对象中，仅当所有分段都上传成功时返回上传成功的

 版权所有 © 华为技术有限公司

结果，否则在回调函数中返回错误码提醒用户通过传入断点续传记录对象再次调用接口进行重新上传。

⚠ 注意

- 断点续传上传接口传入的文件总大小至少要100K以上。
- 浏览器刷新网页后，断点续传会失效，需重新上传文件。

您可以通过ObsClient.uploadFile进行断点续传上传。该接口可设置的主要参数如下：

参数	作用
Bucket	桶名。
Key	对象名。
RequestDate	指定请求时间。 **说明** 当为String类型时，必须符合ISO8601或RFC822规范。
SourceFile	待上传的源文件（浏览器必须支持FileReader）。
UploadCheckpoint	断点续传记录对象，可通过ResumeCallback回调函数获取到。
PartSize	分段大小，单位：字节。 取值范围是100KB~5GB，默认为9MB。
TaskNum	分段上传时的最大并发数，默认为1。
ProgressCallback	获取上传进度的回调函数。 **说明** 该回调函数依次包含三个参数：已上传的字节数、总字节数、已使用的时间（单位：秒）。
EventCallback	获取上传事件的回调函数。 **说明** 该回调函数依次包含三个参数：事件类型，事件参数，事件结果。
ResumeCallback	获取断点续传控制参数的回调函数。 **说明** • 该回调函数依次包含两个参数：取消断点续传上传任务控制参数，断点续传记录对象； • 可以调用取消断点续传上传任务控制参数的cancel方法来暂停断点续传上传任务。

以下代码展示了如何使用断点续传上传接口上传文件：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
```


版权所有 © 华为技术有限公司

```
    // 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
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var cp;
var hook;
obsClient.uploadFile({
        Bucket : 'bucketname',
        Key : 'objectname',
        SourceFile : document.getElementById('input-file').files[0],
        PartSize : 9 * 1024 * 1024,
        ProgressCallback : function(transferredAmount, totalAmount, totalSeconds){
            console.log(transferredAmount * 1.0 / totalSeconds / 1024);
            console.log(transferredAmount * 100.0 / totalAmount);
            if(hook && (transferredAmount / totalAmount) > 0.5){
                // 暂停断点续传任务
                // hook.cancel();
            }
        },
        EventCallback : function(eventType, eventParam, eventResult){
            // 处理事件响应
        },
        ResumeCallback : function(resumeHook, uploadCheckpoint){
            // 获取取消断点续传上传任务控制参数
            hook = resumeHook;
            // 记录断点
            cp = uploadCheckpoint;
        }
}, function(err, result){
    console.error('Error-->' + err);
    // 出现错误，再次调用断点续传接口，继续上传任务
    if(err){
        obsClient.uploadFile({
            UploadCheckpoint : cp,
            ProgressCallback : function(transferredAmount, totalAmount, totalSeconds){
                console.log(transferredAmount * 1.0 / totalSeconds / 1024);
                console.log(transferredAmount * 100.0 / totalAmount);
            },
            EventCallback : function(eventType, eventParam, eventResult){
              // 处理事件响应
            },
         }, function(err, result){
            if(err){
                console.error('Error-->' + err);
            }else{
                if(result.CommonMsg.Status < 300){
                    console.log('RequestId-->' + result.InterfaceResult.RequestId);
                    console.log('Bucket-->' + result.InterfaceResult.Bucket);
                    console.log('Key-->' + result.InterfaceResult.Key);
                    console.log('Location-->' + result.InterfaceResult.Location);
                }else{
                    console.log('Code-->' + result.CommonMsg.Code);
                    console.log('Message-->' + result.CommonMsg.Message);
                }
            }
        });
    }else {
        console.log('Status-->' + result.CommonMsg.Status);
        if (result.CommonMsg.Status < 300 && result.InterfaceResult) {
            console.log('RequestId-->' + result.InterfaceResult.RequestId);
        }
    }
});
```

 版权所有 © 华为技术有限公司

说明

- 断点续传上传接口是利用**分段上传**特性实现的，是对分段上传的封装和加强；
- 通过ResumeCallback回调函数来获取断点记录对象，该记录对象中包含sourceFile字段代表待上传的文件，如果浏览器重启后该字段需要调用者重新进行设置。

8.12 基于表单上传

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

基于表单上传是使用HTML表单形式上传对象到指定桶中，对象最大不能超过5GB。

您可以通过ObsClient.createPostSignatureSync生成基于表单上传的请求参数。使用BrowserJS代码模拟表单上传的完整代码示例，参见**post-object-sample**。您也可以通过如下步骤进行表单上传：

步骤1 使用ObsClient.createPostSignatureSync生成用于鉴权的请求参数。

步骤2 准备表单HTML页面。

步骤3 将生成的请求参数填入HTML页面。

步骤4 选择本地文件，进行表单上传。

----结束

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。

以下代码展示了如何生成基于表单上传的请求参数：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 设置表单参数
var formParams = {
            // 设置对象访问权限为公共读
            'x-obs-acl': obsClient.enums.AclPublicRead,
            // 设置对象MIME类型
            'content-type': 'text/plain'
};
```

```
// 设置表单上传请求有效期，单位：秒
var expires = 3600;

var res = obsClient.createPostSignatureSync({Expires:expires, FormParams: formParams});

// 获取表单上传请求参数
console.log('\t' + res.Policy);
console.log('\t' + res.Signature);
```

示例表单HTML代码如下：

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8" />
</head>
<body>

<form action="http://bucketname.your-endpoint/" method="post" enctype="multipart/form-data">
Object key
<!-- 对象名 -->
<input type="text" name="key" value="objectname" />
<p>
ACL
<!-- 对象ACL权限 -->
<input type="text" name="x-obs-acl" value="public-read" />
<p>
Content-Type
<!-- 对象MIME类型 -->
<input type="text" name="content-type" value="text/plain" />
<p>
<!-- policy的base64编码值 -->
<input type="hidden" name="policy" value="*** Provide your policy ***" />
<!-- AK -->
<input type="hidden" name="AccessKeyId" value="*** Provide your access key ***"/>
<!-- 签名串信息 -->
<input type="hidden" name="signature" value="*** Provide your signature ***"/>

<input name="file" type="file" />
<input name="submit" value="Upload" type="submit" />
</form>
</body>
</html>
```

说明

- HTML表单中的policy，signature的值均是从ObsClient.createPostSignatureSync的返回结果中获取。
- 您可以直接下载表单HTML示例**PostDemo**。

 版权所有 © 华为技术有限公司

9 下载对象

9.1 对象下载简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS BrowserJS SDK提供了丰富的对象下载接口，可以通过以下方式下载对象：

- **文本下载**
- **二进制式下载**
- **文件下载**
- **范围下载**
- **限定条件下载**

您可以通过ObsClient.getObject下载对象。

9.2 文本下载

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

本示例用于下载桶名为“bucketname”里，名称为“objectname”的对象。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境
```


版权所有 © 华为技术有限公司

```
变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 读取对象内容
                console.log('Object Content:');
                console.log(result.InterfaceResult.Content);
            }
        }
});
```

说明

- 文本下载方式下返回结果中的InterfaceResult.Content是一个String对象。

9.3 二进制式下载

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

以下代码展示了如何进行二进制式下载：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        SaveByType : 'arraybuffer'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
```

```
            // 读取对象字节长度
            console.log('Object Length:\n');
            console.log(result.InterfaceResult.Content.byteLength);
        }
    }
});
```

说明

- 设置**SaveByType参数**为“arraybuffer”指定使用二进制式下载。
- 二进制式下载方式下返回结果中的InterfaceResult.Content是一个ArrayBuffer对象。

9.4 文件下载

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

本示例用于下载桶名为“bucketname”里，名称为“objectname”的对象，并保存到“file”。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        SaveByType : 'file'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取下载对象的路径
                console.log('Download Path:');
                console.log(result.InterfaceResult.Content.SignedUrl);
            }
        }
});
```

说明

- 设置**SaveByType参数**为“file”生成文件下载路径。

版权所有 © 华为技术有限公司

9.5 范围下载

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

如果只需要下载对象的其中一部分数据，可以使用范围下载，下载指定范围的数据。如果指定的下载范围是0~1000，则返回第0到第1000个字节的数据，包括第1000个，共1001字节的数据，即[0， 1000]。如果指定的范围无效，则返回整个对象的数据。以下代码展示了如何进行范围下载：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        // 指定下载范围
        Range : 'bytes=0-1000'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 读取对象内容
                console.log('Object Content:');
                console.log(result.InterfaceResult.Content);
            }
        }
});
```

说明

- 使用**Range参数**指定下载范围，格式为“bytes=x-y”。
- 如果指定的范围无效（比如开始位置、结束位置为负数，大于文件大小），则会返回整个对象。

9.6 获取下载进度

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

版权所有 © 华为技术有限公司

您可以通过设置回调函数来获取下载的进度。

本示例用于下载桶名为“bucketname”里，名称为“objectname”的对象并通过ProgressCallback监控下载进度。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var callback = function(transferredAmount, totalAmount, totalSeconds){
    // 获取下载平均速率（KB/S）
    console.log(transferredAmount * 1.0 / totalSeconds / 1024);
    // 获取下载进度百分比
    console.log(transferredAmount * 100.0 / totalAmount);
};

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        ProgressCallback: callback
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 读取对象内容
                console.log('Object Content:');
                console.log(result.InterfaceResult.Content);
            }
        }
});
```

说明

- 支持获取下载进度的接口包括：文本下载和二进制式下载。

9.7 限定条件下载

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

下载对象时，可以指定一个或多个限定条件，满足限定条件时则进行下载，否则返回异常码，下载对象失败。

您可以使用的限定条件如下：

 版权所有 © 华为技术有限公司

参数	作用	格式
IfModifiedSince	如果对象在指定的时间后有修改，则返回对象内容，否则返回错误。	符合http://www.ietf.org/rfc/rfc2616.txt规定格式的HTTP时间字符串。
IfUnmodifiedSince	如果对象在指定的时间后没有修改，则返回对象内容，否则返回错误。	符合http://www.ietf.org/rfc/rfc2616.txt规定格式的HTTP时间字符串。
IfMatch	如果对象的ETag值与该参数值相同，则返回对象内容，否则返回异常码。	字符串。
IfNoneMatch	如果对象的ETag值与该参数值不相同，则返回对象内容，否则返回异常码。	字符串。

说明

- 对象的ETag值是指对象数据的MD5校验值。
- 如果包含IfUnmodifiedSince并且不符合或者包含IfMatch并且不符合，则下载对象失败，返回异常码：412 precondition failed。
- 如果包含IfModifiedSince并且不符合或者包含IfNoneMatch并且不符合，则下载对象失败，返回异常码：304 Not Modified。

以下代码展示了如何进行限定条件下载：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        IfModifiedSince : 'Thu, 31 Dec 2015 16:00:00 GMT'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 读取对象内容
                console.log('Object Content:');
                console.log(result.InterfaceResult.Content);
            }
        }
});
```

 版权所有 © 华为技术有限公司

9.8 重写响应头

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

下载对象时，可以重写部分HTTP/HTTPS响应头信息。可重写的响应头信息见下表：

参数	作用
ResponseContentType	重写HTTP/HTTPS响应中的Content-Type
ResponseContentLanguage	重写HTTP/HTTPS响应中的Content-Language
ResponseExpires	重写HTTP/HTTPS响应中的Expires
ResponseCacheControl	重写HTTP/HTTPS响应中的Cache-Control
ResponseContentDisposition	重写HTTP/HTTPS响应中的Content-Disposition **须知** 当SaveByType为file类型（即文件下载）时，暂不支持重写该响应头，请使用**临时URL**下载文件并重写该响应头。
ResponseContentEncoding	重写HTTP/HTTPS响应中的Content-Encoding

以下代码展示了如何重写响应头：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        ResponseContentType : 'image/jpeg'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取重写后的响应头
                console.log(result.InterfaceResult.ContentType);
            }
        }
});
```


版权所有 © 华为技术有限公司

9.9 获取自定义元数据

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

下载对象成功后会返回对象的自定义元数据。

本示例用于下载桶名为“bucketname”里，名称为“objectname”的对象，同时返回该对象的自定义元数据。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 下载对象，获取对象自定义元数据
obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                console.log('Metadata-->' + JSON.stringify(result.InterfaceResult.Metadata['property']));
            }
        }
});
```

说明

- 要获取对象的自定义元数据，需要在桶的CORS配置中增加允许响应中可返回的附加头域。例如，新增x-obs-meta-property以获取自定义元数据property。

9.10 下载归档存储对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

版权所有 © 华为技术有限公司

如果要下载归档存储对象，需要先将归档存储对象恢复。恢复归档存储对象的恢复选项可支持两类，见下表：

选项	说明	OBS BrowserJS SDK对应值
快速恢复	恢复耗时1~5分钟。	ObsClient.enums.RestoreTierExpedited
标准恢复	恢复耗时3~5小时。默认值。	ObsClient.enums.RestoreTierStandard

⚠ 注意

重复恢复归档存储数据时在延长恢复有效期的同时，也将会对恢复时产生的恢复费用进行重复收取。产生的标准存储类别的对象副本有效期将会延长，并且收取延长时间段产生的标准存储副本费用。

您可以通过ObsClient.restoreObject恢复归档存储对象。以下代码展示了如何下载归档存储对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 恢复归档存储对象
obsClient.restoreObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Days : 1,
        Tier : obsClient.enums.RestoreTierExpedited
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);

            // 等待对象恢复
            setTimeout(function () {

                // 下载对象, 获取对象内容
                obsClient.getObject({
                    Bucket : 'bucketname',
                    Key : 'objectname'
                }, function (err, result) {
                    if(err){
                        console.error('Error-->' + err);
                    }else{
                        console.log('Status-->' + result.CommonMsg.Status);
                        if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                            // 读取对象内容
                            console.log('Object Content:');
```


版权所有 © 华为技术有限公司

```
                        console.log(result.InterfaceResult.Content);
                    }
                }
            });
        }, 6 * 60 * 1000);
    }
});
```

说明

- ObsClient.restoreObject中指定的对象必须是归档存储类型，否则调用该接口会报错。
- 使用**Days参数**指定恢复对象保存的时间，取值范围是1~30；使用**Tier参数**指定恢复选项，表示恢复对象所耗的时间。

9.11 图片处理

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS为用户提供了稳定、安全、高效、易用、低成本的图片处理服务。当要下载的对象是图片文件时，您可以通过传入图片处理参数对图片文件进行图片剪切、图片缩放、图片水印、格式转换等处理。

更多关于图片处理的内容，参见**图片处理特性指南**。

以下代码展示了如何使用下载对象接口实现图片处理：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket: 'bucketname',
        Key: 'objectname.jpg',
        // ImageProcess: 'image/resize,m_fixed,w_100,h_100',        // 强制将图片的宽和高固定为100
        // ImageProcess: 'image/rotate,90',                         // 将图片旋转角度设置为90
        ImageProcess: 'image/resize,m_fixed,w_100,h_100/rotate,90', // 先强制将图片的宽和高固定为100，然后旋转角度设置为90
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

须知

图片处理的更多功能请参考**图片处理功能概述**。

 版权所有 © 华为技术有限公司

说明

- 使用**ImageProcess参数**指定图片处理参数。
- 图片处理参数支持级联处理，可对图片文件依次执行多条命令。

 版权所有 © 华为技术有限公司

10 管理对象

10.1 设置对象元数据

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.setObjectMetadata来设置对象属性，包括CacheControl，ContentDisposition，ContentType，StorageClass，Expires等系统定义的元数据。

本示例用于设置桶名为“bucketname”里，名称为“objectname”的对象Content-Disposition元数据。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
obsClient.setObjectMetadata({
        Bucket : 'bucketname',
        Key : 'objectname',
        Content-Disposition : 'attchment;filename=1.txt',
        MetadataDirective : 'REPLACE_NEW'
}, function (err, result){
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```


版权所有 © 华为技术有限公司

说明

- 设置对象元数据过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

10.2 获取对象属性

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getObjectMetadata来获取对象属性，包括对象长度，对象MIME类型，对象自定义元数据等信息。

本示例用于获取桶名为“bucketname”里，名称为“objectname”的对象元数据。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObjectMetadata({
        Bucket : 'bucketname',
        Key : 'objectname'
}, function (err, result){
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                console.log(result.InterfaceResult.ContentType);
                console.log(result.InterfaceResult.ContentLength);
                console.log(result.InterfaceResult.Metadata['property']);
            }
        }
});
```

说明

- 要获取对象的自定义元数据，需要在桶的CORS配置中增加允许响应中可返回的附加头域。例如，新增x-obs-meta-property以获取自定义元数据property。

版权所有 © 华为技术有限公司

10.3 管理对象访问权限

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

对象访问权限与桶访问权限类似，也可支持预定义访问策略（参见**桶访问权限**）或直接设置。

对象访问权限（**ACL**）可以通过三种方式设置：

1. 上传对象时指定预定义访问策略。
2. 调用ObsClient.setObjectAcl指定预定义访问策略。
3. 调用ObsClient.setObjectAcl直接设置。

上传对象时指定预定义访问策略

以下代码展示如何在上传对象时指定预定义访问策略：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        Body : 'Hello OBS',
        // 设置对象访问权限为公共读
        ACL : obsClient.enums.AclPublicRead
}, function (err, result){
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

为对象设置预定义访问策略

以下代码展示如何为对象设置预定义访问策略：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
```


版权所有 © 华为技术有限公司

```
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setObjectAcl({
        Bucket : 'bucketname',
        Key : 'objectname',
        // 设置对象访问权限为私有读写
        ACL : obsClient.enums.AclPrivate
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

使用**ACL参数**指定对象的访问权限。

直接设置对象访问权限

以下代码展示如何直接设置对象访问权限：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存
放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境
变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setObjectAcl({
        Bucket : 'bucketname',
        Key : 'objectname',
        // 设置对象所有者
        Owner:{'ID':'ownerid'},
        Grants:[
                // 为指定用户设置完全控制权限
                { Grantee : {Type : 'CanonicalUser',ID : 'userid'}, Permission :
obsClient.enums.PermissionFullControl},
                // 为所有用户设置读权限
                { Grantee: {Type : 'Group', URI : obsClient.enums.GroupAllUsers}, Permission :
obsClient.enums.PermissionRead}
        ]
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 使用**Owner参数**指定对象的所有者信息；使用**Grants参数**指定被授权的用户信息。
- ACL中需要填写的所有者（Owner）或者被授权用户（Grantee）的ID，是指用户的账户ID，可通过OBS控制台“我的凭证”页面查看。
- 当前OBS对象支持的可被授权的用户组为：
  - 所有用户：ObsClient.enums.GroupAllUsers

获取对象访问权限

您可以通过ObsClient.getObjectAcl获取对象的访问权限。以下代码展示如何获取对象访问权限：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

10.4 列举对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.listObjects列举出桶里的对象。

该接口可设置的参数如下：

版权所有 © 华为技术有限公司

参数	作用
Prefix	限定返回的对象名必须带有Prefix前缀。
Marker	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	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	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。 对于并行文件系统，不携带此参数时默认列举是递归列举此目录下所有内容，会列举子目录。在大数据场景下（目录层级深、目录下文件多）的列举，建议设置[delimiter='/']，只列举当前目录下的内容，不列举子目录，提高列举效率。

简单列举

以下代码展示如何简单列举对象，最多返回1000个对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listObjects({
        Bucket : 'bucketname'
}, function (err, result) {
      if(err){
        console.error('Error-->' + err);
      }else{
        console.log('Status-->' + result.CommonMsg.Status);
              if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                    for(var j in result.InterfaceResult.Contents){
                            console.log('Contents[' + j +  ']:');
                            console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                            console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']['ID']);
                     }
              }
      }
});
```

说明

- 每次至多返回1000个对象，如果指定桶包含的对象数量大于1000，则返回结果中InterfaceResult.IsTruncated为true表明本次没有返回全部对象，并可通过InterfaceResult.NextMarker获取下次列举的起始位置。
- 如果想获取指定桶包含的所有对象，可以采用分页列举的方式。

 版权所有 © 华为技术有限公司

指定数目列举

以下代码展示如何指定数目列举对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listObjects({
        Bucket : 'bucketname',
        // 只列举100个对象
        MaxKeys : 100
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
                if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                    for(var j in result.InterfaceResult.Contents){
                        console.log('Contents[' + j +  ']:');
                        console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                        console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']['ID']);
                    }
                }
        }
});
```

指定前缀列举

以下代码展示如何指定前缀列举对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listObjects({
        Bucket : 'bucketname',
        // 设置列举带有prefix前缀的100个对象
        MaxKeys : 100,
        Prefix : 'prefix'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
                if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                    for(var j in result.InterfaceResult.Contents){
                        console.log('Contents[' + j +  ']:');
```

```
                        console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                        console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']
['ID']);
                }
            }
    }
});
```

指定起始位置列举

以下代码展示如何指定起始位置列举对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listObjects({
        Bucket : 'bucketname',
        // 设置列举对象名字典序在"test"之后的100个对象
        MaxKeys : 100,
        Marker : 'test'
}, function (err, result) {
        if(err){
            console.error('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
                if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                    for(var j in result.InterfaceResult.Contents){
                        console.log('Contents[' + j +  ']:');
                        console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                        console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']['ID']);
                    }
                }
        }
});
```

分页列举全部对象

以下代码展示分页列举全部对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var listAll = function (marker) {
        obsClient.listObjects({
            Bucket : 'bucketname',
            // 设置每页100个对象
```

```
            MaxKeys : 100,
            Marker : marker
        }, function (err, result) {
            if(err){
                console.error('Error-->' + err);
            }else{
                console.log('Status-->' + result.CommonMsg.Status);
                if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                    for(var j in result.InterfaceResult.Contents){
                        console.log('Contents[' + j +  ']:');
                        console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                        console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']
['ID']);
                    }
                    if(result.InterfaceResult.IsTruncated === 'true'){
                        listAll(result.InterfaceResult.NextMarker);
                    }
                }
            }
        });
};

listAll();
```

列举文件夹中的所有对象

OBS本身是没有文件夹的概念的，桶中存储的元素只有对象。文件夹对象实际上是一个大小为0且对象名以“/”结尾的对象，将这个文件夹对象名作为前缀，即可模拟列举文件夹中对象的功能。以下代码展示如何列举文件夹中的对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var listAll = function (marker) {
      obsClient.listObjects({
            Bucket : 'bucketname',
            MaxKeys : 1000,
            // 设置文件夹对象名"dir/"为前缀
            Prefix : 'dir/'
        }, function (err, result) {
            if(err){
                console.error('Error-->' + err);
            }else{
                console.log('Status-->' + result.CommonMsg.Status);
                if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                    for(var j in result.InterfaceResult.Contents){
                        console.log('Contents[' + j +  ']:');
                        console.log('Key-->' + result.InterfaceResult.Contents[j]['Key']);
                        console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']
['ID']);
                    }
                    if(result.InterfaceResult.IsTruncated === 'true'){
                        listAll(result.InterfaceResult.NextMarker);
                    }
                }
            }
        });
```

 版权所有 © 华为技术有限公司

```
};

listAll();
```

按文件夹分组列举所有对象

以下代码展示如何按文件夹分组，列举桶内所有对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listObjects({
        Bucket: 'bucketname',
        // 设置文件夹分隔符"/"
        Delimiter: '/'
}, function (err, result) {
        if(!err && result.CommonMsg.Status < 300){
            console.log('Objects in the root directory:');
            for(var j in result.InterfaceResult.Contents){
                console.log('\tKey-->' + result.InterfaceResult.Contents[j]['Key']);
                console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']['ID']);
            }
            var listObjectsByPrefix = function (commonPrefixes){
                for(var i in commonPrefixes){
                    var prefix = commonPrefixes[i]['Prefix'];
                    obsClient.listObjects({
                        Bucket: 'bucketname',
                        Delimiter: '/',
                        Prefix: prefix
                    }, function (err, result) {
                        if(!err && result.CommonMsg.Status < 300){
                            console.log('Objects in folder:');
                            for(var j in result.InterfaceResult.Contents){
                                console.log('\tKey-->' + result.InterfaceResult.Contents[j]['Key']);
                                console.log('Owner[ID]-->' + result.InterfaceResult.Contents[j]['Owner']['ID']);
                            }
                            console.log('\n');
                            if(result.InterfaceResult.CommonPrefixes && result.InterfaceResult.CommonPrefixes.length > 0){
                                listObjectsByPrefix(result.InterfaceResult.CommonPrefixes);
                            }
                        }
                    });
                }
            };
            listObjectsByPrefix(result.InterfaceResult.CommonPrefixes);
        }
});
```

 版权所有 © 华为技术有限公司

说明

- 以上代码示例没有考虑文件夹中对象数超过1000个的情况。
- 由于是需要列举出文件夹中的对象和子文件夹，且文件夹对象总是以“/”结尾，因此Delimiter总是为“/”。
- 每次递归的返回结果中InterfaceResult.Contents包含的是文件夹中的对象；InterfaceResult.CommonPrefixes包含的是文件夹的子文件夹。

10.5 删除对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

删除单个对象

您可以通过ObsClient.deleteObject删除单个对象。以下代码展示如何删除单个对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.deleteObject({
        Bucket: 'bucketname',
        Key : 'objectname'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

批量删除对象

您可以通过ObsClient.deleteObjects批量删除对象。

每次最多删除1000个对象，并支持两种响应模式：详细（verbose）模式和简单（quiet）模式。

- 详细模式：返回的删除成功和删除失败的所有结果，默认模式。
- 简单模式：只返回的删除过程中出错的结果。

以下代码展示了如何进行批量删除对象：

 版权所有 © 华为技术有限公司

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.deleteObjects({
        Bucket: 'bucketname',
        // 设置为verbose模式
        Quiet : false,
        Objects : [{Key:'objectname1'},{Key:'objectname2'}, {Key : 'objectname3'}]
}, function (err, result) {
        if(err){
                console.log('Error-->' + err);
        }else{
                console.log('Status-->' + result.CommonMsg.Status);
                if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                  // 获取删除成功的对象
                  console.log('Deleteds:');
                  for(var i in result.InterfaceResult.Deleteds){
                        console.log('Deleted[' + i + ']:');
                        console.log('Key-->'+result.InterfaceResult.Deleteds[i]['Key']);
                        console.log('VersionId-->' + result.InterfaceResult.Deleteds[i]['VersionId']);
                  }
                  // 获取删除失败的对象
                  console.log('Errors:');
                  for(var i in result.InterfaceResult.Errors){
                        console.log('Error[' + i + ']:');
                        console.log('Key-->' + result.InterfaceResult.Errors[i]['Key']);
                        console.log('VersionId-->' + result.InterfaceResult.Errors[i]['VersionId']);
                  }
                }
        }
});
```

说明

使用**Quiet参数**指定响应模式；使用**Objects参数**指定待删除的对象列表。

10.6 复制对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

复制对象特性用来为OBS上已经存在的对象创建一个副本。

您可以通过ObsClient.copyObject来复制对象。复制对象时，可重新指定新对象的属性和设置对象权限，且支持条件复制。

版权所有 © 华为技术有限公司

说明

- 如果待复制的源对象是归档存储类型，则必须先恢复源对象才能进行复制。

简单复制

以下代码展示了如何进行简单复制：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.copyObject({
        Bucket: 'destbucketname',
        Key : 'destobjectname',
        CopySource:'sourcebucketname/sourceobjectname'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

使用**CopySource参数**指定复制时的源对象信息。

重写对象属性

以下代码展示了如何在复制对象时重写对象属性：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.copyObject({
    Bucket: 'destbucketname',
    Key : 'destobjectname',
    CopySource:'sourcebucketname/sourceobjectname',
    ContentType : 'image/jpeg',
    StorageClass : obsClient.enums.StorageClassWarm,
    Metadata:{'property':'property-value'},
    MetadataDirective : ObsClient.enums.ReplaceMetadata
}, function (err, result) {
        if(err){
```

```
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

使用**Metadata参数**指定待重写的自定义对象属性；使用**MetadataDirective参数**指定重写选项，支持ObsClient.enums.CopyMetadata（从源对象复制）和ObsClient.enums.ReplaceMetadata（重写）两个值。

限定条件复制

复制对象时，可以指定一个或多个限定条件，满足限定条件时则进行复制，否则返回异常码，复制对象失败。

您可以使用的限定条件如下：

参数	作用	格式
CopySourceIfModifiedSince	如果源对象在指定的时间后有修改，则进行复制，否则抛出异常。	符合http://www.ietf.org/rfc/rfc2616.txt规定格式的HTTP时间字符串。
CopySourceIfUnmodifiedSince	如果源对象在指定的时间后没有修改，则进行复制，否则抛出异常。	符合http://www.ietf.org/rfc/rfc2616.txt规定格式的HTTP时间字符串。
CopySourceIfMatch	如果源对象的ETag值与该参数值相同，则进行复制，否则返回异常码。	字符串。
CopySourceIfNoneMatch	如果源对象的ETag值与该参数值不相同，则进行复制，否则返回异常码。	字符串。

说明

- 源对象的ETag值是指源对象数据的MD5校验值。
- 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，则复制失败，返回异常码：412 precondition failed。
- CopySourceIfModifiedSince和CopySourceIfNoneMatch可以一起使用；CopySourceIfUnmodifiedSince和CopySourceIfMatch可以一起使用。

以下代码展示了如何进行限定条件复制：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
```

```
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.copyObject({
    Bucket: 'destbucketname',
    Key : 'destobjectname',
    CopySource:'sourcebucketname/sourceobjectname',
    CopySourceIfModifiedSince : 'Thu, 31 Dec 2015 16:00:00 GMT',
    CopySourceIfNoneMatch : 'none-match-etag'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

重写对象访问权限

以下代码展示了如何在复制对象时重写对象访问权限：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.copyObject({
    Bucket: 'destbucketname',
    Key : 'destobjectname',
    CopySource:'sourcebucketname/sourceobjectname',
    // 复制时重写对象访问权限为公共读
    ACL : obsClient.enums.AclPublicRead
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

使用**ACL参数**重写对象访问权限。

 版权所有 © 华为技术有限公司

11 临时授权访问

11.1 使用临时 URL 进行授权访问

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS客户端支持通过访问密钥、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，可将该URL提供给其他用户进行临时访问。在生成URL时，您需要指定URL的有效期来限制访客用户的访问时长。

如果您想授予其他用户对桶或对象临时进行其他操作的权限（例如上传或下载对象），则需要生成带对应请求的URL后（例如使用生成PUT请求的URL上传对象），将该URL提供给其他用户。

通过该方式可支持的操作以及相关信息见下表：

操作名	HTTP请求方法	特殊操作符（子资源）	是否需要桶名	是否需要对象名
列举桶内对象	GET	N/A	是	否
列举桶内多版本对象	GET	versions	是	否
列举分段上传任务	GET	uploads	是	否
获取桶元数据	HEAD	N/A	是	否

版权所有 © 华为技术有限公司

操作名	HTTP请求方法	特殊操作符（子资源）	是否需要桶名	是否需要对象名
获取桶区域位置	GET	location	是	否
获取桶存量信息	GET	storageinfo	是	否
设置桶配额	PUT	quota	是	否
获取桶配额	GET	quota	是	否
设置桶存储类型	PUT	storagePolicy	是	否
获取桶存储类型	GET	storagePolicy	是	否
设置桶访问权限	PUT	acl	是	否
获取桶访问权限	GET	acl	是	否
开启/关闭桶日志	PUT	logging	是	否
查看桶日志	GET	logging	是	否
设置桶策略	PUT	policy	是	否
查看桶策略	GET	policy	是	否
删除桶策略	DELETE	policy	是	否
设置生命周期规则	PUT	lifecycle	是	否
查看生命周期规则	GET	lifecycle	是	否

版权所有 © 华为技术有限公司

操作名	HTTP请求方法	特殊操作符（子资源）	是否需要桶名	是否需要对象名
删除生命周期规则	DELETE	lifecycle	是	否
设置托管配置	PUT	website	是	否
查看托管配置	GET	website	是	否
清除托管配置	DELETE	website	是	否
设置桶多版本状态	PUT	versioning	是	否
查看桶多版本状态	GET	versioning	是	否
查看跨域规则	GET	cors	是	否
删除跨域规则	DELETE	cors	是	否
设置桶标签	PUT	tagging	是	否
查看桶标签	GET	tagging	是	否
删除桶标签	DELETE	tagging	是	否
上传对象	PUT	N/A	是	是
追加上传	POST	append	是	是
下载对象	GET	N/A	是	是
复制对象	PUT	N/A	是	是
删除对象	DELETE	N/A	是	是
批量删除对象	POST	delete	是	是

版权所有 © 华为技术有限公司

操作名	HTTP请求方法	特殊操作符（子资源）	是否需要桶名	是否需要对象名
获取对象属性	HEAD	N/A	是	是
设置对象访问权限	PUT	acl	是	是
查看对象访问权限	GET	acl	是	是
初始化分段上传任务	POST	uploads	是	是
上传段	PUT	N/A	是	是
复制段	PUT	N/A	是	是
列举已上传的段	GET	N/A	是	是
合并段	POST	N/A	是	是
取消分段上传任务	DELETE	N/A	是	是
恢复归档存储对象	POST	restore	是	是

通过OBS BrowserJS SDK生成临时URL访问OBS的步骤如下：

步骤1 通过ObsClient.createSignedUrlSync生成带签名信息的URL。

步骤2 使用任意HTTP库发送HTTP/HTTPS请求，访问OBS服务。

----结束

⚠ 注意

如果遇到跨域报错、签名不匹配问题，请参考以下步骤排查问题：

1. 未配置跨域，需要在控制台配置CORS规则，请参考**配置桶允许跨域请求**。
2. 签名计算问题，请参考**URL中携带签名**排查签名参数是否正确；比如上传对象功能，后端将Content-Type参与计算签名生成授权URL，但是前端使用授权URL时没有设置Content-Type字段或者传入错误的值，此时会出现跨域错误。解决方案为：Content-Type字段前后端保持一致。

版权所有 © 华为技术有限公司

以下代码展示了如何使用临时URL进行授权访问，包括：上传对象、下载对象、列举对象、删除对象。

上传对象

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
// 使用PUT请求上传对象
var bucketName = 'bucketname';
var objectKey = 'objectname';
var method = 'PUT';
var headers = {
        'Content-Type' : 'text/plain'
}
var res = obsClient.createSignedUrlSync({
        Method : method,
        Bucket : bucketName,
        Key : objectKey,
        Expires : 3600,
        Headers : headers
});
var content = 'Hello OBS';

var reopt = {
        method : method,
        url : res.SignedUrl,
        withCredentials: false,
        headers : res.ActualSignedRequestHeaders || {},
        validateStatus: function(status){
                return status >= 200;
        },
        maxRedirects : 0,
        responseType : 'text',
        data : content,
};

axios.request(reopt).then(function (response) {
        if(response.status < 300){
                console.log('Creating object using temporary signature succeed.');
        }else{
                console.log('Creating object using temporary signature failed!');
                console.log('status:' + response.status);
                console.log('\n');
        }
        console.log(response.data);
        console.log('\n');
}).catch(function (err) {
        console.log('Creating object using temporary signature failed!');
        console.log(err);
        console.log('\n');
});
```

下载对象

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存
```

 版权所有 © 华为技术有限公司

```
放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用GET请求下载对象
var bucketName = 'bucketname';
var objectKey = 'objectname';
var method = 'GET';

var res = obsClient.createSignedUrlSync({
        Method : method,
        Bucket : bucketName,
        Key : objectKey,
        Expires : 3600,
});

var reopt = {
        method : method,
        url : res.SignedUrl,
        withCredentials: false,
        headers : res.ActualSignedRequestHeaders || {},
        validateStatus: function(status){
            return status >= 200;
        },
        maxRedirects : 0,
        responseType : 'text',
};

axios.request(reopt).then(function (response) {
        if(response.status < 300){
            console.log('Getting object using temporary signature succeed.');
        }else{
            console.log('Getting object using temporary signature failed!');
            console.log('status:' + response.status);
            console.log('\n');
        }
        console.log(response.data);
        console.log('\n');
}).catch(function (err) {
        console.log('Getting object using temporary signature failed!');
        console.log(err);
        console.log('\n');
});
```

列举对象

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用GET请求获取对象列表
var bucketName = 'bucketname';
```

```
var method = 'GET';

var res = obsClient.createSignedUrlSync({
    Method : method,
    Bucket : bucketName,
    Expires : 3600,
});

var reopt = {
    method : method,
    url : res.SignedUrl,
    withCredentials: false,
    headers : res.ActualSignedRequestHeaders || {},
    validateStatus: function(status){
        return status >= 200;
    },
    maxRedirects : 0,
    responseType : 'text',
};

axios.request(reopt).then(function (response) {
    if(response.status < 300){
        console.log('Listing object using temporary signature succeed.');
    }else{
        console.log('Listing object using temporary signature failed!');
        console.log('status:' + response.status);
        console.log('\n');
    }
    console.log(response.data);
    console.log('\n');
}).catch(function (err) {
    console.log('Listing object using temporary signature failed!');
    console.log(err);
    console.log('\n');
});
```

删除对象

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用DELETE请求删除对象
var bucketName = 'bucketname';
var objectKey = 'objectname';
var method = 'DELETE';

var res = obsClient.createSignedUrlSync({
    Method : method,
    Bucket : bucketName,
    Key : objectKey,
    Expires : 3600,
});

var reopt = {
    method : method,
    url : res.SignedUrl,
    withCredentials: false,
    headers : res.ActualSignedRequestHeaders || {},
```

```
    validateStatus: function(status){
        return status >= 200;
    },
    maxRedirects : 0,
    responseType : 'text',
};

axios.request(reopt).then(function (response) {
    if(response.status < 300){
        console.log('Deleting object using temporary signature succeed.');
    }else{
        console.log('Deleting object using temporary signature failed!');
        console.log('status:' + response.status);
        console.log('\n');
    }
    console.log(response.data);
    console.log('\n');
}).catch(function (err) {
    console.log('Deleting object using temporary signature failed!');
    console.log(err);
    console.log('\n');
});
```

初始化分段上传任务

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用POST请求初始化分段上传任务
var bucketName = 'bucketname';
var objectKey = 'objectname';
var method = 'POST';

var res = obsClient.createSignedUrlSync({
    Method : method,
    Bucket : bucketName,
    Key : objectKey,
    Expires : 3600,
    SpecialParam: "uploads"
});

var reopt = {
    method : method,
    url : res.SignedUrl,
    withCredentials: false,
    headers : res.ActualSignedRequestHeaders || {},
    validateStatus: function(status){
        return status >= 200;
    },
    maxRedirects : 0,
    responseType : 'text',
};

axios.request(reopt).then(function (response) {
    if(response.status < 300){
        console.log('Initiate multipart upload using temporary signature succeed.');
    }else{
        console.log('Initiate multipart upload using temporary signature failed!');
```

 版权所有 © 华为技术有限公司

```
            console.log('status:' + response.status);
            console.log('\n');
        }
        console.log(response.data);
        console.log('\n');
}).catch(function (err) {
        console.log('Initiate multipart upload using temporary signature failed!');
        console.log(err);
        console.log('\n');
});
```

上传段

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用PUT请求上传段
var bucketName = 'bucketname';
var objectKey = 'objectname';
var method = 'PUT';
var headers = {
        'Content-Type' : 'text/plain'
}

var res = obsClient.createSignedUrlSync({
        Method : method,
        Bucket : bucketName,
        Key : objectKey,
        Expires : 3600,
        Headers: headers
        QueryParams: {
            // 指定上传段的段号
            'partNumber': '1',
            // 分段上传任务的ID。任务ID可以通过初始化分段上传任务生成。
            'uploadId': '000001648453845DBB78F2340DD4*****',
        }
});

var content = 'Hello OBS';

var reopt = {
        method : method,
        url : res.SignedUrl,
        withCredentials: false,
        headers : res.ActualSignedRequestHeaders || {},
        validateStatus: function(status){
            return status >= 200;
        },
        maxRedirects : 0,
        responseType : 'text',
        data : content,
};

axios.request(reopt).then(function (response) {
        if(response.status < 300){
            console.log('Upload part using temporary signature succeed.');
        }else{
            console.log('Upload part using temporary signature failed!');
```

 版权所有 © 华为技术有限公司

```
            console.log('status:' + response.status);
            console.log('\n');
        }
        console.log(response.data);
        console.log('\n');
}).catch(function (err) {
        console.log('Upload part upload using temporary signature failed!');
        console.log(err);
        console.log('\n');
});
```

合并段

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 使用POST请求合并段
var bucketName = 'bucketname';
var objectKey = 'objectname';
var method = 'POST';
var headers = {
        'Content-Type' : 'application/xml'
}

var res = obsClient.createSignedUrlSync({
        Method : method,
        Bucket : bucketName,
        Key : objectKey,
        Expires : 3600,
        Headers: headers,
        QueryParams: {
            // 分段上传任务的ID。任务ID可以通过初始化分段上传任务生成。
            'uploadId': '000001648453845DBB78F2340DD4*****',
        }
});

var content = "<CompleteMultipartUpload>";
content += "<Part>";
content += "<PartNumber>1</PartNumber>";
content += "<ETag>da6a0d097e307ac52ed9b4ad551801fc</ETag>";
content += "</Part>";
content += "<Part>";
content += "<PartNumber>2</PartNumber>";
content += "<ETag>da6a0d097e307ac52ed9b4ad551801fc</ETag>";
content += "</Part>";
content += "</CompleteMultipartUpload>";

var reopt = {
        method : method,
        url : res.SignedUrl,
        withCredentials: false,
        headers : res.ActualSignedRequestHeaders || {},
        validateStatus: function(status){
            return status >= 200;
        },
        maxRedirects : 0,
        responseType : 'text',
        data : content,
```

 版权所有 © 华为技术有限公司

```
};

axios.request(reopt).then(function (response) {
    if(response.status < 300){
        console.log('Complete multipart upload using temporary signature succeed.');
    }else{
        console.log('Complete multipart upload using temporary signature failed!');
        console.log('status:' + response.status);
        console.log('\n');
    }
    console.log(response.data);
    console.log('\n');
}).catch(function (err) {
    console.log('Complete multipart upload using temporary signature failed!');
    console.log(err);
    console.log('\n');
});
```

说明

1. 使用Method参数指定HTTP请求方法类型；使用Expires参数指定生成的URL有效期；使用Headers参数指定请求的头信息；使用SpecialParam参数指定特殊操作符；使用QueryParams参数指定请求的查询参数。
2. 后端生成临时URL返回的ActualSignedRequestHeaders可能会携带Host，如果使用浏览器上传对象，则不需要配置Host，即在上传时需要去掉Headers中的Host，否则会报错。

 版权所有 © 华为技术有限公司

12 多版本控制

12.1 多版本控制简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS支持保存一个对象的多个版本，您可以利用多版本控制，在一个桶中保留多个版本的对象。

多版本功能可以方便地检索和还原各个版本，在意外操作或应用程序故障时快速恢复数据。

在默认情况下，OBS中新创建的桶不会开启多版本功能，向同一个桶上传同名的对象时，新上传的对象将覆盖原有的对象。

更多关于多版本控制的内容请参见**多版本控制**。

12.2 设置桶多版本状态

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.setBucketVersioning设置桶的多版本状态。OBS中的桶支持两种多版本状态：

版权所有 © 华为技术有限公司

多版本状态	说明	OBS服务端对应值
启用状态	1. 上传对象时，系统为每一个对象创建一个唯一版本号，上传同名的对象将不再覆盖旧的对象，而是创建新的不同版本号的同名对象。 2. 可以指定版本号下载对象，不指定版本号默认下载最新对象。 3. 删除对象时可以指定版本号删除，不带版本号删除对象仅产生一个带唯一版本号的删除标记，并不删除对象。 4. 列出桶内对象列表（ObsClient.listObjects）时默认列出最新对象列表，可以指定列出桶内所有版本对象列表（ObsClient.listVersions）。 5. 除了删除标记外，每个版本的对象存储均需计费。	Enabled
暂停状态	1. 旧的版本数据继续保留。 2. 上传对象时创建对象的版本号为null，上传同名的对象将覆盖原有同名的版本号为null的对象。 3. 可以指定版本号下载对象，不指定版本号默认下载最新对象。 4. 删除对象时可以指定版本号删除，不带版本号删除对象将产生一个版本号为null的删除标记，并删除版本号为null的对象。 5. 除了删除标记外，每个版本的对象存储均需计费。	Suspended

以下代码展示了如何设置桶的多版本状态：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 启用桶多版本状态
obsClient.setBucketVersioning({
        Bucket : 'bucketname',
        VersionStatus : 'Enabled'
}, function (err, result) {
        if(err){
                console.log('Error-->' + err);
        }else{
                console.log('Status-->' + result.CommonMsg.Status);
        }
});

// 暂停桶多版本状态
obsClient.setBucketVersioning({
```

 版权所有 © 华为技术有限公司

```
    Bucket : 'bucketname',
    VersionStatus : 'Suspended'
}, function (err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

使用**VersionStatus参数**指定桶的多版本状态。

12.3 查看桶多版本状态

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketVersioning查看桶的多版本状态。

本示例用于获取桶名为“bucketname”里的多版本状态。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 启用桶多版本
obsClient.getBucketVersioning({
    Bucket : 'bucketname'
}, function (err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
        if(result.CommonMsg.Status < 300 && result.InterfaceResult){
            console.log('VersionStatus-->' + result.InterfaceResult.VersionStatus);
        }
    }
});
```

说明

- 查看桶多版本状态过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

 版权所有 © 华为技术有限公司

12.4 获取多版本对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getObject接口指定**VersionId参数**来获取多版本对象，示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 设置versionId获取多版本对象
obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        VersionId : 'versionid'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                console.log('Content-->' + result.InterfaceResult.Content);
            }
        }
});
```

说明

versionId参数为对象的版本号，可通过**指定前缀列举多版本对象**获取。

如果版本号为空则默认下载最新版本的对象。

12.5 复制多版本对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.copyObject接口在**CopySource参数**中指定待复制对象的versionId来复制多版本对象。

版权所有 © 华为技术有限公司

本示例用于通过设置versionId复制sourcebucket桶中sourceobjectname的多版本对象，复制到destbucketname桶中destobjectname。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.copyObject({
        Bucket : 'destbucketname',
        Key : 'destobjectname',
        // 设置要复制对象的版本号
        CopySource : 'sourcebucket/sourceobjectname?versionId=versionid'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 复制多版本对象过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

12.6 恢复多版本归档存储对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.restoreObject接口指定**VersionId参数**来恢复多版本归档存储对象。

本示例用于通过设置versionId值将destbucketname桶中destobjectname的多版本归档存储对象，快速恢复为标准存储对象。

代码示例如下所示：

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

 版权所有 © 华为技术有限公司

```
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.restoreObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        VersionId : 'versionid',
        Days : 1,
        // 使用快速恢复方式，恢复多版本对象
        Tier : obsClient.enums.RestoreTierExpedited
}, function (err, result) {
        if(err){
                console.log('Error-->' + err);
        }else{
                console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

⚠ 注意

重复恢复归档存储数据时在延长恢复有效期的同时，也将会对恢复时产生的恢复费用进行重复收取。产生的标准存储类别的对象副本有效期将会延长，并且收取延长时间段产生的标准存储副本费用。

12.7 列举多版本对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.listVersions列举多版本对象。

该接口可设置的参数如下：

参数	作用
Prefix	限定返回的对象名必须带有Prefix前缀。
KeyMarker	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	列举多版本对象的最大数目，取值范围为1~1000，当超出范围时，按照默认的1000进行处理。
Delimiter	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	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。

版权所有 © 华为技术有限公司

说明

- 如果VersionIdMarker不是KeyMarker的一个版本号，则该参数无效。
- ObsClient.listVersions返回结果包含多版本对象和对象删除标记。

简单列举

以下代码展示如何简单列举多版本对象，最多返回1000个对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listVersions({
        Bucket : 'bucketname'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取多版本对象
                for(var j in result.InterfaceResult.Versions){
                    console.log('Version[' + j +  ']:');
                    console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                }
                // 获取对象删除标记
                for(var i in result.InterfaceResult.DeleteMarkers){
                    console.log('DeleteMarker[' + i +  ']:');
                    console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
                }
            }
        }
});
```

说明

- 每次至多返回1000个多版本对象，如果指定桶包含的对象数量大于1000，则返回结果中InterfaceResult.IsTruncated为true表明本次没有返回全部对象，并可通过InterfaceResult.NextKeyMarker和InterfaceResult.NextVersionIdMarker获取下次列举的起始位置。
- 如果想获取指定桶包含的所有多版本对象，可以采用分页列举的方式。

指定数目列举

以下代码展示如何指定数目列举多版本对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境
```

```
变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listVersions({
        Bucket : 'bucketname',
        MaxKeys : 100
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取多版本对象
                for(var j in result.InterfaceResult.Versions){
                    console.log('Version[' + j +  ']:');
                    console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                }
                // 获取对象删除标记
                for(var i in result.InterfaceResult.DeleteMarkers){
                    console.log('DeleteMarker[' + i +  ']:');
                    console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
                }
            }
        }
});
```

指定前缀列举

以下代码展示如何指定前缀列举多版本对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存
放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境
变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 设置列举带有prefix前缀的100个多版本对象
obsClient.listVersions({
        Bucket : 'bucketname',
        MaxKeys : 100,
        Prefix : 'prefix'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取多版本对象
                for(var j in result.InterfaceResult.Versions){
                    console.log('Version[' + j +  ']:');
```

```
                console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
            }
            // 获取对象删除标记
            for(var i in result.InterfaceResult.DeleteMarkers){
                console.log('DeleteMarker[' + i +  ']:');
                console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
             }
        }
    }
});
```

指定起始位置列举

以下代码展示如何指定起始位置列举多版本对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 设置列举对象名字典序在"test"之后的100个多版本对象
obsClient.listVersions({
        Bucket : 'bucketname',
        MaxKeys : 100,
        KeyMarker : 'test'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取多版本对象
                for(var j in result.InterfaceResult.Versions){
                    console.log('Version[' + j +  ']:');
                    console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                }
                // 获取对象删除标记
                for(var i in result.InterfaceResult.DeleteMarkers){
                    console.log('DeleteMarker[' + i +  ']:');
                    console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
                }
            }
        }
});
```

分页列举全部多版本对象

以下代码展示分页列举全部多版本对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
```

```
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var listAll = function (keyMarker, versionIdMarker) {
      obsClient.listVersions({
            Bucket : 'bucketname',
            MaxKeys : 100,
            KeyMarker : keyMarker,
            VersionIdMarker : versionIdMarker
      }, function (err, result) {
            if(err){
                  console.log('Error-->' + err);
            }else{
                  console.log('Status-->' + result.CommonMsg.Status);
                  if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                        // 获取多版本对象
                        for(var j in result.InterfaceResult.Versions){
                              console.log('Version[' + j +  ']:');
                              console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                              console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                              console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                         }
                        // 获取对象删除标记
                        for(var i in result.InterfaceResult.DeleteMarkers){
                              console.log('DeleteMarker[' + i +  ']:');
                              console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                              console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                              console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
                        }
                        if(result.InterfaceResult.IsTruncated === 'true'){
                              listAll(result.InterfaceResult.NextKeyMarker, result.InterfaceResult.NextVersionIdMarker);
                        }
                  }
            }
      });
};

listAll();
```

列举文件夹中的所有多版本对象

OBS本身是没有文件夹的概念的，桶中存储的元素只有对象。文件夹对象实际上是一个大小为0且对象名以“/”结尾的对象，将这个文件夹对象名作为前缀，即可模拟列举文件夹中对象的功能。以下代码展示如何列举文件夹中的多版本对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
```

```
var listAll = function (keyMarker, versionIdMarker) {
    obsClient.listVersions({
        Bucket : 'bucketname',
        MaxKeys : 100,
        KeyMarker : keyMarker,
        VersionIdMarker : versionIdMarker,
        // 设置文件夹对象名"dir/"为前缀
        Prefix : 'dir/'
    }, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                // 获取多版本对象
                for(var j in result.InterfaceResult.Versions){
                    console.log('Version[' + j +  ']:');
                    console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                }
                // 获取对象删除标记
                for(var i in result.InterfaceResult.DeleteMarkers){
                    console.log('DeleteMarker[' + i +  ']:');
                    console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                    console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
                }
                if(result.InterfaceResult.IsTruncated === 'true'){
                    listAll(result.InterfaceResult.NextKeyMarker,
result.InterfaceResult.NextVersionIdMarker);
                }
            }
        }
    });
};

listAll();
```

按文件夹分组列举所有多版本对象

以下代码展示如何按文件夹分组，列举桶内所有多版本对象：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.listVersions({
    Bucket : 'bucketname',
    // 设置文件夹分隔符"/"
    Delimiter : '/'
}, function (err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        console.log('Objects in the root directory:');
        if(result.CommonMsg.Status < 300 && result.InterfaceResult){
            // 获取多版本对象
```

```
            for(var j in result.InterfaceResult.Versions){
                console.log('Version[' + j +  ']:');
                console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
            }
            // 获取对象删除标记
            for(var i in result.InterfaceResult.DeleteMarkers){
                console.log('DeleteMarker[' + i +  ']:');
                console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                console.log('VersionId-->'+ result.InterfaceResult.DeleteMarkers[i]['VersionId']);
                console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']['ID']);
            }
        }

        var listVersionsByPrefix = function (commonPrefixes) {
            for(var n in commonPrefixes){
                obsClient.listVersions({
                    Bucket : 'bucketname',
                    Delimiter : '/',
                    Prefix: commonPrefixes[n]['Prefix']
                }, function (err, result) {
                    console.log('Objects in folder:');
                    if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                      // 获取多版本对象
                      for(var j in result.InterfaceResult.Versions){
                          console.log('Version[' + j +  ']:');
                          console.log('Key-->' + result.InterfaceResult.Versions[j]['Key']);
                          console.log('VersionId-->' + result.InterfaceResult.Versions[j]['VersionId']);
                          console.log('Owner[ID]-->' + result.InterfaceResult.Versions[j]['Owner']['ID']);
                      }
                      // 获取对象删除标记
                      for(var i in result.InterfaceResult.DeleteMarkers){
                          console.log('DeleteMarker[' + i +  ']:');
                          console.log('Key-->' + result.InterfaceResult.DeleteMarkers[i]['Key']);
                          console.log('VersionId-->' + result.InterfaceResult.DeleteMarkers[i]
['VersionId']);
                          console.log('Owner[ID]-->' + result.InterfaceResult.DeleteMarkers[i]['Owner']
['ID']);
                      }
                      console.log('\n');
                      if(result.InterfaceResult.CommonPrefixes &&
result.InterfaceResult.CommonPrefixes.length > 0){
                          listVersionsByPrefix(result.InterfaceResult.CommonPrefixes);
                      }
                    }
                });
            }
        };
        listVersionsByPrefix(result.InterfaceResult.CommonPrefixes);
    }
});
```

说明

- 以上代码示例没有考虑文件夹中多版本对象数超过1000个的情况。
- 由于是需要列举出文件夹中的对象和子文件夹，且文件夹对象总是以“/”结尾，因此Delimiter总是为“/”。
- 每次递归的返回结果中InterfaceResult.Versions包含的是文件夹中的对象；InterfaceResult.DeleteMarkers包含的是文件夹中的删除标记；InterfaceResult.CommonPrefixes包含的是文件夹的子文件夹。

 版权所有 © 华为技术有限公司

12.8 多版本对象权限

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

设置多版本对象访问权限

您可以通过ObsClient.setObjectAcl接口指定**VersionId参数**设置多版本对象的访问权限，示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setObjectAcl({
        Bucket : 'bucketname',
        Key : 'objectname',
        VersionId : 'versionid',
        // 通过预定义访问策略设置多版本对象访问权限为公共读
        ACL : obsClient.enums.AclPublicRead
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});

obsClient.setObjectAcl({
        Bucket : 'bucketname',
        Key : 'objectname',
        VersionId : 'versionid',
        // 设置对象所有者
        Owner:{'ID':'ownerid'},
        Grants:[
            // 为所有用户设置读权限和写ACP权限
            { Grantee:{Type : 'Group', URI : obsClient.enums.GroupAllUsers}, Permission : obsClient.enums.PermissionRead},
            { Grantee:{Type : 'Group', URI : obsClient.enums.GroupAllUsers}, Permission : obsClient.enums.PermissionWriteAcp}
        ]
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```


版权所有 © 华为技术有限公司

说明

- 使用**Owner参数**指定对象的所有者信息；使用**Grants参数**指定被授权的用户信息。
- ACL中需要填写的所有者（Owner）或者被授权用户（Grantee）的ID，是指用户的账户ID，可通过OBS控制台“我的凭证”页面查看。
- 当前OBS对象支持的可被授权的用户组为：
  - 所有用户：ObsClient.enums.GroupAllUsers

获取多版本对象访问权限

您可以通过ObsClient.getObjectAcl接口指定**VersionId参数**获取多版本对象的访问权限，示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObjectAcl({
        Bucket : 'bucketname',
        Key : 'objectname',
        VersionId : 'versionid'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                console.log('Owner[ID]-->' + result.InterfaceResult.Owner.ID);
                console.log('Owner[Name]-->' + result.InterfaceResult.Owner.Name);
                for(var i in result.InterfaceResult.Grants){
                    console.log('Grant[' + i + ']:');
                    console.log('Grantee[ID]-->' + result.InterfaceResult.Grants[i]['Grantee']['ID']);
                    console.log('Grantee[URI]-->' + result.InterfaceResult.Grants[i]['Grantee']['URI']);
                    console.log('Permission-->'+ result.InterfaceResult.Grants[i]['Permission']);
                }
            }
        }
});
```

12.9 删除多版本对象

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

删除单个多版本对象

您可以通过ObsClient.deleteObject接口指定**VersionId参数**删除多版本对象。

 版权所有 © 华为技术有限公司

本示例用于通过设置多版本对象的VersionId值，删除桶名为“bucketname”里，名称为“objectname”的对象。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.deleteObject({
        Bucket : 'bucketname',
        Key : 'objectname',
        VersionId : 'versionid'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

批量删除多版本对象

您可以通过ObsClient.deleteObjects接口传入每个待删除对象的**VersionId参数**批量删除多版本对象。

本示例用于通过设置多版本对象的VersionId值，批量删除桶名为“bucketname”里，名称为“objectname1”和“objectname2”的对象。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.deleteObjects({
        Bucket: 'bucketname',
        // 设置为verbose模式
        Quiet : false,
        Objects : [{Key:'objectname1', VersionId : 'version1'},{Key:'objectname2', VersionId : 'version2'}, {Key : 'objectname3', VersionId : 'version3'}]
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
```

```
                // 获取删除成功的对象
                console.log('Deleteds:');
                for(var i in result.InterfaceResult.Deleteds){
                    console.log('Deleted[' + i + ']:');
                    console.log('Key-->'+result.InterfaceResult.Deleteds[i]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.Deleteds[i]['VersionId']);
                }
                // 获取删除失败的对象
                console.log('Errors:');
                for(var j in result.InterfaceResult.Errors){
                    console.log('Error[' + j + ']:');
                    console.log('Key-->' + result.InterfaceResult.Errors[j]['Key']);
                    console.log('VersionId-->' + result.InterfaceResult.Errors[j]['VersionId']);
                }
            }
        }
});
```

 版权所有 © 华为技术有限公司

13 生命周期管理

13.1 生命周期管理简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS允许您对桶设置生命周期规则，实现自动转换对象的存储类型、自动淘汰过期的对象，以有效利用存储特性，优化存储空间。针对不同前缀的对象，您可以同时设置多条规则。一条规则包含：

- 规则ID，用于标识一条规则，不能重复。
- 受影响的对象前缀，此规则只作用于符合前缀的对象。
- 最新版本对象的转换策略，指定方式为：
  a. 指定满足前缀的对象创建后第几天时转换为指定的存储类型。
  b. 直接指定满足前缀的对象转换为指定的存储类型的日期。
- 最新版本对象过期时间，指定方式为：
  a. 指定满足前缀的对象创建后第几天时过期。
  b. 直接指定满足前缀的对象过期日期。
- 历史版本对象转换策略，指定方式为：
  - 指定满足前缀的对象成为历史版本后第几天时转换为指定的存储类型。
- 历史版本对象过期时间，指定方式为：
  - 指定满足前缀的对象成为历史版本后第几天时过期。
- 是否生效标识。

更多关于生命周期的内容请参考**生命周期管理**。

版权所有 © 华为技术有限公司

说明

- 对象过期后会被OBS服务端自动删除。
- 对象转换策略中的时间必须早于对象过期时间；历史版本对象转换策略中的时间也必须早于历史版本对象的过期时间。
- 桶的多版本状态必须处于Enabled或者Suspended，历史版本对象转换策略和历史版本对象过期时间配置才能生效。

13.2 设置生命周期规则

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.setBucketLifecycle设置桶的生命周期规则。

设置对象转换策略

以下代码展示了如何设置最新版本对象和历史版本对象的转换策略：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketLifecycle({
        Bucket: 'bucketname',
        Rules:[
            {
                ID:'rule1',Prefix:'prefix1',Status:'Enabled',
                // 指定满足前缀的对象创建30天后过期转换为低频访问存储
                Transitions:[{StorageClass: obsClient.enums.StorageClassWarm, Days:30}],
                // 指定满足前缀的对象成为历史版本30天后过期转换为归档存储
                NoncurrentVersionTransitions:[{StorageClass: obsClient.enums.StorageClassCold, NoncurrentDays : 30}]
            },
            {
                ID:'rule2',Prefix:'prefix2',Status:'Enabled',
                // 直接指定满足前缀的对象转换为低频访问存储的时间
                Transitions:[{StorageClass: obsClient.enums.StorageClassWarm, Date: '2018-10-31T00:00:00Z'}],
            }
        ]
}, function(err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```


版权所有 © 华为技术有限公司

设置对象过期时间

以下代码展示了如何设置最新版本对象和历史版本对象的过期时间：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketLifecycle({
        Bucket: 'bucketname',
        Rules:[
            {
                ID:'rule1',Prefix:'prefix1',Status:'Enabled',
                // 指定满足前缀的对象创建60天后过期
                Expiration:{Days:60},
                // 指定满足前缀的对象成为历史版本60天后过期
                NoncurrentVersionExpiration:{NoncurrentDays : 60}
            },
            {
                ID:'rule2',Prefix:'prefix2',Status:'Enabled',
                // 直接指定满足前缀的对象过期时间，该值必须符合ISO8601格式，而且必须是UTC午夜0点
                Expiration:{Date: '2018-12-31T00:00:00Z'},
            }
        ]
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

使用**Rules参数**指定桶的生命周期规则。

13.3 查看生命周期规则

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketLifecycle查看桶的生命周期规则。

本示例用于查看桶名为“bucketname”的生命周期规则。

代码示例如下所示：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存
```

 版权所有 © 华为技术有限公司

```
放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getBucketLifecycle({
        Bucket: 'bucketname'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                for(var i=0;i<result.InterfaceResult.Rules.length;i++){
                    console.log('Rule[' + i + ']:');
                    console.log('ID-->' + result.InterfaceResult.Rules[i]['ID']);
                    console.log('Prefix-->' + result.InterfaceResult.Rules[i]['Prefix']);
                    console.log('Status-->' + result.InterfaceResult.Rules[i]['Status']);
                    for(var j=0;j<result.InterfaceResult.Rules[i]['Transitions'].length;j++){
                        console.log('Transition[' + j + ']:');
                        console.log('Transition[StorageClass]-->' + result.InterfaceResult.Rules[i]['Transitions'][j]['StorageClass']);
                        console.log('Transition[Date]-->' + result.InterfaceResult.Rules[i]['Transitions'][j]['Date']);
                        console.log('Transition[Days]-->' + result.InterfaceResult.Rules[i]['Transitions'][j]['Days']);
                    }
                    console.log('Expiration[Date]-->' + result.InterfaceResult.Rules[i]['Expiration']['Date']);
                    console.log('Expiration[Days]-->' + result.InterfaceResult.Rules[i]['Expiration']['Days']);
                    for(var k=0;k<result.InterfaceResult.Rules[i]['NoncurrentVersionTransitions'].length;k++){
                        console.log('NoncurrentVersionTransition[' + k + ']:');
                        console.log('NoncurrentVersionTransition[StorageClass]-->' + result.InterfaceResult.Rules[i]['NoncurrentVersionTransitions'][k]['StorageClass']);
                        console.log('NoncurrentVersionTransition[NoncurrentDays]-->' + result.InterfaceResult.Rules[i]['NoncurrentVersionTransitions'][k]['NoncurrentDays']);
                    }
                    console.log('NoncurrentVersionExpiration[NoncurrentDays]-->' + result.InterfaceResult.Rules[i]['NoncurrentVersionExpiration']['NoncurrentDays']);
                }
            }
        }
});
```

说明

- 查看桶生命周期规则过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

13.4 删除生命周期规则

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.deleteBucketLifecycle删除桶的生命周期规则。

本示例用于删除桶名为“bucketname”的生命周期规则。

 版权所有 © 华为技术有限公司

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

说明

- 删除桶生命周期规则过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

 版权所有 © 华为技术有限公司

14 设置访问日志

14.1 日志简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS允许您对桶设置访问日志记录，设置之后对于桶的访问会被记录成日志，日志存储在OBS上您指定的目标桶中。

出于分析或审计等目的，用户可以开启日志记录功能。通过访问日志记录，桶的拥有者可以深入分析访问该桶的用户请求性质、类型或趋势。

当用户开启一个桶的日志记录功能后，OBS会自动对这个桶的访问请求记录日志，并生成日志文件写入用户指定的桶（即目标桶）中。

日志文件存放位置需要在开启桶日志功能时指定，可以存放到您拥有的，且与开启日志功能的桶位于同一区域的任一存储桶，当然也包括开启日志功能的桶本身。

更多关于访问日志的内容请参考**日志记录**。

14.2 开启桶日志

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.setBucketLogging开启桶日志功能。

版权所有 © 华为技术有限公司

说明

- 日志目标桶与源桶必须在同一个区域（region）。
- 如果桶的存储类型为低频访问存储或归档存储，则不能作为日志目标桶。

开启桶日志

以下代码展示了如何开启桶日志：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 设置桶的日志配置
obsClient.setBucketLogging({
    Bucket:'bucketname',
    // 通过IAM创建的委托名
    Agency: 'Agency name'
    LoggingEnabled:{
        // 生成日志文件的桶名
        TargetBucket: 'logBucketName',
        // 指定生成的日志文件的对象名前缀
        TargetPrefix: 'logs/',
    }
}, function(err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

使用**LoggingEnabled参数**指定桶的日志配置。

为日志对象设置权限

以下代码展示了如何为日志对象设置权限：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 设置桶的日志配置
obsClient.setBucketLogging({
    Bucket:'bucketname',
```

```
    // 目标桶Owner通过IAM创建对OBS服务的委托名称
    Agency: 'Agency name',
    LoggingEnabled:{
        // 生成日志文件的桶名
        TargetBucket: 'LogBucketName',
        // 指定生成的日志文件的对象名前缀
        TargetPrefix: 'logs/',
        TargetGrants:[
            // 为所有用户设置对日志对象的读权限
          { Grantee:
{Type:'Group',URI:obsClient.enums.GroupAllUsers},Permission:obsClient.enums.PermissionRead },
            // 为所有用户设置对日志文件的写权限
          { Grantee:
{Type:'Group',URI:obsClient.enums.GroupAllUsers},Permission:obsClient.enums.PermissionWrite }
        ]
     }
}, function(err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

14.3 查看桶日志配置

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketLogging查看桶日志配置。

本示例用于查看桶名为“bucketname”的日志配置。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getBucketLoggingConfiguration({
        Bucket:'bucketname'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                if(result.InterfaceResult.LoggingEnabled){
                    console.log('TargetBucket-->' + result.InterfaceResult.LoggingEnabled.TargetBucket);
                    console.log('TargetPrefix-->' + result.InterfaceResult.LoggingEnabled.TargetPrefix);
                }
```


版权所有 © 华为技术有限公司

```
                for(var i in result.InterfaceResult.LoggingEnabled.TargetGrants){
                    console.log('Grant[' + i + ']:');
                    console.log('Grantee[ID]-->' + result.InterfaceResult.LoggingEnabled.TargetGrants[i]
['Grantee']['ID']);
                    console.log('Grantee[URI]-->' + result.InterfaceResult.LoggingEnabled.TargetGrants[i]
['Grantee']['URI']);
                    console.log('Permission-->' + result.InterfaceResult.LoggingEnabled.TargetGrants[i]
['Permission']);
                }
            }
        }
});
```

说明

- 查看桶日志过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

14.4 关闭桶日志

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

关闭桶日志功能实际上就是调用ObsClient.setBucketLogging将日志配置清空。

本示例用于关闭桶名为“bucketname”的日志配置。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketLoggingConfiguration({
        Bucket:'bucketname',
        LoggingEnabled : {}
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

- 关闭桶日志过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

 版权所有 © 华为技术有限公司

15 静态网站托管

15.1 静态网站托管简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

静态网站通常仅包含静态网页，以及可能包含部分可在客户端运行的脚本，如JavaScript、Flash等。相比之下，动态网站则依赖于服务器端处理脚本，包括PHP、JSP或ASP.Net等。

您可以将静态网站文件上传至OBS的桶中作为对象，并对这些对象赋予公共读权限，然后将该桶配置成静态网站托管模式，以实现在OBS上托管静态网站的目的。

第三方用户在访问您网站的时候，实际上是在访问OBS的桶中的对象。

在使用静态网站托管功能时，OBS还支持配置请求重定向，通过重定向配置您可以将特定的请求或所有请求实施重定向。

更多关于静态网站托管的内容请参考**静态网站托管**。

15.2 网站文件托管

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可通过以下步骤实现网站文件托管：

步骤1 将网站文件上传至OBS的桶中，并设置对象MIME类型。

版权所有 © 华为技术有限公司

步骤2 设置对象访问权限为公共读。

步骤3 通过浏览器访问对象。

----结束

以下代码展示了如何实现网站文件托管：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 上传对象
obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'test.html',
        Body: '<html><header></header><body><h1>Hello OBS</h1></body></html>',
        // 设置对象MIME类型
        ContentType: 'text/html',
        // 设置对象访问权限为公共读
        ACL: obsClient.enums.AclPublicRead
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

上例中您可以使用http://bucketname.your-endpoint/test.html在浏览器直接访问托管的文件。

15.3 设置托管配置

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.setBucketWebsite设置桶的托管配置。

配置默认主页和错误页面

以下代码展示了如何配置默认主页和错误页面：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
```

 版权所有 © 华为技术有限公司

```
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketWebsite({
        Bucket: 'bucketname',
        // 配置默认主页
        IndexDocument:{Suffix:'index.html'},
        // 配置错误页面
        ErrorDocument:{Key:'error.html'}
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

配置重定向规则

以下代码展示了如何配置重定向规则：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存
放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境
变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/
intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketWebsite({
        Bucket: 'bucketname',
        // 配置默认主页
        IndexDocument:{Suffix:'index.html'},
        // 配置错误页面
        ErrorDocument:{Key:'error.html'},
        // 配置重定向规则
        RoutingRules:[

                {
                    Condition:{HttpErrorCodeReturnedEquals:'404', KeyPrefixEquals: 'keyprefix'},
                    Redirect:{Protocol:'http',ReplaceKeyWith:'replacekeyprefix', HostName:
'www.example.com', HttpRedirectCode: '305'}
                }
        ]
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

说明

使用**RoutingRules参数**指定桶的重定向规则。

配置所有请求重定向

以下代码展示了如何配置所有请求重定向：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.setBucketWebsite({
      Bucket: 'bucketname',
      RedirectAllRequestsTo : {HostName : 'www.example.com', Protocol : 'http'}
}, function (err, result) {
      if(err){
            console.log('Error-->' + err);
      }else{
            console.log('Status-->' + result.CommonMsg.Status);
      }
});
```

说明

使用**RedirectAllRequestsTo参数**指定桶的所有请求重定向规则。

15.4 查看托管配置

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketWebsite查看桶的托管配置。

本示例用于查看桶名为“bucketname”的托管配置。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getBucketWebsite({
      Bucket: 'bucketname'
```


版权所有 © 华为技术有限公司

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

说明

- 查看桶托管配置过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

15.5 清除托管配置

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.deleteBucketWebsite清除桶的托管配置。

本示例用于清除桶名为“bucketname”的托管配置。

代码示例如下所示:

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.deleteBucketWebsite({
```


版权所有 © 华为技术有限公司

```
    Bucket: 'bucketname'
}, function (err, result) {
    if(err){
        console.log('Error-->' + err);
    }else{
        console.log('Status-->' + result.CommonMsg.Status);
    }
});
```

说明

- 清除桶托管配置过程中返回的错误码含义、问题原因及处理措施可参考**OBS服务端错误码**

版权所有 © 华为技术有限公司

16 标签管理

16.1 标签简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

标签用于标识OBS中的桶，以此来达到对OBS中的桶进行分类的目的。

更多关于桶标签的内容请参考**标签**。

16.2 设置桶标签

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.setBucketTagging设置桶标签。以下代码展示了如何设置桶标签：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
```


版权所有 © 华为技术有限公司

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

说明

- 使用**TagSet参数**指定桶的标签信息。
- 每个桶支持最多10个标签。
- 标签的Key和Value支持Unicode。

16.3 查看桶标签

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.getBucketTagging查看桶标签。以下代码展示了如何查看桶标签：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

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

 版权所有 © 华为技术有限公司

16.4 删除桶标签

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

您可以通过ObsClient.deleteBucketTagging删除桶标签。以下代码展示了如何删除桶标签：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.deleteBucketTagging({
        Bucket: 'bucketname'
}, function (err, result) {
        if(err){
                console.log('Error-->' + err);
        }else{
                console.log('Status-->' + result.CommonMsg.Status);
        }
});
```


版权所有 © 华为技术有限公司

17 服务端加密

17.1 服务端加密简介

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS支持服务端加密功能，使对象加密的行为在OBS服务端进行。

更多关于服务端加密的内容请参考**服务端加密**。

17.2 加密说明

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

OBS BrowserJS SDK支持服务端加密的接口见下表：

OBS BrowserJS SDK接口方法	描述	支持加密类型
ObsClient.putObject	上传对象时设置加密算法、密钥，对对象启用服务端加密。	SSE-KMS SSE-C
ObsClient.getObject	下载对象时设置解密算法、密钥，用于解密对象。	SSE-C

版权所有 © 华为技术有限公司

OBS BrowserJS SDK接口方法	描述	支持加密类型
ObsClient.copyObject	1. 复制对象时设置源对象的解密算法、密钥，用于解密源对象。 2. 复制对象时设置目标对象的加密算法、密钥，对目标对象启用加密算法。	SSE-KMS SSE-C
ObsClient.getObjectMetadata	获取对象元数据时设置解密算法、密钥，用于解密对象。	SSE-C
ObsClient.initiateMultipartUpload	初始化分段上传任务时设置加密算法、密钥，对分段上传任务最终生成的对象启用服务端加密。	SSE-KMS SSE-C
ObsClient.uploadPart	上传段时设置加密算法、密钥，对分段数据启用服务端加密。	SSE-C
ObsClient.copyPart	1. 复制段时设置源对象的解密算法、密钥，用于解密源对象。 2. 复制段时设置目标段的加密算法、密钥，对目标段启用加密算法。	SSE-C

OBS BrowserJS SDK两种加密方式支持的请求参数：

加密类型	OBS BrowserJS SDK对应请求参数	说明
SSE-KMS	SseKms	表示服务端加密是SSE-KMS方式，目前仅支持：**kms**。
	SseKmsKey	表示SSE-KMS方式下的主密钥，可为空。
SSE-C	SseC	表示服务端加密是SSE-C方式，目前仅支持：**AES256**。
	SseCKey	表示SSE-C方式下的密钥，由AES256算法得到。上传对象时作为加密密钥；下载对象时作为解密密钥。注意：不需要base64编码处理。
	CopySourceSseC	适用于ObsClient.copyObject和ObsClient.copyPart，表示以SSE-C方式解密源对象，目前仅支持：**AES256**。
	CopySourceSseCKey	适用于ObsClient.copyObject和ObsClient.copyPart，表示以SSE-C方式解密源对象时使用的密钥，由AES256算法得到。

版权所有 © 华为技术有限公司

17.3 加密示例

须知

开发过程中，您有任何问题可以在github上**提交issue**。**接口参考文档**详细介绍了每个接口的参数和使用方法。

上传对象加密

以下代码展示了在上传对象时使用服务端加密功能：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'objectname',
        SourceFile: document.getElementById('input-file').files[0],
        // 设置SSE-C算法加密对象
        SseC: 'AES256',
        // 设置AES-256原始字符串，非base64编码后的密钥
        SseCKey: 'your sse-c key generated by AES-256 algorithm'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});

obsClient.putObject({
        Bucket: 'bucketname',
        Key: 'objectname2',
        SourceFile: document.getElementById('input-file2').files[0],
        // 设置SSE-KMS算法加密对象
        SseKms: 'kms'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
        }
});
```

下载对象解密

以下代码展示了在下载对象时使用服务端解密功能：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
```


版权所有 © 华为技术有限公司

```
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.getObject({
        Bucket: 'bucketname',
        Key: 'objectname',
        // 设置SSE-C算法解密对象
        SseC: 'AES256',
        // 此处的密钥必须和上传对象加密时使用的密钥一致
        SseCKey: 'your sse-c key generated by AES-256 algorithm'
}, function (err, result) {
        if(err){
            console.log('Error-->' + err);
        }else{
            console.log('Status-->' + result.CommonMsg.Status);
            if(result.CommonMsg.Status < 300 && result.InterfaceResult){
                console.log('Content-->' + result.InterfaceResult.Content);
            }
        }
});
```

 版权所有 © 华为技术有限公司

18 异常处理

18.1 HTTP 状态码

OBS服务端遵照HTTP规范，在接口调用完成均会返回标准的HTTP状态码，HTTP状态码分类以及OBS中常见的HTTP状态码如下：

- HTTP状态码分类：

分类	分类描述
1XX	信息，服务器收到请求，需要请求者继续执行操作，一般对客户调用函数不可见。
2XX	成功，操作被成功接收并处理。
3XX	重定向，需要进一步的操作以完成请求。
4XX	客户端错误，请求包含语法错误或无法完成请求。
5XX	服务器错误，服务器在处理请求的过程中发生了错误

- OBS中常见的HTTP状态码及其含义：

HTTP状态码	描述	常见原因
400 Bad Request	请求参数错误	• 请求参数不合法； • 客户端携带MD5请求后一致性校验失败； • 无效的参数（使用SDK时传递了不合法的参数）； • 无效的桶名（使用了不合法的桶名）；

 版权所有 © 华为技术有限公司

HTTP状态码	描述	常见原因
403 Forbidden	拒绝访问	• 请求的签名不匹配（一般是AK/SK错误）； • 权限不足（账号对请求的资源无权限）； • 账号欠费； • 桶的空间不足（出现在对桶设置了配额的场景）； • 无效的AK； • 客户端时间和服务端时间相差过大（客户端所在机器的时间与NTP服务不同步）；
404 Not Found	请求的资源不存在	• 桶不存在； • 对象不存在； • 桶的策略配置不存在（桶CORS配置不存在、桶Policy配置不存在等）； • 分段上传任务不存在；
405 Method Not Allowed	请求的方法不支持	请求的方法/特性未在该桶所在的区域上线
408 Request Timeout	请求超时	服务端与客户端Socket连接超时
409 Conflict	请求冲突	• 在不同区域重复创建桶名桶； • 尝试删除非空桶；
500 Internal Server Error	服务端内部错误	服务端内部错误
503 Service Unavaliable	服务不可用	服务端暂时不可访问

18.2 OBS 服务端错误码

在向OBS服务端发出请求后，如果遇到错误，会在响应中包含响应的错误码描述错误信息。详细的错误码及其对应的描述和HTTP状态码见下表：

HTTP状态码	错误码	错误信息	处理措施
301 Moved Permanently	PermanentRedirect	尝试访问的桶必须使用指定的地址，请将以后的请求发送到这个地址。	按照返回的重定向地址发送请求。

版权所有 © 华为技术有限公司

HTTP状态码	错误码	错误信息	处理措施
301 Moved Permanently	WebsiteRedirect	Website请求缺少bucketName。	携带桶名后重试。
307 Moved Temporarily	TemporaryRedirect	临时重定向，当DNS更新时，请求将被重定向到桶。	会自动重定向，也可以将请求发送到重定向地址。
400 Bad Request	BadDigest	客户端指定的对象内容的MD5值与系统接收到的内容MD5值不一致。	检查头域中携带的MD5与消息体计算出来的MD5是否一致。
400 Bad Request	BadDomainName	域名不合法。	使用合法的域名。
400 Bad Request	BadRequest	请求参数不合法。	根据返回的错误消息体提示进行修改。
400 Bad Request	CustomDomainAreadyExist	配置了已存在的域。	已经配置过了，不需要再配置。
400 Bad Request	CustomDomainNotExist	删除不存在的域。	未配置或已经删除，无需删除。
400 Bad Request	EntityTooLarge	用户POST上传的对象大小超过了条件允许的最大大小。	修改POST上传的policy中的条件或者减少对象大小。
400 Bad Request	EntityTooSmall	用户POST上传的对象大小小于条件允许的最小大小。	修改POST上传的policy中的条件或者增加对象大小。
400 Bad Request	IllegalLocationConstraintException	用户未带Location在非默认Region创桶。	请求发往默认Region创桶或带非默认Region的Location创桶。
400 Bad Request	IncompleteBody	由于网络原因或其他问题导致请求体未接受完整。	重新上传对象。
400 Bad Request	IncorrectNumberOfFilesInPostRequest	每个POST请求都需要带一个上传的文件。	带上一个上传文件。
400 Bad Request	InvalidArgument	无效的参数。	根据返回的错误消息体提示进行修改。
400 Bad Request	InvalidBucket	请求访问的桶已不存在。	更换桶名。

版权所有 © 华为技术有限公司

HTTP状态码	错误码	错误信息	处理措施
400 Bad Request	InvalidBucketName	请求中指定的桶名无效，超长或带不允许的特殊字符。	更换桶名。
400 Bad Request	InvalidEncryptionAlgorithmError	错误的加密算法。下载SSE-C加密的对象，携带的加密头域错误，导致不能解密。	携带正确的加密头域下载对象。
400 Bad Request	InvalidLocationConstraint	创建桶时，指定的Location不合法或不存在。	指定正确的Location创桶。
400 Bad Request	InvalidPart	一个或多个指定的段无法找到。这些段可能没有上传，或者指定的entity tag与段的entity tag不一致。	按照正确的段和entity tag合并段。
400 Bad Request	InvalidPartOrder	段列表的顺序不是升序，段列表必须按段号升序排列。	按段号升序排列后重新合并。
400 Bad Request	InvalidPolicyDocument	表单中的内容与策略文档中指定的条件不一致。	根据返回的错误消息体提示修改构造表单的policy重试。
400 Bad Request	InvalidRedirectLocation	无效的重定向地址。	指定正确的地址。
400 Bad Request	InvalidRequest	无效请求。	根据返回的错误消息体提示进行修改。
400 Bad Request	InvalidRequestBody	请求体无效，需要消息体的请求没有上传消息体。	按照正确的格式上传消息体。
400 Bad Request	InvalidTargetBucketForLogging	delivery group对目标桶无ACL权限。	对目标桶配置ACL权限后重试。
400 Bad Request	KeyTooLongError	提供的Key过长。	使用较短的Key。
400 Bad Request	KMS.DisabledException	SSE-KMS加密方式下，主密钥被禁用。	更换密钥后重试，或联系技术支持。
400 Bad Request	KMS.NotFoundException	SSE-KMS加密方式下，主密钥不存在。	携带正确的主密钥重试。

版权所有 © 华为技术有限公司

HTTP状态码	错误码	错误信息	处理措施
400 Bad Request	MalformedACLError	提供的XML格式错误，或者不符合我们要求的格式。	使用正确的XML格式重试。
400 Bad Request	MalformedError	请求中携带的XML格式不正确。	使用正确的XML格式重试。
400 Bad Request	MalformedLoggingStatus	Logging的XML格式不正确。	使用正确的XML格式重试。
400 Bad Request	MalformedPolicy	Bucket policy检查不通过。	根据返回的错误消息体提示结合桶policy的要求进行修改。
400 Bad Request	MalformedQuotaError	Quota的XML格式不正确。	使用正确的XML格式重试。
400 Bad Request	MalformedXML	当用户发送了一个配置项的错误格式的XML会出现这样的错误。	使用正确的XML格式重试。
400 Bad Request	MaxMessageLengthExceeded	拷贝对象，带请求消息体。	拷贝对象不带消息体重试。
400 Bad Request	MetadataTooLarge	元数据消息头超过了允许的最大元数据大小。	减少元数据消息头。
400 Bad Request	MissingRegion	请求中缺少Region信息，且系统无默认Region。	请求中携带Region信息。
400 Bad Request	MissingRequestBodyError	当用户发送一个空的XML文档作为请求时会发生。	提供正确的XML文档。
400 Bad Request	MissingRequiredHeader	请求中缺少必要的头域。	提供必要的头域。
400 Bad Request	MissingSecurityHeader	请求缺少一个必须的头。	提供必要的头域。
400 Bad Request	TooManyBuckets	用户拥有的桶的数量达到了系统的上限，并且请求试图创建一个新桶。	删除部分桶后重试。
400 Bad Request	TooManyCustomDomains	配置了过多的用户域	删除部分用户域后重试。
400 Bad Request	TooManyWrongSignature	因高频错误请求被拒绝服务。	更换正确的Access Key后重试。

版权所有 © 华为技术有限公司

HTTP状态码	错误码	错误信息	处理措施
400 Bad Request	UnexpectedContent	该请求需要消息体而客户端没带，或该请求不需要消息体而客户端带了。	根据说明重试。
400 Bad Request	UserKeyMustBeSpecified	该操作只有特殊用户可使用。	请联系技术支持。
403 Forbidden	AccessDenied	拒绝访问，请求没有携带日期头域或者头域格式错误。	请求携带正确的日期头域。
403 Forbidden	AccessForbidden	权限不足，桶未配置CORS或者CORS规则不匹配。	修改桶的CORS配置，或者根据桶的CORS配置发送匹配的OPTIONS请求。
403 Forbidden	AllAccessDisabled	用户无权限执行某操作。桶名为禁用关键字。	更换桶名。
403 Forbidden	DeregisterUserId	用户已经注销。	充值或重新开户。
403 Forbidden	InArrearOrInsufficientBalance	用户欠费或余额不足而没有权限进行某种操作。	充值。
403 Forbidden	InsufficientStorageSpace	存储空间不足。	超过配额限制，增加配额或删除部分对象。
403 Forbidden	InvalidAccessKeyId	系统记录中不存在客户提供的Access Key Id。	携带正确的Access Key Id。
403 Forbidden	NotSignedUp	你的账户还没有在系统中注册，必须先在系统中注册了才能使用该账户。	先注册OBS服务。
403 Forbidden	RequestTimeTooSkewed	请求的时间与服务器的时间相差太大。	检查客户端时间是否与当前时间相差太大。
403 Forbidden	SignatureDoesNotMatch	请求中带的签名与系统计算得到的签名不一致。	检查你的Secret Access Key和签名计算方法。
403 Forbidden	Unauthorized	用户未实名认证。	请实名认证后重试。
404 Not Found	NoSuchBucket	指定的桶不存在。	先创桶再操作。

 版权所有 © 华为技术有限公司

HTTP状态码	错误码	错误信息	处理措施
404 Not Found	NoSuchBucketPolicy	桶policy不存在。	先配置桶policy。
404 Not Found	NoSuchCORSConfiguration	CORS配置不存在。	先配置CORS。
404 Not Found	NoSuchCustomDomain	请求的用户域不存在。	先设置用户域。
404 Not Found	NoSuchKey	指定的Key不存在。	先上传对象。
404 Not Found	NoSuchLifecycleConfiguration	请求的LifeCycle不存在。	先配置LifeCycle。
404 Not Found	NoSuchUpload	指定的多段上传不存在。Upload ID不存在，或者多段上传已经终止或完成。	使用存在的段或重新初始化段。
404 Not Found	NoSuchVersion	请求中指定的version ID与现存的所有版本都不匹配。	使用正确的version ID。
404 Not Found	NoSuchWebsiteConfiguration	请求的Website不存在。	先配置Website。
405 Method Not Allowed	MethodNotAllowed	指定的方法不允许操作在请求的资源上。 对应返回的Message为：Specified method is not supported.	方法不允许。
408 Request Timeout	RequestTimeout	用户与Server之间的socket连接在超时时间内没有进行读写操作。	检查网络后重试，或联系技术支持。
409 Conflict	BucketAlreadyExists	请求的桶名已经存在。桶的命名空间是系统中所有用户共用的，选择一个不同的桶名再重试一次。	更换桶名。
409 Conflict	BucketAlreadyOwnedByYou	发起该请求的用户已经创建过了这个名字的桶，并拥有这个桶。	不需要再创桶了。

版权所有 © 华为技术有限公司

HTTP状态码	错误码	错误信息	处理措施
409 Conflict	BucketNotEmpty	用户尝试删除的桶不为空。	先删除桶中对象，然后再删桶。
409 Conflict	InvalidBucketState	无效的桶状态，配置跨Region复制后不允许关闭桶多版本。	不关闭桶的多版本或取消跨Region复制。
409 Conflict	OperationAborted	另外一个冲突的操作当前正作用在这个资源上，请重试。	等待一段时间后重试。
409 Conflict	ServiceNotSupported	请求的方法服务端不支持。	服务端不支持，请联系技术支持。
411 Length Required	MissingContentLength	必须要提供HTTP消息头中的Content-Length字段。	提供Content-Length消息头。
412 Precondition Failed	PreconditionFailed	用户指定的先决条件中至少有一项没有包含。	根据返回消息体中的Condition提示进行修改。
416 Client Requested Range Not Satisfiable	InvalidRange	请求的range不可获得。	携带正确的range重试。
500 Internal Server Error	InternalError	系统遇到内部错误，请重试。	请联系技术支持。
501 Not Implemented	ServiceNotImplemented	请求的方法服务端没有实现。	当前不支持，请联系技术支持。
503 Service Unavailable	ServiceUnavailable	服务器过载或者内部错误异常。	等待一段时间后重试，或联系技术支持。
503 Service Unavailable	SlowDown	请降低请求频率。	请降低请求频率。

18.3 SDK 公共结果对象

调用ObsClient的相关接口完成后，如异常信息参数为空，则均会返回公共结果对象。该对象包含的内容见下表：

字段名	类型	说明
CommonMsg	Object	接口调用完成后的公共信息，包含HTTP状态码，操作失败的错误码等。

版权所有 © 华为技术有限公司

字段名		类型	说明
-	Status	Number	HTTP状态码，小于300表明操作成功；反之，表明操作失败。
	Code	String	OBS服务端错误码，当Status小于300时为空。
	Message	String	OBS服务端错误描述，当Status小于300时为空。
	HostId	String	请求的服务端ID，当Status小于300时为空。
	RequestId	String	OBS服务端返回的请求ID。
	Id2	String	OBS服务端返回的请求ID2。
	Indicator	String	OBS服务端返回的详细错误码，当Status小于300时为空。
InterfaceResult		Object	操作成功后的结果数据，当Status大于300时为空。
-	RequestId	String	OBS服务端返回的请求ID。
	Id2	String	OBS服务端返回的请求ID2。
	其他字段	请查阅《对象存储服务BrowserJS SDK API参考》。	

处理公共结果对象的示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

// 调用接口进行操作，例如下载对象
obsClient.getObject({
        Bucket : 'bucketname',
        Key : 'objectname',
}, function (err, result) {
        if(!err){
            if(result.CommonMsg.Status < 300){
                // 获取RequestId
                console.log('RequestId-->' + result.InterfaceResult.RequestId);
                // 获取其他参数
                console.log('Content-->' + result.InterfaceResult.Content);
            }else{
                // 获取Code和Message
                console.log('Code-->' + result.CommonMsg.Code);
                console.log('Message-->' + result.CommonMsg.Message);
            }
        }
});
```

 版权所有 © 华为技术有限公司

18.4 日志分析

日志配置

OBS BrowserJS SDK提供了日志功能，您可以通过ObsClient.initLog开启日志功能并进行配置。示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

obsClient.initLog({
      level:'warn', // 配置日志级别
});
```

说明

- SDK打印的日志均显示在浏览器提供的开发者工具的Console中。
- 日志功能默认是关闭的，需要主动开启。

日志内容格式

SDK日志格式为：日志时间|日志级别|调用接口|日志内容。示例如下：

```
2018/2/11 下午9:22:45|info|ListObjects|enter ListObjects...
2018/2/11 下午9:22:45|info|ListObjects|prepare request parameters ok,then Send request to service start
2018/2/11 下午9:22:45|info|ListObjects|http cost 19 ms
2018/2/11 下午9:22:45|info|ListObjects|get response start, statusCode:200
```

日志级别

当系统出现问题需要定位且当前的日志无法满足要求时，可以通过修改日志的级别来获取更多的信息。其中debug日志信息最丰富，error日志信息最少。

具体说明如下：

- debug：调试级别，如果设置为这个级别，将打印SDK记录的所有日志。
- info：信息级别，如果设置为这个级别，除了打印warn级别的信息外，还将打印HTTP/HTTPS请求的耗时时间等信息。
- warn：告警级别，如果设置为这个级别，除了打印error级别的信息外，还将打印一些关键事件的信息。
- error：错误级别，如果设置为这个级别，仅打印发生异常时的错误信息。

 版权所有 © 华为技术有限公司

19 FAQ

19.1 如何在不支持 window.File 的浏览器上传文件?

在不支持window.File的浏览器上（例如低版本的IE浏览器），不能使用ObsClient.putObject或ObsClient.uploadFile上传对象，如果要支持这类浏览器上传，请参考**基于表单上传**。

19.2 如何使对象可以被匿名用户访问?

如果想对象可以被匿名用户，可通过以下三步完成。

步骤1 参考**管理对象访问权限**章节，设置对象的访问权限为公共读。

步骤2 参考**如何获取对象URL**章节，获取对象的URL提供给匿名用户。

步骤3 匿名用户通过浏览器打开获取的URL，可访问到该对象。

----结束

19.3 如何获取桶的静态网站访问地址?

桶配置成静态网站托管模式后，可通过以下方式拼接桶的静态网站访问地址：

https://*桶名.静态网站托管域名*

说明

- 各区域对应的静态网站托管域名可以从**这里**查看。

19.4 如何获取对象 URL?

按**https://*桶名.域名/文件夹目录层级/对象名***的方式进行拼接。

 版权所有 © 华为技术有限公司

说明

- 如果该对象存在于桶的根目录下，则链接地址将不需要有文件夹目录层级。
- 各区域对应的域名可以从**这里**的终端节点查看。
- 例如需访问区域为“中国-香港”的桶名为“testbucket”中“test”文件夹下对象名为“test.txt”的对象，则该对象的URL为https://testbucket.obs.ap-southeast-1.myhuaweicloud.com/test/test.txt。

19.5 公网环境下如何提高上传大文件速度?

在公网环境下对于超过100MB的大文件建议通过分段上传方式上传。分段上传是将单个对象拆分为一系列段分别上传。每个段都是对象数据的连续部分。您可以按照任意顺序上传段。如果其中某个段传输失败，可以重新传输该段且不会影响其他段。通过多线程并发上传同一对象的多个段，可大大提高传输效率。

具体代码样例可参见**分段上传**。您也可以使用SDK提供的**断点续传上传接口**进行大文件上传。

19.6 如何暂停断点续传上传任务?

断点续传上传接口支持暂停上传任务，示例代码如下：

```
// 创建ObsClient实例
var obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 前端本身没有process对象，可以使用webpack类打包工具定义环境变量，就可以在代码中运行了。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});

var uploadCheckpoint;
obsClient.uploadFile({
        Bucket : 'bucketname',
        Key : 'objectname',
        SourceFile : document.getElementById('input-file').files[0],
        PartSize : 9 * 1024 * 1024,
        ProgressCallback : function(transferredAmount, totalAmount, totalSeconds){
            // 获取上传进度
            console.log(transferredAmount * 1.0 / totalSeconds / 1024);
            console.log(transferredAmount * 100.0 / totalAmount);
        },
        ResumeCallback : function(resumeHook, uploadCheckpoint){
            // 运行3秒钟后暂停上传，暂停后接口会返回错误
            setTimeout(function(){
                hook.cancel();
            }, 3000);
            // 记录断点
            cp = uploadCheckpoint;
        }
}, function(err, result){
    console.error('Error-->' + err);
    // 出现错误，再次调用断点续传接口，继续上传任务
    if(err){
        obsClient.uploadFile({
            UploadCheckpoint : uploadCheckpoint,
```

 版权所有 © 华为技术有限公司

```
        ProgressCallback : function(transferredAmount, totalAmount, totalSeconds){
            // 获取上传进度
            console.log(transferredAmount * 1.0 / totalSeconds / 1024);
            console.log(transferredAmount * 100.0 / totalAmount);
        },
      }, function(err, result){
        // 再次处理回调函数
      });
   }else {
       console.log('Status-->' + result.CommonMsg.Status);
       console.log('RequestId-->' + result.CommonMsg.RequestId);
   }
});
```

19.7 如何在不暴露 AKSK 的条件下实现与 OBS 交互?

使用BrowserJS SDK与OBS服务交互时，必须将AK/SK暴露到前端，这样会存在安全问题。为避免该问题出现，可让后端生成临时签名URL给前端后，前端再与OBS交互。

比如：实现上传功能。

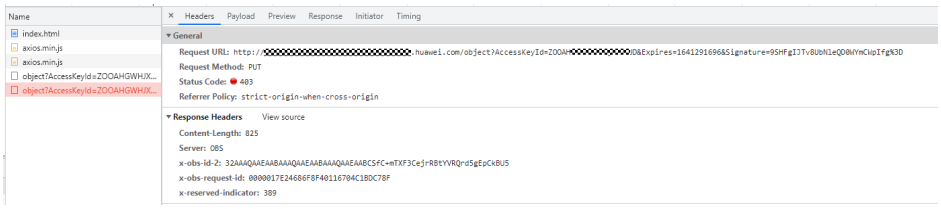
```
// 后端使用NodeJS SDK
// 引入obs库
const ObsClient = require('esdk-obs-nodejs');
// 初始化
const obsClient = new ObsClient({
    // 认证用的ak和sk硬编码到代码中或者明文存储都有很大的安全风险，建议在配置文件或者环境变量中密文存放，使用时解密，确保安全；本示例以ak和sk保存在环境变量中为例，运行本示例前请先在本地环境中设置环境变量AccessKeyID和SecretAccessKey。
    // 您可以登录访问管理控制台获取访问密钥AK/SK，获取方式请参见https://support.huaweicloud.com/intl/zh-cn/usermanual-ca/ca_01_0003.html
    access_key_id: process.env.AccessKeyID,
    secret_access_key: process.env.SecretAccessKey,
    // 这里以中国-香港为例，其他地区请按实际情况填写
    server: 'https://obs.ap-southeast-1.myhuaweicloud.com'
});
// 以下三个参数由前端传给后端，方便后端计算签名
// 桶名
const bucketName = 'bucketName';
// 对象名称
const objectKey = 'object';
// Http请求方法
const method = 'PUT'
// 添加Content-Type,按文件类型填写，此处以text/plain为例
const headers = {
    'Content-Type': 'text/plain'
}
// 计算签名URL，然后返给前端
const res = obsClient.createSignedUrlSync({ Method : method, Bucket : bucketName, Key: objectKey, Expires: 3600, Headers : headers });

// 前端拿到计算签名URL，向OBS发起Http请求
// 前端使用axios请求库
const url = res.SignedUrl;
const file = document.getElementById('input[type=file]')[0].file
if (!file) {
   console.log('your file is undefined')
}

axios.put(url, file, { headers })
  .then(res => console.log(res))
  .catch(err => console.error(err));
```

 版权所有 © 华为技术有限公司

⚠ 注意

在使用该方案时可能会遇到跨域问题，请按照以下步骤依次排查问题：

1. 未配置跨域，需要在控制台配置CORS规则，请参考**配置桶允许跨域请求**。
2. 签名计算问题，请参考**URL中携带签名**排查签名参数是否正确，比如上述demo中，**Axios的PUT方法会自动添加请求头**，而后端生成的临时授权URL并没有参与计算，这个时候就会出现跨域问题，查看network标签查看相应的请求会出现403状态码，如下图所示。



19.8 如何上传 base64 编码的图片

base64先转码成指定格式图片，然后调用OBS上传接口进行上传。

```
const base64ImgtoFile = function base64ImgtoFile(base64Content, filename) {
    const arr = base64Content.split(',');
    const mime = arr[0].match(/:(.*?);/)[1];
    const bstr = atob(arr[1]);
    let n = bstr.length;
    const u8arr = new Uint8Array(n);
    while (n--) {
        u8arr[n] = bstr.charCodeAt(n);
    }
    // 如果环境支持文件格式, 也可以使用: return new File([u8arr], filename, { type: mime });
    return new Blob([u8arr], { type: mime });
};
// obsClient表示OBS client实例
const uploadBase64Img = function uploadBase64Img(obsClient) {
    // base64格式的内容
    const base64Content = "data:image:xxxxxxxxxxxxx";
    const filename = 'img.png';
    const imgfile = base64ImgtoFile(base64Content, filename);
    obsClient.putObject({
        Bucket: 'bucketname',
        Key: filename,
        SourceFile: imgfile
    }, function (err, result) {
        if (err) {
            console.error('Error-->' + err);
        } else {
            console.log('Status-->' + result.CommonMsg.Status);
        }
    });
};
```

19.9 如何解决断点续传接口报 400 InvalidPart 的错误?

可能存在的原因有：

1. 合并段时请求段列表中包含了不存在的段；
2. 合并段时请求段列表中包含的段的Etag错误。

 版权所有 © 华为技术有限公司

可按照以下步骤排查原因：

步骤1 打开浏览器的开发者工具。

步骤2 检查合并段接口的请求体Body是否符合API接口规范。

- 如果发现请求体Body中的ETag值都是undefined，则说明ETag字段未配置成CORS的扩展头域，请参考**配置桶的CORS**章节进行配置，符合原因2的场景。

步骤3 如果合并段请求体Body符合API接口规范，请继续排查合并段请求体Body体内容是否合理，以及Etag值的一致性。

- 排查请求段列表中是否包含了不存在的段信息，例如：对象A分为a、b、c三个段上传，但是段列表中，多出了一个不存在的段d。此时您需要删除多余段的相关信息，此场景符合原因1。
- 排查过程中段ETag值是否和服务端返回的ETag值一致，如果不一致，请修改请求体中携带的Etag值为正确值，此场景符合原因2。

----结束

 版权所有 © 华为技术有限公司