京东云存储 PHP SDK 使用说明

更新时间	更新内容
2013/06/15	创建文档
2013/12/06	添加 MultipartUpload 相关内容

目录

1 系统要求

京东云存储 PHP SDK 需要 CURL 库支持，在使用前请检查您系统中安装的 PHP 是否已支持 PHP CURL。

2 基本概念

2.1 AccessKey && AccessSecret

京东云存储所有 API 操作都需要对请求做 AccessSecret 签名的校验，摘要字段将作为请求头发送给云存储系统，所以在使用 PHP SDK 时必须先设定有效的 AccessKey 和 AccessSecret。

AccessSecret 涉及您存储资料的安全性，所以请妥善保存您的 AccessSecret，不要泄漏给第三方。

注意：任何时候 AccessSecret 都不应作为请求头或内容发送给京东云存储系统。

1） **AccessKey**

AccessKey 由京东云存储颁发，用于标识用户的唯一身份。AccessKey 在所有的操作中都会被使用，并且以明文形式传输。

2） **AccessSecret**

AccessSecret 由京东云存储颁发，用于请求头部的签名。AccessSecret 总是随同 AccessKey 一起分发，一个 AccessKey 对应一个 AccessSecret。

2.2 Bucket

Bucket 是数据存储的基本容器。你可以把 Bucket 想象成文件系统中的目录，与目录不同的是 Bucket 不支持嵌套。

在用户空间内，用户可以根据需要创建不同的 Bucket。京东云存储系统中每个用户空间内最多只能创建 **1000** 个 Bucket。

注意：京东云存储系统中 Bucket 名称全局唯一，即在云存储系统中任何 Bucket 名称都不能重复。例如：用户 A 创建了名为“a_bucket”的 Bucket 后，用户 B 尝试创建名字同样为“a_bucket”的 Bucket 将会失败。

1） **Bucket 命名规则**

- 由小写字母或数字或点号(.)或下划线(_)或破折号(-)组合而成
- 开头必须是小写字母或数字
- 长度必须大于等于 3 字节且小于等于 255 字节
- 不能是一个 IP 地址形式，如 192.168.1.1 是不允许的
- 不能以 jingdong 作为 Bucket 名称前缀

- 如果希望以后提供 DNS 解析，则 Bucket 命名还必须需符合 DNS 主机名的命名

2.3 Object

Object 是数据存储的的基本单元。用户在京东云存储中存储的任何内容都以 Object 形式存在。每个 Object 由两个部分组成：Object Data 和 Object Meta。Object Data 即存储对象的实际内容数据，Object Meta 为包含对象属性的一系列 Key-Value 键值对。Object 必须存储在 Bucket 下。

注意：同一 Bucket 下的 Object 名称必须唯一！

1） **Object 命名规则**

- 使用 UTF-8 编码
- 长度必须大于等于 1 字节且小于等于 1024 字节

3 PHP SDK 概述

京东云存储 PHP SDK 为多个文件，核心文件为 JDCloudStorage.php，通常您只需要导入该文件便可完成所有 SDK 目前能提供的所有接口。

代码范例：

```
<?php
require_once dirname(__FILE__).'/../sdk/JDCloudStorage.php';
$service = new JDCloudStorage("ACCESS_KEY","ACCESS_SECRET");
$bucket = "your bucket"
$local_file = '/path/to/your/local/file';
$key = "logo.png";
try{
        // list bucket
        $service->list_buckets();
        //create new    bucket
        $service->put_bucket($bucket);
        //put object
        $service->put_object($bucket,$key,$local_file);
        //head object
        $service->head_object($bucket,$key);
        //get object
        $service->get_object($bucket,$key,dirname(__FILE__) . "/".$key);
        //get pre-sign url
        $url = $service->get_object_resource($bucket,$key);
                echo "url for {$bucket}/{$key} :".$url."\n";

        // list objects
        $jss_entity = $service->list_objects($bucket);
```

```
            $object_list = $jss_entity->get_object();
        foreach($object_list as $object){
            // delete object
            $service->delete_object($bucket,$object->get_key());
            echo "Delete object {$object->get_key()} success!\n";
        }
        // delete bucket
        $service->delete_bucket($bucket);
        } catch (JSSError $e) {
            echo "HTTP Code : ".$e->getCode()."\n"; //获取http返回错误码
          echo "Error Code : ".$e->getErrorCode()."\n"; //获取错误信息码
          echo "Error Message:".$e->getErrorMessage()."\n";//获取错误信息
          echo "RequestId : ".$e->getRequestId()."\n"; //获取请求ID
          echo "requestResource : ".$e->getResource()."\n";//获取请求资源

        } catch (Exception $e) {
            echo "Error Code : ".$e->getCode()."\n";
            echo "Error Message : ".$e->getMessage()."\n";

}
?>
```

4 PHP SDK 文档

PHP SDK 定义了三个用于云存储的对象：

- JSSBucket：每个 JSSBucket 对象表示一个京东云存储 Bucket；
- JSSObject：每个 JSSObject 对象表示一个京东云存储 Object；
- JSSEntity：每个 JSSEntity 对象表示一个京东云存储 Bucket 及其包含的 Object 列表的集合；

PHP SDK 定义了一个通用的异常对象：

- JSSError：每个请求对应的非 20x 响应（http status code 不在 200~299 之间）都会转换成一个 JSSError 异常对象并抛出；

PHP SDK 定义了两个辅助的 HTTP 对象

- JSSRequest：几乎封装了所有的 HTTP 请求，用户不需要关心。
- JSSResponse：封装了 SDK 中 http 请求的返回信息，包括 response_code,response body,response header 三个基本信息维护。

PHP SDK 定义了一个文件类型辅助对象：

- JSSMIME：你可以调用 JSSMIME::get_type($file_extension)根据文件扩展名获取其对应的类型；

注意：PHP SDK 中所有的 API 调用只有在成功时才返回正确的数据，其他任何错误都

将以异常的形态抛出！

4.1 通用接口

4.1.1 初始化 JDCloudStorage 对象

接口定义：

```
/**
 * constructor
 * @param string $access_key
 * @param string $access_secret
 * @return $this object
 */
```

代码示例：

```
<?php
require_once dirname(__FILE__).'/../sdk/JDCloudStorage.php';
// 实例化JDCloudStorage对象
$storage = new JDCloudStorage(ACCESS_KEY,ACCESS_SECRET);
// do something…
?>
```

4.1.2 JSSError 异常信息

对象定义：

在访问云存储过程中，所有没有能够正常完成服务请求的操作，都会返回 JSSError,该 Exception 是由 Exception 派生而来，JSSError 的对象中， 并会得出以下由存储服务器获得到的错误返回的响应：错误码，错误信息，请求资源，请求 ID。

代码示例：

例如在创建 2 次 Bucket 时候,代码如下

```
<?php
try {
    $storage->put_bucket($bucket_name);
    $storage->put_bucket($bucket_name);
    success("Put bucket({$bucket_name}) success!");
} catch (JSSError $e) {
    echo "HTTP Code : ".$e->getCode()."\n"; //获取http返回错误码
    echo "Error Code : ".$e->getErrorCode()."\n";   //获取错误信息码
    echo "Error Message:".$e->getErrorMessage()."\n";//获取错误信息
    echo "RequestId : ".$e->getRequestId()."\n"; //获取请求ID
```

```
    echo "requestResource : ".$e->getResource()."\n";//获取请求资源
} catch (Exception $e) {
    echo "Error Code : ".$e->getCode()."\n";
    echo "Error Message : ".$e->getMessage()."\n";
}
?>
```

4.2 Service 相关接口

4.2.1 获取所有 Bucket

该接口对应于 API 中的 GET Service,可以通过该接口获得用户的所有 Bucket 信息
接口定义：

```
/**
* Get all buckets,corresponds to "GET Service" in API
* @param void
* @return JSSBucket objects list
* @exception see JSSError
*/
```

代码示例：

```
<?php
$bucketslist = $storage->list_buckets();
foreach($bucketslist as $jss_bucket) {
        print_r("Bucket:" . $jss_bucket->get_name() . "\n");
        print_r("CTime: " . $jss_bucket->get_ctime()    . "\n\n");
    }
?>
```

4.3 Bucket 相关接口

4.3.1 新建 Bucket

该接口对应于京东云存储 API 中的 PUT Bucket 接口，该接口可以创建一个新的 Bucket
接口定义：

```
/**
* Create new bucket,corresponds to "PUT Bucket" in API
* @param string $name    bucket's name to create
* @param string $options bucket's properties,
* is useless now,future it may contains something like bucket's region,
```

```
and so on
* @return JSSResponse on success
* @exception see JSSError
*/
```

代码示例：

```
<?php
    $storage->put_bucket($bucket_name);
?>
```

4.3.2 删除指定 Bucket

该接口对应于京东云存储 API 中的 DELETE Bucket 接口，可通过该接口删除指定的 Bucket，注意：必须确保要删除的 Bucket 中没有任何数据。

接口定义：

```
/**
* Delete specified bucket,corresponds to "Delete Bucket" in API
* @param string $name    bucket's name to delete
* @return JSSResponse on success
* @exception throw JSSError when bucket is not empty or response invalid
*/
```

代码示例：

```
<?php
  ..
 $jss_response = $storage->delete_bucket($bucket_name);
  ...
?>
```

4.3.3 列出指定 Bucket 下 Objects

该接口对应于京东云存储 API 的 Get Bucket 接口，通过该接口可以获得指定 Bucket 中的 Object 信息列表，请求时可以通过一些查询条件来限制返回的结果。

接口定义：

```
/**
* List all objects of specified bucket,corresponds to "GET Bucket" in API
* @param string $bucket    bucket's name
* @param array $options,search key for this list operations, example:
*     $options = array(
*         "marker" => '',
*         "max-keys" => 100,
*         "prefix" => 'a',
*         "delimiter" => '/',
*         );
```

```
*          can be:
                    @option integer $maxkeys    max response objects number of per-request
                    @option string $marker    response objects offset
                    @option string $delimiter    response objects name filter
                    @option string $prefix    response objects name filter

* @return JSSEntity object
* @exception see JSSError
*/
```

代码示例：

```
<?php
$options = array(
                "marker" => '',
                "maxKeys" => 100,
                "prefix" => 'a',
                "delimiter" => '',
                );
$jssentity = $storage->list_objects($bucket,$options);
$objects = $jssentity->get_object();
foreach($objects as $object) {
            print_r($object->get_key()."\n");
            //print_r($object->get_size()."\n");
            //print_r($object->get_etag()."\n");
            //print_r($object->get_last_modified()."\n");
    }
?>
```

4.4 Object 相关接口

4.4.1 新建 Object

该接口对应于京东云存储 API 中的 PUT OBJECT 接口，该接口用来上传一个新的 Object 到指定的 Bucket 中，数据的最大长度限制为 5GB。

接口定义：

```
/**
* Put object to storage(corresponds to "PUT Object" in API)
* @param string $bucketname,bucket's name
* @param string $objectname    object's name
* @param string $source    local file path(/path/to/filename.ext) or stream
* @param array $request headers,can be empty
* @return JSSResponse on success
* @exception see JSSError
```

```
*/
```

代码示例：

```
<?php
// put_object()方法可以接收一个文件路径
$object_name = 'test.jpg';
$local_file = dirname(__FILE__) . '/logo.png';
$storage->put_object($bucket_name,$key, $local_file);
// put_object()方法可以接收一个 stream 对象作为输入
$object_name = 'test2.jpg';
$local_stream = fopen(dirname(__FILE__) . '/logo.png', 'rb');
$storage->put_object($bucket_name,$object_name, $local_stream);
?>
```

注意：如果 Bucket 下已存在$name 对象，此操作将会失败。
注意：如果新建 Object 大小太大，此接口调用可能会由于超时意外退出，请正确设置脚本超时时间（set_time_limit()）。

4.4.2 获取（下载）Object

该接口对应京东云存储 API 的 GET OBJECT 接口，可通过该接口获取指定 Object 内容
接口定义：

```
/**
* Get object from storage(corresponds to "GET Object" in API)
* @param string $bucket,bucket's name
* @param string $key    object's name
* @param string $target    write to local file path(/path/to/filename.ext) or stream
* @param boolean $auto_close    if auto close the $target passed when it is a stream?
* @param array,$other_headers ,a key-value array,indicate the information such as
Range,If-Modified-Since,and so on.
* @return JSSResponse on success,false when $target is null;
* @exception see JSSError
*/
```

代码示例：

```
<?php
//可传入如 Range 等其他该请求可用的 request header
$other_headers = array();
$storage->get_object($bucket_name,$key,$local_file,$other_headers);
?>
```

注意：如果$localfile 已存在，此操作将会覆盖本地文件。
注意：如果 Object 大小太大，此接口调用可能会由于超时意外退出，请正确设置脚本超时时间（set_time_limit()）。

4.4.3 获取 Object 资源链接 URI

该接口用于获取 Object 的资源链接 URI，可通过设置参数 expire 设定链接的过期时间。
接口定义：

```
/**
 * get object resource from storage
 * @param string $bucket,bucket's name
 * @param string $key    object's name
 * @param integer $expire    expire of resource
 * @return resource on success
 */
```

代码示例：

```
<?php
$expire = 10*60; //十分钟后失效
$url = $storage->get_object_resource($bucket,$key,$expire);
?>
```

4.4.4 获取 Object 的 meta 信息

该接口对应于京东云存储 API 的 HEAD Object 接口，通过该接口可以获取指定 Object 的元数据信息。
接口定义：

```
/**
* Get object's metas(corresponds to "HEAD Object" in API)
* @param string $bucket,bucket's name
* @param string $key,object's name
* @return JSSResponse when success
* @exception see JSSError
*/
```

代码示例：

```
<?php
$jss_response = $storage->head_object($bucket_name,$key);
print_r("Meta of {$key} is ");
print_r($jss_response->get_headers());
?>
```

4.4.5 删除 Object

该接口对应于京东云存储 API 的 DELETE Object 接口，用于删除指定的 Object
接口定义：

```
/**
* Delete object from storage (corresponds to "Delete Object" in API)
* @param string $bucket,bucket's name
* @param string $key,object's name
* @return JSSResponse on success
* @exception see JSSError
*/
```

代码示例：

```
<?php
$storage->delete_object($bucket,$key);
?>
```

4.5 Multipart Upload 相关接口

4.5.1. Put Object with Multipart Upload

该接口对应于京东云存储的 multipart 相关接口，用于使用 Multipart Upload 上传一个大文件。
接口定义：

```
/**
        * Put object to storage with multipart upload
        * @param string $bucketname,bucket's name
        * @param string $objectname    object's name
        * @param string $source    local file path(/path/to/filename.ext) or stream
        * @param array $options headers,can be empty
        * @return JSSResponse on success
        * @exception see JSSError
        */
```

代码示例：

```
$multipartUpload =
                        $storage->init_multipart_upload($bucket,$object_key);
echo "Bucket:".$multipartUpload->get_bucket()."\n";
echo "object key:".$multipartUpload->get_key()."\n";
echo "uploadId:".$multipartUpload->get_uploadid()."\n";
```

4.5.2. Init Multipart upload

该接口对应于京东云存储的 init multipart upload 接口，用于初始化一个分块上传。
接口定义：

```
/**
        * Create new bucket,corresponds to "PUT Bucket" in API
        * @param string $name    bucket's name to create
```

```
        * @param string $options bucket's properties,
        * is useless now,future it may contains something like bucket's region,and so on
        * @param array,request_headers,request header needed in init mulitpart upload,can be
empty
        * @return MultipartUpload on success
        * @exception see JSSError
        */
```

代码示例：

```
            $jss_response = $storage->put_mpu_object($bucket,$object_key,$filePath);
            echo "response code:".$jss_response->get_code()."\n";
            echo "response body:".$jss_response->get_body()."\n";
```

4.5.3. Upload Part

该接口对应于京东云存储的 Upload Part 接口，用于向一个指定 uploadID 的 MultipartUpload 上传一个分块。

接口定义：

```
/**
        * Upload part to storage
        * @param string $bucketname bucket's name
        * @param string $key    object's name
        * @param string $uploadid multipart upload's id
        * @param int $partnumber of this part
        * @param string $source    local file path(/path/to/filename.ext) or stream
        * @param array, $request_headers,can be empty,request headers setted by user
        * @return etag on success
        * @exception throw exception when failed
        */
```

代码示例：

```
$etag = $storage->upload_part($bucket, $key, $uploadid, $part_number, $file_path);
//do something..
```

4.5.4. List Parts

该接口对应于京东云存储的 List Part 接口，用于获取一个指定 uploadID 的 MultipartUpload 所有已上传的分块列表。

接口定义：

```
      /**
        * list parts of multipart upload
        * @param string,$bucket,bucket's name
```

```
        * @param string,$key,object's name
        * @param string $uploadId,the uploadid of the multipart upload
        * @param array,$options,key=>velue,the key can be:
        *                  max-parts,int,the maximum number of parts returned in the response
body
        *                  part-number-marker,the part to start with
        *                  ...
        * @return ListPartsEntity on success
        * @exception,throw exception when failed
        */
```

代码示例：

```
    $options = array(
                "max-parts"=> 100,
                "part-number-marker" => 0
    );
   $listPartEntity = $storage->list_parts($bucket, $key, $uploadId,$options);
   echo "Bucket:".$listPartEntity->get_bucket()."\n";
   echo "ObjectKey:".$listPartEntity->get_objectKey()."\n";
   echo "UploadId:".$listPartEntity->get_uploadId()."\n";
   $parts_list = $listPartEntity->get_part_list();
   foreach($parts_list as $one_part) {
                echo ".....\n";
                echo "PartNumber:".$one_part->get_part_numer()."\n";
                echo "Etag:".$one_part->get_etag()."\n";
                echo "Last modified:".$one_part->get_last_modified()."\n";
                echo ".....\n";
  }
     //do something..
```

4.5.5. Abort MulitpartUplod

该接口对应于京东云存储的 Abort MultipartUpload 接口，用于删除一个指定 uploadId 所在的 MultipartUpload.

接口定义：

```
        /**
         * Abort multipart upload
         * @param string $bucket,bucket's name
         * @param string $key,object's name
         * @param string $uploadId, the uploadid of the multipart upload
         * @throws Exception when failed
         * @true on success
         */
```

代码示例：

```
$storage->abortMultipartUpload($bucket,$object_key,$uploadId);
//do something..
```

4.5.6. Complete MulitpartUplod

该接口对应于京东云存储的 Complete MultipartUpload 接口，用于完成一个指定 uploadId 所在的 MultipartUpload.

接口定义：

```
/**
 * complete a multipartupload
 * @param string $bucket
 * @param string $key
 * @param string $uploadid
 * @param string $complete_json,can be empty.if is '',then we will get it by list_parts($uploadid)
 * @throws Exception when failed
 * @return $jss_response
 */
```

代码示例：

```
$jss_response = $storage->complete_multipartupload($bucket,$object_key,$uploadid);
echo "response code:".$jss_response->get_code()."\n";
echo "response body:".$jss_response->get_body()."\n";
//do something..
```

4.5.7. List MulitpartUplod

该接口对应于京东云存储的 List MultipartUpload 接口，用于获取一个指定 Bucket 下所有未完成的 MultipartUpload 对象信息

接口定义：

```
/**
 * List multipart upload (corresponds to "List Multipart Upload" in API)
 * @param string $bucket, your bucketname
 * @param array,$options, pairs of $key=>$value,the key can be:
 *                          key-marker,the key to start with
 *                          upload-id-marker,the uploadid to start with
 *                          max-uploads,the maximum number of keys returned in the response body
 *                          prefix,the prefix parameter to the key of the multipart upload you want to retrive
 *                          delimiter,the param you use to group keys
 * @return MultipartUploadEntity object on success
```

```
     * @exception throw exception when response invalid
     */
```

代码示例：

```
    $options = array(
                                "key-marker"=>'',
                                "upload-id-marker"=>'');

    $multipartUploadEntity = $storage->list_multipart_upload($bucket,$options);
    print_r($multipartUploadEntity->to_array());
    //do something..
```

如果在使用中遇到任何问题，请向 jcloud@jd.com 反馈，我们将及时跟进。

谢谢！