QINIU
七牛云存储

# 七牛云存储文件上传 - 基础上传服务

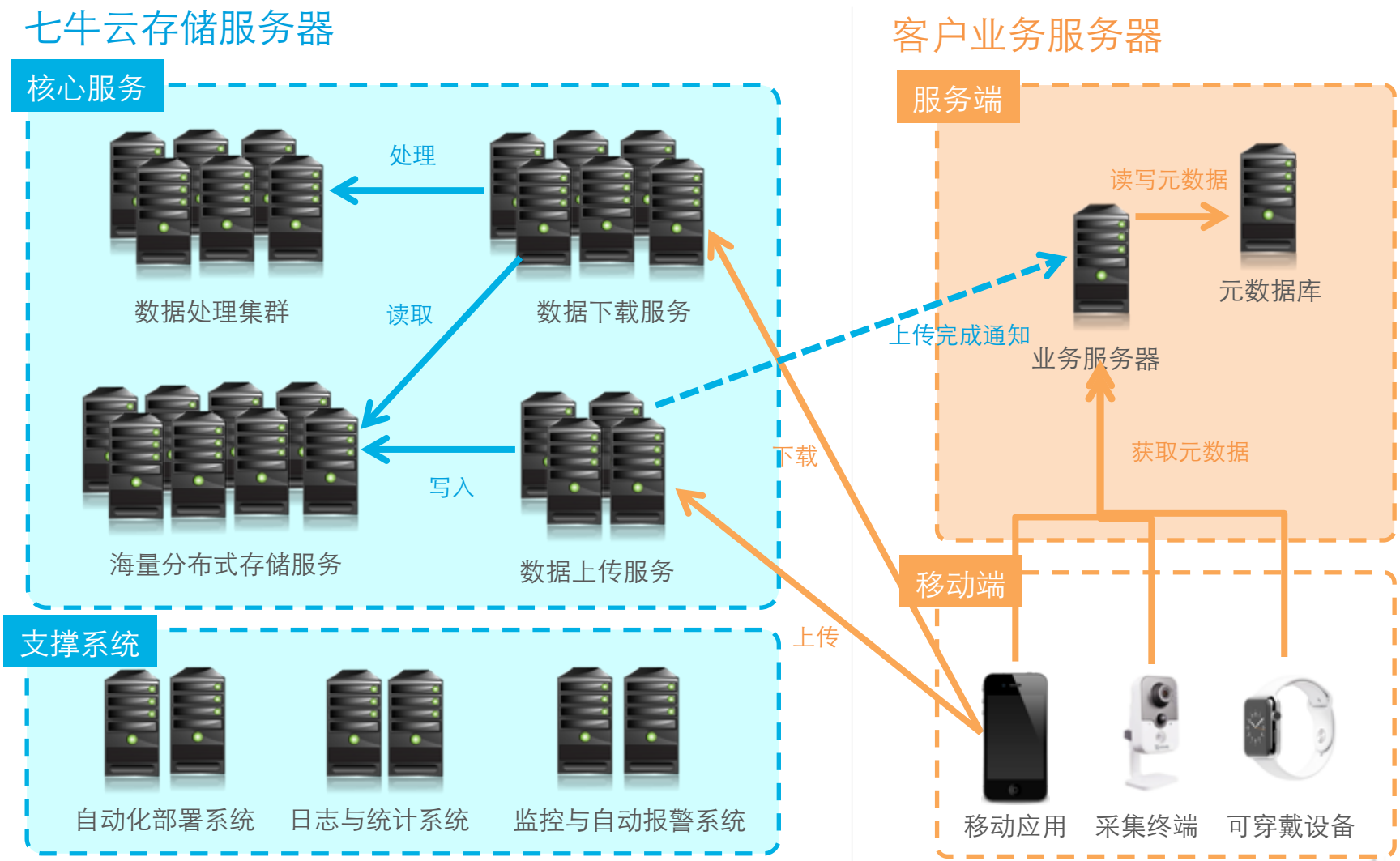
@七牛云存储

QINIU
七牛云存储

# 主要内容

- 七牛云存储上传文件的基础知识
- 七牛云存储变量（魔法变量和扩展变量）
- 七牛云存储命名上传文件的方式
- 简单文件上传 - 基本步骤
- 简单文件上传 - 不指定上传文件key
- 简单文件上传 - 指定上传文件key
- 简单文件上传 - 使用SaveKey作为文件名
- 简单文件上传 - 使用变量作为SaveKey
- 简单文件上传 - 使用ReturnBody自定义返回内容
- 简单文件上传 - 文件同名覆盖上传
- 简单文件上传 - 限制文件的上传大小
- 简单文件上传 - 限制文件的上传类型
- 简单文件上传 - 自动检测文件类型
- 简单文件上传 - 使用EndUser字段标识文件属主
- 简单文件上传 - 带CRC32校验码的文件上传
- IE系列浏览器修复回复的JSON字符串被当作文件下载问题

# 七牛云存储上传文件的基本知识



# 七牛云存储上传文件的基本知识

# 七牛云存储上传文件的基本知识

七牛云存储的文件上传采用标准的multipart/form-data表单来进行文件上传。

可以在表单的参数中指定文件要保存的名字（文件key），多个扩展变量，文件二进制内容，文件内容CRC32校验码，accept参数（用于修复IE的Bug）以及最重要的用于七牛云存储服务器进行访问权限控制和PutPolicy参数解析的上传token字符串。
http://developer.qiniu.com/docs/v6/api/overview/up/form-upload.html

最简单的上传表单可以只有token和要上传的文件内容file这两个参数。

上传文件表单中的上传token需要从业务服务器获取，业务服务器指定客户端文件上传所采用的PutPolicy，并结合AccessKey和SecretKey来生成客户端文件上传所需要的token。

上传token的第三部分是PutPolicy的JSON格式字符串经过URL安全的Base64编码后所得，基本上是PutPolicy的明文。

# 七牛云存储上传文件的基本知识

注意表单参数和PutPolicy参数的区别。

表单参数是HTTP POST请求的参数。PutPolicy参数是上传策略PutPolicy里面的参数，二者不可混淆。简单来讲，表单参数就是HTTP标准参数，而PutPolicy参数则是指定了这个表单和七牛云存储之间关系的参数。PutPolicy包含在上传的token里面，而上传token又是表单参数中不可或缺的一个。

无论是表单参数还是PutPolicy参数都可以根据自己实际开发过程中的需求进行合理的组合，也就是说每一个文件上传的表单中所涉及到的表单参数和PutPolicy参数都是七牛云存储所支持的表单参数和PutPolicy参数集合的一个子集。

七牛云存储文件上传之后的回复永远都是JSON格式的字符串，回复的内容根据实际上传时所设置的参数不同而不同，但是上传失败时返回的回复格式都是一个{"error": "real upload error message"}这样的字符串。

# 七牛云存储上传文件的基本知识

标准HTML文件上传表单

```
<form method="post" action="http://upload.qiniu.com/"
 enctype="multipart/form-data">

  <input name="token" type="hidden" value="<upload_token>"/>
  <input name="file" type="file" />

  <input name="key" type="hidden" value="<file_key>"/>

  <input name="x:<ex_param1>" type="hidden" value="<ex_param1>"/>
  <input name="x:<ex_param2>" type="hidden" value="<ex_param2>"/>

  <input name="crc32" type="hidden" />
  <input name="accept" type="hidden" />

  <input type="submit"  value="Upload"/>
</form>
```

# 七牛云存储上传文件的基本知识

| 参数名 | 类型 | 必填 | 描述 |
| --- | --- | --- | --- |
| token | string | 是 | 文件上传凭证，由业务服务器颁发给客户端。必须是正确的上传凭证，否则返回401错误。 |
| file | file | 是 | 文件本身 |
| key | string | 否 | 指定七牛保存该文件时使用的名字，必须是UTF8编码。在使用覆盖上传的时候，必须指定该参数。 |
| x:extra_param | string | 否 | 扩展参数，由用户自己提供，必须以x:开头，可以指定的个数不限。 |
| crc32 | string | 否 | 文件内容的crc32校验码 |
| accept | string | 否 | 指定为text/plain时用于修正低版本IE的Bug。否则七牛返回application/json格式。 |

# 七牛云存储变量(魔法变量和扩展变量)

1. 你想在上传文件后从七牛服务器获取更多关于上传文件的信息吗？
2. 你想在上传文件的表单里面指定更多的自定义信息吗？

对于第1种需求，七牛云存储提供了魔法变量功能。
http://developer.qiniu.com/docs/v6/api/overview/up/response/vars.html#magicvar

对于第2种需求，七牛云存储提供了自定义变量功能。
http://developer.qiniu.com/docs/v6/api/overview/up/response/vars.html#xvar

魔法变量是由七牛云存储提供的，你只需要指定需要的魔法变量的名字即可，七牛云存储服务器会自动帮你填充魔法变量的值。

自定义变量又叫做扩展变量，你可以自己指定任意多个自定义变量，并指定自定义变量的值，七牛服务器会原封不动地返回这些参数。

# 七牛云存储命名上传文件的方式

文件上传之后，七牛云存储对文件的命名遵循以下的规则：

1. 文件上传的POST请求中，如果指定了参数key，那么以key来命名文件。

2. 文件上传的POST请求中，没有指定参数key，但是上传策略PutPolicy里面指定了saveKey参数，那么以saveKey命名文件。

3. 文件上传的POST请求中，如果没有指定参数key，而且在上传策略PutPolicy里面也没有指定saveKey参数，那么七牛云存储服务器会使用根据文件内容计算得来的hash值作为文件的名字。

备注：

上面的命名方式是按顺序检测的，一旦确认了文件的名字，就不会再检测后面的规则。比如同时指定了表单参数key和上传策略参数saveKey，那么也是以key来命名上传的文件。

七牛云存储服务器会对每个上传的文件都会计算它的hash值，所以上传的文件一定是有名字的。

# 简单文件上传 - 基本步骤

上传文件基本步骤（不含七牛云存储服务器和业务服务器交互的部分）：

1. 业务服务器颁发上传token给文件上传端
2. 文件上传端构建文件上传表单，并填充请求参数
3. 文件上传端向七牛云存储服务器发送HTTP POST请求
4. 文件上传端解析七牛云存储服务器返回的回复信息

备注：最简单的上传请求必须包含上传token和所要上传的文件内容。

# 简单文件上传 - 不指定上传文件key

本例是最简单的文件上传示例：

步骤：
1. 业务服务器生成上传token
2. 创建包含上传token和文件内容的表单
3. 向七牛服务器发送Post请求

```
//业务服务器生成上传token
require_once ("../../lib/qiniu/rs.php");
Qiniu_SetKeys($Qiniu_AccessKey, $Qiniu_SecretKey);
$putPolicy = new Qiniu_RS_PutPolicy($Qiniu_Public_Bucket);
$token = $putPolicy -> Token(null);
//文件上传端表单
<form method="post" action="http://upload.qiniu.com"
enctype="multipart/form-data”  >
      <input name="token" type="text" value="<?php echo $token; ?>"
      <input name="file"     type="file"/>
      <input type="submit" value="Upload" / >
</form>
```

# 简单文件上传 - 指定上传文件key

本例是在刚刚的例子基础上，加上了请求参数key，一旦设置了参数key，那么七牛会使用这个key来给上传的文件命名，如果这个key为空字符串，那么文件命名就是空字符串：

步骤：

1. 业务服务器生成上传token
2. 创建包含上传token，文件名key和文件内容的表单
3. 向七牛服务器发送Post请求

```
//业务服务器生成token的方式不变
//文件上传端表单
<form method="post" action="http://upload.qiniu.com"
enctype="multipart/form-data”  >
      <input name="key" type="text" value=""/>
      <input name="token" type="text" value="<?php echo $token; ?>”
      <input name="file"    type="file"/>
      <input type="submit" value="Upload" / >
</form>
```

# 简单文件上传 - 使用SaveKey作为文件名

本例演示在表单中不指定文件key的情况下，如何使用上传策略PutPolicy中的参数SaveKey来作为文件的名字：

步骤：

1. 业务服务器使用指定了SaveKey的PutPolicy生成上传token
2. 创建包含上传token和文件内容的表单
3. 向七牛服务器发送Post请求

```
//业务服务器生成token的方式不变
require_once ("../../lib/qiniu/rs.php");
Qiniu_SetKeys($Qiniu_AccessKey, $Qiniu_SecretKey);
$putPolicy = new Qiniu_RS_PutPolicy($Qiniu_Public_Bucket);
$putPolicy -> SaveKey = "qiniu_cloud_storage_" . time();
$token = $putPolicy -> Token(null);
```

//文件上传端表单
文件上传表单中不指定key参数即可。

# 简单文件上传 - 使用变量作为SaveKey

上一个例子中，我们演示了如何使用SaveKey作为文件的名字，这里我们还可以看一下，如何使用变量来组成文件的名字。变量当然包括魔法变量何扩展变量了。

步骤：

1. 使用魔法变量和扩展变量组合成一个新的字符串作为SaveKey的值
2. 业务服务器使用指定了SaveKey的PutPolicy生成上传token
3. 创建包含上传token，扩展参数和文件内容的表单
4. 向七牛服务器发送Post请求

```
require_once ("../../lib/qiniu/rs.php");
Qiniu_SetKeys($Qiniu_AccessKey, $Qiniu_SecretKey);
$putPolicy = new Qiniu_RS_PutPolicy($Qiniu_Public_Bucket);
$putPolicy -> SaveKey = "$(x:saveKeyEx)$(fname)";
$token = $putPolicy -> Token(null);
```

上面的SaveKey由魔法变量$(fname)和扩展变量$(x:saveKeyEx)组成。

# 简单文件上传 - 使用变量作为SaveKey

```
<form method="post" action="http://upload.qiniu.com"
enctype="multipart/form-data" >
    <input name="token" type="text" value="<?php echo $token; ?>"
    <input name="x:saveKeyEx" type="text"/>
    <input name="file" type="file"/>
    <input type="submit" value="Upload"/>
</form>
```

魔法变量是由七牛云存储默认提供的，自定义变量或者称为扩展变量是由用户自定义表单中提交的，比如上面的x:saveKeyEx，这里扩展变量必须以x:开头，后面的部分命名方式遵循一般变量命名即可。这个x:就告诉了七牛云存储的服务器，这个是扩展变量，否则的话，表单中任何多余的参数都会被七牛云存储服务器忽略。

魔法变量的可用范围，请参考：
http://developer.qiniu.com/docs/v6/api/overview/up/response/vars.html#magicvar

# 简单文件上传 - 使用ReturnBody 自定义返回内容

上面的例子中，我们看到了七牛云存储服务器返回给上传端的信息都是一个只有文件hash和key的JSON字符串。那如何才能获取更多的关于文件的信息呢？
这个就需要设置上传策略PutPolicy里面的参数ReturnBody了。在ReturnBody里面可以使用七牛云存储支持的魔法变量和自定义变量来组成需要返回的内容。组织的格式一般为JSON数据格式或者URL参数格式。

URL参数格式：

```
fname=$(fname)&fszie=$(fsize)&bucket=$(bucket)&exParam1=$(x:exParam1)
```

JSON数据格式：

```
{
    "fname" => "$(fname)",
    "fsize" => "$(fsize)",
    "bucket" => "$(bucket)",
    "exParam1" => "$(x:exParam1)",
}
```

# 简单文件上传 - 文件同名覆盖上传

默认情况下，如果以指定key或者saveKey的方式上传一个和已经存在于空间中的文件名字相同而且内容相同的文件的时候，七牛服务器会根据新上传的文件内容的hash推断出该文件已经存在，不会覆写已有的文件，如果上传的文件被指定的名字和空间中已有文件的名字相同，但是内容不同时，会返回如下错误信息 {“error”：“file exists”}。

简而言之，七牛云存储默认不会轻易地让你覆盖已有的文件的。

那么，如果真的有这种文件内容不同，但是需要指定相同名字的覆盖上传需求怎么办呢？

很简单，设置PutPolicy里面的scope和insertOnly参数，并且在POST请求里面指定需要覆盖的文件的key。在覆盖上传里面，scope的值需要设置为bucket:key这样的方式，这个key和POST请求里面的key值相同，另外必须保证insertOnly为0，保证了这些，文件就可以覆盖上传了。

# 简单文件上传 - 文件同名覆盖上传

覆盖上传文件的步骤：

1. 按照bucket:key的方式设置PutPolicy里面的scope参数，另外设置insertOnly参数为0。
2. 构建包含指定要覆盖的文件的key，文件内容和上传token的表单
3. 向七牛云存储服务器提交HTTP POST请求
4. 解析七牛云存储服务器返回的回复。

```
//生成上传token
Qiniu_SetKeys($Qiniu_AccessKey, $Qiniu_SecretKey);
$putPolicy = new Qiniu_RS_PutPolicy($Qiniu_Public_Bucket);
$putPolicy -> Scope = $Qiniu_Public_Bucket . ":" . $keyToOverwrite;
$putPolicy -> InsertOnly = 0;
$token = $putPolicy -> Token(null);
```

# 简单文件上传 - 文件同名覆盖上传

//需要的表单格式

```
<form method="post" action="http://upload.qiniu.com"
enctype="multipart/form-data">
      <input name="key" value="<?php echo $keyToOverwrite; ?>" />
      <input name="token" type="text" value="<?php echo $token; ?>"/>
      <input name="file" type="file"/>
      <input type="submit" value="Upload"/>
</form>
```

备注：

1. 上传策略里面的scope必须按照bucket:key的格式指定，表示保存的文件要覆盖bucket所代表空间里面名字为key的文件。
2. 上传策略里面的insertOnly必须为0，否则即使scope如第1步那样指定，也不能覆盖已有的文件。
3. 上传策略里面的key和请求表单里面的参数key必须值相同才能覆盖。

# 简单文件上传 - 限制文件的上传大小


如果你希望用户上传的文件有大小限制，就像我们平时开发web应用时遇到的情况一样，你可以在上传策略PutPolicy里面指定上传文件的大小限制。

在PutPolicy里面，你可以指定fSizeLimit参数来限定上传的文件不能超过这个大小，这个参数是一个整数，单位是字节。比如限制上传的文件不能超过1KB，那么fSizeLimit就设置为1024。

上传的文件超过fSizeLimit限制的情况下，会被判定为上传失败，并且返回413状态码。

默认情况下，七牛不限定上传文件的大小。只要你敢传，七牛就有空间存储。这个得益于七牛独特的分布式存储架构体系。

七牛的文件存储并不是以文件的原始状态存在，而是将文件切割为块，每个块有固定的大小，这样一来无论多大的文件都可以存储。

# 简单文件上传 - 限制文件的上传类型

如果你希望用户只能上传图片，而不是音频或者视频，那么这个时候你可以设置上传策略PutPolicy的参数mimeLimit，用来限制用户可以上传的文件的类型。

一旦你设置了这个参数，七牛服务器会侦测上传文件的类型，并且来和你设置的规则比对，如果匹配就存储，否则返回{“error”：“limited mimeType: this file type is forbidden to upload”}。

这个mimeLimit参数的用法如下(不可以同时指定允许和不允许的文件类型)：
指定允许上面的文件类型：

1. 使用通配符，比如image/*表示允许上传所有图片类型的文件。
2. 分别指定允许上传的文件类型，用分号(;)隔开，比如image/jpeg;image/png表示只允许上传jpeg和png格式的图片。

指定不允许上传的文件类型：

1. 使用通配符，比如!image/*表示不允许所有图片类型的数据。
2. 分别指定不允许上传的文件类型，用分号(;)隔开，比如!application/json;text/plain表示不允许上传json格式数据和纯文本。

# 简单文件上传 - 自动检测文件类型

上传策略里面有一个参数叫做detectMime，可以用来开启七牛服务器自动检测上传文件的MimeType，这个参数默认为0。当被设置为非0值时，七牛服务器会忽略上传端指定的文件MimeType而是遵循下面的规则自动侦测文件的MimeType。当然如果上传端也没有指定文件的MimeType的时候，也会遵循下面的规则来侦测文件的MimeType。

侦测规则：

1. 检查文件的扩展名（原始文件名）
2. 检查key的扩展名
3. 侦测内容

如果上面的步骤，都无法判断出文件的MimeType，那么默认使用application/octect-stream来作为文件的MimeType。

备注：上面的侦测动作是在七牛云存储服务器进行的。一般你不需要指定上传文件的MimeType，七牛服务器会自动判断，如果需要指定也是以HTTP Multipart Data的Content-Type来指定的，这个在SDK中已经实现了。你只需要传入相应参数即可。

# 简单文件上传 - 使用EndUser字段标志文件属主

上传策略PutPolicy里面有一个字段叫做endUser，这个字段一般来讲没有什么太大的用处，因为七牛支持扩展参数，如果有什么额外的自定义需求完全可以使用扩展参数来解决，但是为什么还存在endUser这个字段呢？

一般来讲，这个字段用在App开发里面，可以用来表示这个图片或者视频是谁发的，用这个endUser值来打个水印什么的。

如果你在文件上传的时候指定了这个PutPolicy参数，你可以通过魔法变量（endUser）来获取到它的值。

```
require_once ("../../lib/qiniu/rs.php");
Qiniu_SetKeys($Qiniu_AccessKey, $Qiniu_SecretKey);
$putPolicy = new Qiniu_RS_PutPolicy($Qiniu_Public_Bucket);
$putPolicy->EndUser="jemygraw";
$putPolicy->ReturnBody="{\"endUser\": $(endUser)}";
$token = $putPolicy -> Token(null);
```

# 简单文件上传 - 带CRC32校验码的文件上传

如果你对你上传的文件的完整性十分关心或者你的用户文件上传环境太恶劣的情况下，你可以在HTTP的POST请求中添加一个请求参数crc32，这个参数的值就是你本地计算这个文件内容的crc32的值，七牛会根据你提供的crc32的值来和七牛服务器对上传的文件重新计算crc32得到的值做比对，如果符合，则返回上传文件成功，否则判定为失败并返回错误{"error":"invalid crc32"}。

crc32是一个HTTP上传POST请求参数，它的值是一个10进制的整数。

其实还有很多种方式来保证上传的文件一个字节不少，比如七牛的hash算法。你也可以根据七牛提供的hash算法实现来在上传端对上传文件的内容计算hash，然后和七牛服务器返回的回复中的文件hash做比对，你也可以知道文件是否一个字节不少地传上去了，而且没有被篡改。

总之，确认文件上传完整性可以有两种方式：

1. crc32文件内容校验码
2. 七牛文件内容hash码

# IE系列浏览器修复回复的JSON字符串被当作文件下载问题：


我们上面了解到七牛云存储服务器返回的回复都是JSON类型的字符串，所以回复的Content-Type一般都是application/json。但是，这个回复的Content-Type在低版本的IE系列(IE7-IE10)浏览器上面表现出来的行为是弹出文件下载确认框，这样很不利于我们后面对这个回复内容的处理。

这个问题可以在确认是IE浏览器在上传的情况下，给上传的表单添加一个accept参数，该参数的值为text/plain; charset=utf-8即可解决这个问题。

QINIU
七牛云存储

# 成为最专业的云存储服务提供商！