

ALIBABA

阿里云 OSS 图片处理服务(IMG)

API 使用手册

2014/03/11

目录

1	介绍	2
	基本概念	3
	<i>Object</i>	3
	<i>Channel</i> （频道）	3
	<i>Style</i> （样式）	3
	分隔符	4
1.1	规则	5
1.2	关键词说明	6
1.2.1	顺序无关	6
1.2.2	覆盖处理	6
1.2.3	冲突处理	6
1.2.4	长边与短边	6
1.2.5	URL 安全的 <i>Base64</i> 位编码	7
1.3	用户签名验证（AUTHENTICATION）	7
1.3.1	在 <i>Header</i> 里包含签名	7
1.3.2	在 <i>URL</i> 里包含签名	8
1.3.3	如何快速开发 <i>IMG</i> 签名	9
2	接口	12
2.1	生成指定规格的缩略图	12
	访问规则	12
	使用示例	16
2.2	水印请求	18
	访问规则	18
	使用示例	21
2.3	管道	23
	访问规则	23
	使用示例	23

2.4 样式.....	25
访问规则.....	25
使用示例.....	25
3. 图片处理服务错误响应	26
3.1 图片处理服务错误的响应格式	27
3.2 图片处理服务的错误码	27

1 介绍

阿里云 OSS 图片处理服务(Image Service, 简称 IMG) , 是阿里云 OSS 对外提供的海量, 安全, 低成本高可靠的图片处理服务。用户将原始图片上传保存在 OSS 上, 通过简单的 RESTful 接口, 在任何时间、任何地点、任何互联网设备上对图片进行处理。基于 IMG, 用户可以搭建出跟图片相关的服务。

基本概念

Object

在 IMG 中，用户操作图片的基本数据单元是 Object。即 OSS 对应的 Object， 单个 Object (即每张图片)允许的最大大小是 10MB。

Object 命名规范：

- (一)使用 UTF-8 编码。
- (二)长度必须在 1-1023 字节之间。
- (三)不能以 “/” 或者 “\” 字符开头。

Channel （频道）

Channel 是 IMG 上的命名空间，也是计费、权限控制、日志记录等高级功能的管理实体。IMG 名称在整个图片处理服务中具有全局唯一性，且不能修改。一个用户最多可创建 10 个 Channel，但每个 Channel 中存放的 Object 的数量和大小总和没有限制，用户不需要考虑数据的容量扩展。

目前 Channel 跟 OSS 的 Bucket 相对应，即用户只能创建与自己在 OSS 上 Bucket 同名的 Channel。

Channel 命名规范：

- (一)只能包括小写字母，数字，短横线(-)。
- (二)必须以小写字母或者数字开头和结尾。
- (三)长度必须在 3-63 字节之间。

Style （样式）

图片处理服务提供用户将图片的处理操作和参数保存成一个别名，即样式。一系列操作，利用样式功能后，只需要用一个很短的 URL 就能实现相同的效果。

- 一个 **Channel** 下面有多个样式，目前一个 **Channel** 允许最多有 20 个样式。
- 样式适应于 **Channel** 下面的 **object** 图像变化操作。假如在 **ChannelA** 下面有样式，名称为 **abc**，样式内容是 **100w.jpg**（按宽缩略成 100，保存成 jpg 格式）那么 **ChannelA** 下面所有的 **object** 都能使用样式 **abc**，实现缩略成 **100w.jpg** 的效果。
- 样式的作用范围只在一个 **Channel** 下，即 **ChannelA** 不能使用 **Channel B** 的样式。

Style 命名规范：

- (一) 只能包括小写字母，数字，短横线(-)。
- (二) 必须以小写字母或者数字开头和结尾。
- (三) 长度必须在 3-63 字节之间。

注意：样式的分隔符是：**@!**

分隔符

图片处理服务使用通过 **URL** 来访问处理的图片。所以需要分隔符来区分一些关键字段。不要在使用图片文件名称中包含图片处理服务设定的分隔符。不然会导致解析出错的问题。

分隔符名称	分隔符	含义
处理分隔符	@	区分 object 名称跟处理字符串，详见 2.1.1
样式分隔符	@!	区分 object 跟样式内容，详见 2.4.1
管道分隔符		区分多种操作，详见 2.3.1

1.1 规则

图片处理服务使用通过 URL 来访问处理的图片，本文定义访问方式的规范。形式为：`http://userdomain/object@100w_100h_90Q.jpg`。

其中 `userdomain` 为用户开通图片服务绑定的域名，这个域名会关联到一个 Bucket。`object` 为用户所关联 Bucket 上存储的原图片。`@100w_100h_90Q` 为转换字符串，用来转换处理图片的一段参数。通过指定转换字符串，生成并返回另一张转换处理后的图片。

一个典型的转换字符串，如“`@100w_100h_90Q.jpg`”，代表需要一张宽(w)100px、高(h)100px、绝对质量(Q)90%、jpg 格式的图片，转换字符串的组成如图 1 转换字符串。

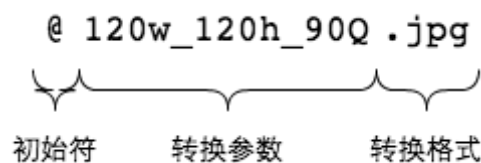


图 1 转换字符串

转换字符串分为 3 部分：初始符、转换参数、转换格式：

- 初始符是一个“@”符号，从此符号开始，后面都为转换字符串。
- 转换参数由一个或多个键值对（以“_”连接）组成，“值”在前“键”在后，“值”为数字类型，“键”为一位字母。
- 转换格式是一种特殊的转换参数，通过指定转换格式，我们对原图处理并返回用户期望的图片文件格式。（支持格式是：jpg， jpeg， webp， png， bmp）。

1.2 关键词说明

1.2.1 顺序无关

转换参数中键值对是循序无关的，即"120w_120h_90Q"和"90Q_120w_120h"都能取到想要的图片，系统会对参数按照本规范以下定义的顺序重新排序后再处理。

（由于参数的顺序不同有时会表达不同的语义，如“100w_100h_2x”表达的是“先缩放到 100*100，再放大 2 倍”，即得到 200*200 的图片；而"2x_100w_100h"按照字面顺序理解是“先放大 2 倍再缩放到 100*100”，即得到 100*100 的图片，为了避免这样的理解误差，同时简化处理方式，IMG 会对参数按照文档中出现的顺序排序后处理。上例中的"2x_100w_100h"会被理解为“100w_100h_2x”，得到 200*200 的图片。）

1.2.2 覆盖处理

如果转换参数中出现多个相同“键”，后面定义的覆盖前面定义。如"120w_120h_240w"等同于“120h_240w”。

1.2.3 冲突处理

见每个参数中关于冲突的说明。

1.2.4 长边与短边

关于“长边”和“短边”的定义需要特别注意，它们表达的是在缩放中相对比例的长或短。“长边”是指原尺寸与目标尺寸的比值大的那条边；“短边”同理。如原图 400 * 200，缩放为 800 * 100，（400/800=0.5，200/100=2，0.5 < 2），所以在这个缩放中 200 那条是长边，400 是短边。

1.2.5 URL 安全的 Base64 位编码

在图片处理服务里会有很多参数需要变成 Base64 位编码。编码的格式是：

- 先将内容编码成 Base64 结果；
- 将结果中的加号“+”替换成中划线“-”；
- 将结果中的斜杠“/”替换成下划线“_”；
- 将结果中尾部的“=”号全部去除；

1.3 用户签名验证（Authentication）

图片处理服务的签名验证与 OSS 签名验证方法使用相同的验证逻辑。其中

CanonicalizedResource 的构成由三部分构成：

- 使用绑定域名对应的图片处理服务频道，这个频道名称与 Bucket 名称相同
- 原 OSS object 文件名称
- 转换字符串。

如：用户绑定的域名 `www.test.com` 对应的频道名字为 `image-demo`，object 名字为 `example.jpg`，转换字符串为：`100w.jpg`

在图片处理服务中，CanonicalizedResource 为

`/image-demo/example.jpg@100.jpg`

注意：上例中的转换字符串可以是简单缩略，文字水印，图片水印、管道和样式（样式的分隔符是@!）

具体 OSS 签名规则可以参考 OSS API 手册。

1.3.1 在 Header 里包含签名

Authorization 字段计算方法如下：

```
"Authorization: OSS " + Access Key Id + ":" + Signature  
Signature = base64(hmac-sha1(VERB + "\n"  
    + CONTENT-MD5 + "\n"  
    + CONTENT-TYPE + "\n"
```

```
+ DATE + "\n"
+ CanonicalizedOSSHeaders
+ CanonicalizedResource))
```

■ CONTENT-MD5 表示请求内容数据的 MD5 值,对于图片处理服务这里为空字符串

■ CONTENT-TYPE 表示请求内容的类型,对于图片处理服务这里为空字符串

■ DATE 表示此次操作的时间,且必须为 HTTP1.1 中支持的 GMT 格式

■ CanonicalizedOSSHeaders 表示 http 中的 object meta 组合,对于图片处理服务这里为空字符串

■ CanonicalizedResource 表示用户想要访问的 OSS 资源,在图片处理服务中这项的组成,格式为/channelname/object@处理参数

其中,DATE 和 CanonicalizedResource 不能为空;如果请求中的 DATE 时间和 OSS 服务器的时间差正负 15 分钟以上,OSS 图片处理服务将拒绝该服务,并返回 HTTP 403 错误。

1.3.2 在 URL 里包含签名

Signature 字段计算方法如下:

```
Signature = base64(hmac-sha1(VERB + "\n"
+ CONTENT-MD5 + "\n"
+ CONTENT-TYPE + "\n"
+ Expires + "\n"
+ CanonicalizedOSSHeaders
+ CanonicalizedResource))
```

■ CONTENT-MD5 表示请求内容数据的 MD5 值,对于图片处理服务这里为空字符串

■ CONTENT-TYPE 表示请求内容的类型,对于图片处理服务这里为空字符串

■ Expires 授权给用户 URL 签名过期时间

■ CanonicalizedOSSHeaders 表示 http 中的 object meta 组合,对于

图片处理服务这里为空字符串

■ CanonicalizedResource 表示用户想要访问的 OSS 资源，在图片处理服务中这项的组成，格式为/channelname/object@处理参数

其中，Expires 和 CanonicalizedResource 不能为空；

在 URL 里包含签名的示例链接如下：

```
http://www.test.com/example.jpg%40100w.jpg?OSSAccessKeyId=j4y55h3z8
8ihxxhIr9nhjjs&Expires=1392949804&Signature=IDBJ09e8Ow4GaPRM1yIf7
pIH/CI%3D
```

在 URL 签名，必须在参数后包含，OSSAccessKeyId， Expires， Signature 这三项构造方法可以参考 OSS API 文档。

1.3.3 如何快速开发 IMG 签名

本节主要介绍如何使用 OSS 的 Python SDK 去获取 private bucket 的图片处理服务，因为图片处理服务都是 GET 操作，所以使用 OSS Python SDK 主要以 Get Object 为主。

以 Python sdk 为例（Python sdk 下载可以在官网查找）

在 Python sdk 里有一个 get_object 操作，传入的参数一般是 bucket，object。

在 OSS：

获取 bucket: image-demo， object: example.jpg 的方法是（以 Python sdk 为例）

```
bucket = 'image-demo'
```

```
object = 'example.jpg'
```

```
self.oss.get_object(bucket, object)
```

在 IMG：

1.简单缩略：

例如：获取 bucket: image-demo， object: example.jpg ，

转换字符是： 100w_100h.jpg

代码：

```
bucket = 'image-demo'
```

```
object = 'example.jpg'
query = '100w_100h.jpg'
object = object + '@' + query
self.oss.get_object(bucket, object)
```

2. 图片水印:

例如: 获取 bucket: image-demo, object: example.jpg ,
转换字符是: watermark=1&object=cGFuZGEucG5n&t=90&p=5
代码:

```
bucket = 'image-demo'
object = 'example.jpg'
query = 'watermark=1&object=cGFuZGEucG5n&t=90&p=5'
object = object + '@' + query
self.oss.get_object(bucket, object)
```

3. 文字水印:

例如: 获取 bucket: image-demo, object: example.jpg
转换字符是: watermark=2&text=SGVsbG8g5Zu-54mH5pyN5YqhIQ
代码:

```
bucket = 'image-demo'
object = 'example.jpg'
query = 'watermark=2&text=SGVsbG8g5Zu-54mH5pyN5YqhIQ '
object = object + '@' + query
self.oss.get_object(bucket, object)
```

4. 样式:

例如: 获取 bucket: image-demo, object: example.jpg
样式名: pipe1
代码:

```
bucket = 'image-demo'
object = 'example.jpg'
style = ' pipe1 '
object = object + '@!' + style
self.oss.get_object(bucket, object)
```

5. 管道

例如: 获取 bucket: image-demo, object: example.jpg
管道操作: 200w.jpg|watermark=1&object=cGFuZGEucG5n&t=90&p=5
代码:

```
bucket = 'image-demo'
object = 'example.jpg'
query = ' 200w.jpg|watermark=1&object=cGFuZGEucG5n&t=90&p=5'
```

```
object = object + '@' + query  
self.oss.get_object(bucket, object)
```

2 接口

2.1 生成指定规格的缩略图

访问规则

<文件 URL>@

<width>w_<height>h_<quality>q_<edge>e_<multiple>x_<cut>c_<immobilize>i_<orient>o.<Format>

例如：userdomain /example.jpg@100w_100h_90q_1e_1x_1o.jpg

参数的取值与详细解析范围见下表。

表 1 缩略图请求参数详解

参数名称	说明	是否必须	取值范围
<width>(w)	<p>指定目标缩略图的宽度。</p> <p>当单独使用时，代表按照宽度等比缩放。如 "120w"，代表原图宽度缩放到 120px，对高度不限制，按照等比例缩放。</p> <p>当与“h”混合使用时，代表长边/短边优先（由参数 e 决定，默认长边优先，详情见 1.5.3）的等比缩放，如原图为 "200 * 400"，使用 "100w_100h" 后，长边（400）缩放到 100px，短边（200）缩放到 50px。同样的原图，使用 "100w_100h_0e" 后，短边（200）缩放到 100px，长边（400）缩放到 200px。</p>	Height/width 必须有一个	1-4096


参数名称	说明	是否必须	取值范围
<height>(h)	<p>指定目标缩略图的高度，单位：像素（px）</p> <p>当单独使用时，代表按照高度等比缩放。如"120h"，代表原图高度缩放到 120px，对宽度不限制，按照等比例缩放。</p> <p>当与“w”混合使用时，代表长边/短边优先（由参数 e 决定，默认长边优先，详情见 1.5.3）的等比缩放。如原图为"200 * 400"，使用"100w_100h"后，长边（400）缩放到 100px，短边（200）缩放到 50px。同样的原图，使用"100w_100h_0e"后，短边（200）缩放到 100px，长边（400）缩放到 200px。</p>	Height/width 必须有一个	1-4096
<quality>(q)	<p>决定 jpg 图片的相对 quality，对原图按照 q% 进行 quality 压缩。如果原图 quality 是 100%，使用"90q"会得到 quality 90% 的图片；如果原图 quality 是 80%，使用“90q”会得到 quality 72% 的图片</p> <p>只能在 jpg 格式的图片上使用，才有相对压缩的概念。如果保存图片是 png，那么相对质量就相当于绝对质量。</p>	否	1-100
<Quality>(Q)	<p>决定 jpg 图片的绝对 quality，把原图 quality 压到 Q%，如果原图 quality 小于指定数字，则不压缩。如果原图 quality 是 100%，使用"90Q"会得到 quality 90% 的图片；如果原图 quality 是 95%，使用“90Q”还会得到 quality</p>	否	1-100

参数名称	说明	是否必须	取值范围
	<p>90%的图片；如果原图 quality 是 80%，使用“90Q”不会压缩，返回 quality 80%的原图。</p> <p>只能在 jpg/png 效果上使用，其他格式无效果。</p> <p>如果一个转换 url 里，即指定了 q 和 Q，按 Q 来处理</p>		
<multiple>(x)	<p>尺寸调整倍数，默认值：1</p> <p>原图按照指定比例缩放，如原图尺寸 200 * 100，使用“3x”，得到尺寸为 600 * 300 的图片。</p> <p>当与 w、h 等缩放参数混合使用时，即对 w 和 h 乘 x 倍，再进行缩放，如原图 400 * 200，使用“50w_50h_2x”（等同于“100w_100h”）后，得到 100 * 50 的图片。</p>	否	必须保证转换后的 height/width 在 (1-4096) 这个范围
<immobilize>(i)	<p>是否固定宽高，默认值是：0（否）</p> <p>此参数与参数 c 配合使用，当使用 i 时会固定图片的宽高，进行非缩放裁剪。</p> <p>使用 i 裁剪时，会以图片中线为中心，裁剪指定 w 和 h 的区域，如原图 200 * 400，使用参数“100w_100h_1c_1i”，会裁剪出图片中心 100 * 100 的区域。</p> <p>冲突：此参数需要与“w”、“h”、“c”同时使用，单独使用无意义。</p> <p>冲突：i 和 e 同时存在，以 i 为准，忽略 e</p>	否	0：否（默认值），1：是

参数名称	说明	是否必须	取值范围
<cut>(c)	<p>是否裁剪，默认值是：0（否）</p> <p>对缩放后超出范围的图片内容进行剪裁，这种情况一般发生在按照短边优先的等比缩放中。</p> <p>缩放会以图片中线为中心，进行上下/左右的裁剪，得到相应的尺寸，如原图 200 * 400，使用参数"100w_100h_1e_1c"，首先按照短边优先缩放一张 100 * 200 的图片，之后按照中线裁剪高度，得到一张 100 * 100 的图片。</p> <p>冲突：此参数需要与“w”、“h”、“e”或“w”、“h”、“i”同时使用，单独使用无意义。</p> <p>“w”、“h”、“e”、“c”表示缩放后裁剪 “w”、“h”、“i”、“c”表示不缩放裁剪 当有 i，e 时存在时，相当于“w”、“h”、“i”、“c”</p>	否	0：否（默认值），1：是
<edge> (e)	<p>缩放优先边，默认值：0：长边（默认值）</p> <p>由于图片缩放过程中，原图尺寸与缩放尺寸不一定是相同比例，需要我们指定以长边还是短边优先进行缩放，如原图 200 * 400（比例 1:2），需要缩放为 100 * 100（比例 1:1）。长边优先时，缩放为 50 * 100；短边优先时，缩放为 100 * 200，若不特别指定，则代表长边优先。</p> <p>由于多数情况下，我们会使用长边优先的缩放，e 的默认值就是 0，一般我们不用显式指</p>	否	0：长边（默认值），1：短边

参数名称	说明	是否必须	取值范围
	<p>定。如“100w_100h_0e”，使用“100w_100h”就好。</p> <p>当使用短边优先的缩放时，由于不同图片长边缩放后的长度不一致，在实际使用中往往需要对长边进行剪裁，这时就需要使用到 c 参数</p> <p>冲突：此参数需要与“w”、“h”同时使用，与单独的“w”或“h”一起使用无意义。</p>		
Orient(o)	<p>有些相机拍的照片会出现旋转，本参数就是根据原图 EXIF 信息自动适应方向。</p> <p>0 表示按原图默认处理</p> <p>1 表示按原图 EXIF 信息自动旋转图片。</p> <p>默认值：</p> <p>0 （按原图默认）</p>	否	0/1
<Format>(.)	<p>默认 jpg</p> <p>指定目标缩略图的输出格式。</p>	否	Jpg/png/webp/bmp/jpeg

使用示例

说明	链接	图片
只针对原图缩略，只指定高度 80，绝对质量为 50，	http://userdomain/example.jpg@80h_50Q_1x.jpg	

只针对原图缩略, 只指定宽度 80, 绝对质量为 50,	http://userdomain/example.jpg@80w_50Q_1x.jpg	
只针对原图缩略, 只指定宽度 80.绝对质量是 50, 放大倍数是 2	http://userdomain/example.jpg@80w_50Q_2x.jpg	
对原图缩略, 指定宽高都是 80, 按长边压缩, 绝对质量是 50	http://userdomain/example.jpg@80w_80h_50Q_1x.jpg http://userdomain/example.jpg@80w_80h_0e_50Q_1x.jpg (两者等同)	
对原图缩略, 指定宽高都是 80, 按短边压缩, 绝对质量是 50	http://userdomain/example.jpg@80w_80h_1e_0c_0i_50Q_1x.jpg	
对原图进行缩略, 按短边缩略, 指定宽高 80, 并进行居中裁剪	http://userdomain/example.jpg@80w_80h_1e_1c_50Q_1x.jpg	
对原图进行缩略, 并按短边缩略, 指定宽高 80, 并进行居中裁剪, 保存成 png	http://userdomain/example.jpg@80w_80h_1e_1c_50Q_1x.png	
不对原图进行缩略, 直接从原图, 剪裁出宽高 120 的区域	http://userdomain/example.jpg@120w_120h_1c_1i_50Q_1x.jpg	

2.2 水印请求

水印的处理分成文字水印与图片水印。

访问规则

图片水印

@watermark=1&object=<encodedobject>&t=<transparency>&x=<distanceX>&y=<distanceY>&p=<position>

文字水印

@watermark=2&text
=<encodefontText>&type=<encodefontType>&color=<encodefontColor>&t=<transparency>&x=<distanceX>&y=<distanceY>&p=<position>

表 2 水印请求参数详解

参数名称	说明	是否必须
<watermark>	<p>参数意义：表示水印类型</p> <p>缩写：无</p> <p><mode>=1: 表示图片水印</p> <p><mode>=2: 表示文字水印</p>	是
<encodedobject>	<p>参数意义：表示图片水印 LOGO 的 object 名字</p> <p>缩写：object</p> <p>注意：必须是 Base64 编码</p> <p>EncodedObject = url_safe_base64_encode(object)</p> <p>base64 位编码后 object</p>	当 mode=1 时，encodedobject 是不可以缺少的

参数名称	说明	是否必须
<fontText>	<p>参数意义：表示文字水印的文字内容</p> <p>缩写： text</p> <p>注意：必须是 Base64 编码</p> <p>EncodefontText = url_safe_base64_encode (fontText) 最大长度为 64 个字符(即支持汉字最多 20 个左右)</p>	当 mod=2 时是必须参数
<fontType>	<p>参数意义：表示文字水印的文字类型</p> <p>缩写： type</p> <p>注意：必须是 Base64 编码</p> <p>EncodeFontType = url_safe_base64_encode (fontType)</p> <p>取值范围：(括号里表示的是中文意思)</p> <p>wqy-zenhei(文泉驿正黑)</p> <p>wqy-microhei(文泉微米黑)</p> <p>fangzhengshusong(方正书宋)</p> <p>fangzhengkaiti(方正楷体)</p> <p>fangzhengheiti(方正黑体)</p> <p>fangzhengfangsong(方正仿宋)</p> <p>droidsansfallback(DroidSansFallback)</p> <p>默认值： wqy-zenhei</p>	当 mod=2 时是必须参数
<fontSize>	<p>参数意义：文字水印文字大小(px)</p> <p>缩写： size</p> <p>取值范围： (0, 1000]</p> <p>默认值： 40</p>	否
<fontColor>	<p>参数意义：文字水印文字的颜色</p>	否

参数名称	说明	是否必须									
	<p>缩写： color</p> <p>注意： 参数必须是 Base64 位编码</p> <p>EncodeFontColor = url_safe_base64_encode(fontColor)</p> <p>参数的构成必须是：# + 六个十六进制数</p> <p>如：#000000 表示黑色。</p> <p>#是表示前缀，000000 每两位构成 RGB 颜色， #FFFFFF 表示的是白色</p> <p>默认值： #000000 黑色 base64 编码后值： IzAwMDAwMA</p>										
<transparency>	<p>参数意义： 透明度</p> <p>缩写： t</p> <p>默认值： 100， 表示 100%（不透明）</p> <p>取值范围：0-100</p>	否									
<position>	<p>参数意义： 位置</p> <p>缩写： p</p> <p>默认值： 9，表示在右下角打水印</p> <p>取值范围： (1-9)</p> <table border="1"> <tr> <td>1 左上</td><td>2 中上</td><td>3 右上</td></tr> <tr> <td>4 左中</td><td>5 中部</td><td>6 右中</td></tr> <tr> <td>7 左下</td><td>8 中下</td><td>9 右下</td></tr> </table>	1 左上	2 中上	3 右上	4 左中	5 中部	6 右中	7 左下	8 中下	9 右下	否
1 左上	2 中上	3 右上									
4 左中	5 中部	6 右中									
7 左下	8 中下	9 右下									
<distancex>	<p>参数意义： 水平边距</p>	否									

参数名称	说明	是否必须
<distancey>	<p>缩写：x</p> <p>默认值：10</p> <p>取值范围：1 - 4096</p> <p>单位：像素（px）</p> <p>参数意义：垂直边距</p> <p>缩写：y</p> <p>默认值：10</p> <p>取值范围：1 - 4096</p> <p>单位：像素（px）</p>	否

使用示例

在 example.jpg 上打水印，水印图片 panda.png 经对 urlsafe_base64_encoding 编码后是 cGFuZGEucG5n 位置是右下角，透明度是 51%，水平边距是 10，垂直边距是 10

<http://userdomain/example.jpg@watermark=1&object=cGFuZGEucG5n&t=51&p=9&x=10&y=10>



在 `example.jpg` 上打文字水印，水印文字内容“你好，图片处理服务！”经过 `url_safe_base64_encode` 编码过后是 `5L2g5aW977yM5Zu-54mH5pyN5Yqh77yB`，字体 `fangzhengheiti` 经过 `urlsafe_base64_encoding` 编码之后是 `ZmFuZ3poZW5naGVpdGk` 字体颜色是黑色，位置是中间，透明度是 72%，水平边距是 10，垂直边距是 10。

`http://userdomain/example.jpg@watermark=2&type=ZmFuZ3poZW5naGVpdGk&size=40&text=SGVsbG8g5Zu-54mH5pyN5YqhIQ&color=lzAwMDAwMA&t=90&p=5`



2.3 管道

访问规则

<文件 URL>@<action1>|<action2>

URL 通过@符号传参来实现即时云处理，如果有多任务可以用管道来实现，执行顺序按管道指定顺序执行，目前只支持二级管道。

管道的分隔符是“|”

上述表示先做对文件 URL 做处理 action1 然后再在上述的基础上做处理 action2，然后输出结果。上述 action1，action2 可以是简单缩略，文字水印，图片水印任意一种。

使用示例

表 3 管道使用示例

说明	链接	图片
将一个原图缩略，然后在缩略图上打上另外一个图片作为水印，如先将图缩略成 150 * 150，再打 logo	http://userdomain/example.jpg@150w_150h_1e_1c_0i_100q_1x.jpg watermark=1&&object=cGFuZGEucG5n&t=51&p=9&x=10&y=10	
将原图缩略成 250 * 250，再从中间裁剪出 150 * 150 的区域	http://userdomain/example.jpg@250w_250h_0e_0c_0i_90q_1x.jpg 150w_150h_0e_1c_1i_90q_1x.jpg	

<p>将原图缩略成 180*180</p> <p>再进行文字水印：</p> <p>水印的内容是:Hello 图片处理服务</p> <p>字体类型：文泉驿正黑(wqy-zenhei)</p> <p>字体颜色：黑色(#000000)</p> <p>字体大小：30</p> <p>水印位置：右下角</p> <p>水平边距：10px</p> <p>垂直边距：10px</p>	<p>http://userdomain/example.jpg@180w_180h_1e_1c_0i_0o_90q_1x.jpg watermark=2&&type=d3F5LXplbmhlaQ&size=25&text=SGVsbG8g5Zu-54mH5pyN5YqhlQ&color=lzAwMDAwMA&t=90&p=9&x=10&y=10</p>	
---	--	---

2.4 样式

因为所有对图片的变换都会加在 URL 后面，会导致 URL 变得冗长，不方便管理与阅读。图片处理服务提供用户将常见的操作保存成一个别名，即样式。一个复杂操作，利用样式功能后，只要用一个很短的 URL 就能实现相同的效果。

一个 Channel 下面有多个样式，样式的作用范围只在一个 Channel 下，目前一个 Channel 允许最多有 20 个样式。

访问规则

<文件 URL>@!StyleName

文件 URL 即是由 Channel+object 组成的 url 地址

1. @!是样式的分隔符，在 URL 带了这个分隔符，图片处理服务会把该分隔符后面的内容当成样式的名称。
2. StyleName 表示的是样式的名字。
3. 创建样式、删除样式和修改样式都在前端控制台实现。
4. 当访问的样式在指定 Channel 不存在时，将返回 NotSuchStyle 错误。

使用示例

假如我们对 image-demo 这个 Channel 创建三个样式分别对应上述[管道使用示例](#)

样式名	样式内容
pipe1	150w_150h_1e_1c_0i_100q_1x.jpg watermark=1&&object=cGFuZGEucG5n&t=51&p=9&x=10&y=10
pipe2	250w_250h_0e_0c_0i_90q_1x.jpg 150w_150h_0e_1c_1i_90q_1x.jpg
pipe3	180w_180h_1e_1c_0i_0o_90q_1x.jpg watermark=2&&type=d3F5LX

	plbmhlaQ&size=25&text=SGVsbG8g5Zu-54mH5pyN5YqhIQ&color=lzAwMDAwMA&t=90&p=9&x=10&y=10
--	--

表 4 管道使用示例

说明	链接	图片
将一个原图缩略，然后在缩略图上打上另外一个图片作为水印，如先将图缩略成 150 * 150，再打 logo	http://userdomain/example.jpg@!pipe1	
将原图缩略成 250 * 250，再从中间裁剪出 150 * 150 的区域	http://userdomain/example.jpg@!pipe2	
将原图缩略成 180*180 再进行文字水印： 水印的内容是:Hello 图片处理服务 字体类型：文泉驿正黑 (wqy-zenhei) 字体颜色：黑色(#000000) 字体大小：30 水印位置：右下角 水平边距：10px 垂直边距：10px	http://userdomain/example.jpg@!pipe3	

3. 图片处理服务错误响应

当用户访问图片处理服务出现错误的时候，图片处理服务会返回给用户相应的错误码和错误信息，以帮助用户定位与处理问题。

3.1 图片处理服务错误的响应格式

```
<Error>
  <Code>BadRequest</Code>
  <Message>Input is not base64 decoding.</Message>
  <RequestId>52B155D2D8BD99A15D0005FF</RequestId>
  <HostId>userdomain</HostId>
</Error>
```

错误响应的消息体例子

错误包含以下元素：

- **Code**: 图片处理服务返回给用户的错误码。
- **Message**: 图片处理服务给出的详细错误信息。
- **RequestId**: 用以标识错误请求的唯一 UUID，在无法解决问题时候，可以使用此错误 ID 发送给图片处理服务的工程师去定位错误的原因。
- **HostId**: 用来标识访问的图片处理服务集群。

3.2 图片处理服务的错误码

错误码	描述	HTTP 状态码
TooManyPipe	管道数目超过限制	400
InvalidArgument	参数错误	400

BadRequest	错误请求	400
MissingArgument	缺少参数	400
ImageTooLarge	图片大小超过限制	400
WatermarkError	水印错误	400
AccessDenied	拒绝访问	403
SignatureDoesNotMatch	签名不匹配	403
NoSuchFile	图片不存在	404
NoSuchStyle	样式不存在	404
NoSuchChannel	频道不存在	404
InternalServerError	服务内部错误	500
NotImplemented	方法未实现	501