

腾讯云无服务器云函数

新手入门

产品文档



腾讯云

【版权声明】

©2013-2017 腾讯云版权所有

本文档著作权归腾讯云单独所有，未经腾讯云事先书面许可，任何主体不得以任何形式复制、修改、抄袭、传播全部或部分本文档内容。

【商标声明】



及其它腾讯云服务相关的商标均为腾讯云计算（北京）有限责任公司及其关联公司所有。本文档涉及的第三方主体的商标，依法由权利人所有。

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

文档声明.....	2
新手入门.....	4
新手入门总览.....	4
示例一：创建 Hello World 函数.....	5
步骤一：创建 Hello World 函数.....	5
步骤二：调用 Hello World 函数.....	7
示例二：创建云函数下载 COS Bucket 中上传的文件	9
步骤一：创建 DownloadImage 函数	9
步骤二：调用 DownloadImage 函数	14

新手入门

新手入门总览

在此章节中，我们为初次使用 SCF 无服务器云函数的用户准备了一些示例，您可以跟随此教程执行以下操作：

- 创建并测试简单的 Hello World 函数；
- 通过设置 COS 的文件上传动作作为触发器，创建并测试一个函数：在某个COS Bucket上有文件上传时，下载该文件到环境的文件系统上。

用户能通过这两个示例完成一些 SCF 控制台操作，包括：

- 使用模版。每个模版均提供了实现某些逻辑的示例代码和示例配置，您只需选择需要的模版即可轻松创建已包含示例代码的云函数。Hello World入门练习使用了

Hello World

模版，下载文件入门练习使用了

COS Put Object

模版。

- 创建和更新云函数的配置信息。
- 手动调用云函数并观察函数运行结果。

阅读完新手入门后，您还可以阅读 云函数的工作原理、构建 SCF 云函数等内容，了解更多关于使用云函数构建生产系统的详细说明。

示例一：创建 Hello World 函数

步骤一：创建 Hello World 函数

1) 登录腾讯云控制台，选择无服务器云函数。

2) 在

广州

地域下点击【创建函数】按钮，进入新建函数页面。

3) 填写函数名称

hello-world

，其他配置项保持默认选项即可。

4) 点击【下一步】按钮，进入函数代码编辑页，默认选择【在线编辑】，并在【模版】中选择

Hello World

模版。此时，执行方法和代码将填入模版的默认值：

- 执行方法显示

index.main_handler

。表示无服务器云函数控制台会将此段代码自动保存为

index.py

文件，并压缩该文件上传至 SCF 平台以创建云函数。

- 函数代码显示以下代码片段：

```
print('Start Hello World function')
def main_handler(event, context):
    print("value1 = " + event['key1'])
    print("value2 = " + event['key2'])
    return event['key1'] #return the value of key "key1"
```

该示例代码将从

event

参数中获取类似以下形式的数据：

```
{
  "key1": "test value 1",
  "key2": "test value 2"
}
```

5) 点击【下一步】，进入触发方式页面。对于此示例代码来说，不需要配置任何触发器，直接点击【完成】按钮。

6) 此时控制台会自动生成代码程序包并上传至 SCF 平台以创建云函数。您可以点击云函数列表页中刚刚创建的

hello-world

函数进入云函数详情页。

步骤二：调用 Hello World 函数

请用户按照以下步骤使用控制台提供的测试模版来调用刚刚创建的Hello World函数。

1) 在列表页点击刚刚创建的Hello World函数进入函数详情页，点击右侧【测试】按钮。

2) 在测试函数弹出框中，从测试模版中选择

Hello World测试代码

，下列数据将出现在窗口中。用户可以任意更改该JSON数据的值（如将

test_1

改为

my_own_data

），但不可改变该示例数据结构。

```
{
  "key1": "test value 1",
  "key2": "test value 2"
}
```

3) 点击【运行】按钮，代码开始运行并将显示测试结果。其中：

- 函数返回值部分将显示运行结果（成功或失败），还将显示代码中

```
return
```

语句返回的函数执行结果。

- 运行信息部分将显示函数运行的时间、内存等信息。
- 日志部分将显示函数运行时生成的日志，包括用户代码中的打印语句、函数运行失败trace

stack等，将会写入至日志模块。

4) 您可以多运行几次，并点击【日志】选项卡来查看每一次运行的日志信息。

示例二：创建云函数下载 COS Bucket 中上传的文件

步骤一：创建 DownloadImage 函数

1) 登录腾讯云控制台，选择【对象存储服务】。

2) 点击【Bucket列表】选项卡下的【创建Bucket】按钮，新建一个COS Bucket。

3) 设置COS Bucket的名称如

testbucket

，选择地域为

华南

，设置访问权限为默认值

公有读私有写

并设置CDN加速为默认值

关闭

，点击【确定】按钮新建一个COS Bucket。

4) 在控制台上导航至【无服务器云函数】，在

广州

地域下点击【创建函数】按钮，进入新建函数页面。

5) 填写函数名称

DownloadImage

，其他配置项保持默认选项即可。

6) 点击【下一步】按钮，进入函数代码编辑页，默认选择【在线编辑】，并在【模版】中选择

COS Put Object

模版。此时，执行方法和代码将填入模版的默认值：

执行方法显示

index.main_handler

。表示无服务器云函数控制台会将此段代码自动保存为

index.py

文件，并压缩该文件上传至 SCF 平台以创建云函数。

函数代码显示以下代码片段，修改其中

appid, secret_id, secret_key, region

参数字段，改为您的实际数据。其中：

- appid可在控制台【账户信息】中获得



- secret_id 和 secret_key可在控制台【云API密钥】中获得



云 API 密钥

个人 API 密钥

使用云 API 请在调用前使用 API 密钥获取签名，否则将无法调用 API，详见[鉴权签名算法](#)。

API 密钥是构建腾讯云 API 请求的重要凭证，使用腾讯云 API 可以操作您名下的所有腾讯云资源，为了您的财产和服务安全，请妥善保管和定期更换或删除旧密钥。

+ 新建密钥

APPID	密钥	创建时间	状态
1251111111	SecretId: AKIDYDh085xQp48161... SecretKey: ***** 显示	2016-12-11 15:22:18	已启用
1251111111	SecretId: AKIDutrojKl3CKQZNAr76.. SecretKey: ***** 显示	2017-05-16 16:29:02	已启用

- region 为函数和 COS Bucket 所在地域，支持

sh、gz、bj

三个值。请注意，必须保持和 COS Bucket

在同一个地域。此处第一步中创建的存储桶位于华南（广州），因此代码中的 region 值必须为

gz

```
import json
import urllib
import commands
import logging
from qcloud_cos import CosClient
from qcloud_cos import DownloadFileRequest
print('Loading function')
appid = 1251111111 #please change to your appid. Find it in Account Info
secret_id = u'AKIDYDh085xQp48161uOn2CKKVbeebvDu9Ib' #please change to your API secret id.
Find it in API secret key pair
secret_key = 'lLkxx40kIfuyqW0IOIOWqyueCYjlgzqE' #please change to your API secret key. Find it in
API secret key pair
region = u'gz'
def main_handler(event,context):
    logger = logging.getLogger()
```

```
bucket = event['Records'][0]['cos']['cosBucket']['name']
key = urllib.unquote_plus(event['Records'][0]['cos']['cosObject']['key'].encode('utf8'))
try:
    cos_client = CosClient(appid, secret_id, secret_key, region)
    request = DownloadFileRequest(bucket, key, '/tmp'+key)
    download_file_ret = cos_client.download_file(request)
    if download_file_ret['code'] == 0:
        logger.info("Download file [%s] Success" % key)
        logger.info("find local file:" + commands.getoutput('ls /tmp'))
        return "download success"
    else:
        logger.error("Download file [%s] Failed, err: %s" % (key, download_file_ret['message']))
        return -1
except Exception as e:
    print(e)
    print('Error getting object {} from bucket {}. Make sure the object exists and your bucket is in the
same region as this function.'.format(key, bucket))
    raise e
```

7) 点击【下一步】按钮，进入触发方式页面。点击【新建触发方式】按钮为函数添加一个新的触发器，选择触发方式为

COS触发

，并选择刚刚创建的

testbucket

存储桶，事件类型选择为

文件上传

，点击【保存】按钮。

8) 点击底部【完成】按钮，此时控制台会自动生成代码程序包并上传至 SCF 平台以创建云函数。您可以点击云函数列表页中刚刚创建的

DownloadImage

函数进入云函数详情页。

步骤二：调用 DownloadImage 函数

请用户按照以下步骤来测试刚刚创建的

DownloadImage

函数。

使用 COS 上传文件测试

1. 登录腾讯云控制台，导航至【对象存储服务】，在 Bucket 列表中选择第一步中创建的

TestBucket

Bucket，点击【上传文件】按钮，上传一张[示例图片:testimage.jpeg](#)

（ 您可将此示例图片下载到本地并用作测试 ）。

2. 导航至【无服务器云函数】，选择刚刚创建的

DownloadImage

。

3. 点击【日志】选项卡，观察是否有刚刚上传图片后的函数运行日志。日志中应有本地下载下来的图片的日志，类似：

Loading function

Uri is http://TestBucket-1251111111.cosgz.myqcloud.com/testimage.jpeg?sign=QIEyq+WH8g5RpD+L6sPk05XhVQthPTEyNTE3NjIyMjcmaz1BS0IEWURoMDg1eFFwNDgxNjF1T24yQ0tLVmJlZWJ2RHU2ak8mZT0xNDk2ODM5NDQ3JnQ9MTQ5NjgzOTE0NyZyPTk1NDI3NjgyNCZmPS8xNDcyNjQwNzgwXzg0X3cxNjE0X2g0NDAucG5nJmI9ZG9uZ3l1YW50ZXEE

Starting new HTTP connection (1): TestBucket-1251111111.cosgz.myqcloud.com

http://TestBucket-1251111111.cosgz.myqcloud.com:80 "GET /testimage.jpeg?sign=QIEyq+WH8g5RpD+L6sPk05XhVQthPTEyNTE3NjIyMjcmaz1BS0IEWURoMDg1eFFwNDgxNjF1T24yQ0tLVmJlZWJ2RHU2

ak8mZT0xNDk2ODM5NDQ3JnQ9MTQ5NjgzOTE0NyZyPTk1NDI3NjgyNCZmPS8xNDcyNjQwNzgwXzg
0X3cxNjE0X2g0NDAucG5nJmI9ZG9uZ3I1YW50ZXEE HTTP/1.1" 200 62296
Download file [/testimage.jpeg] Success
testimage.jpeg

手动模拟测试

您还可以通过手动输入类似COS触发的测试数据来观察函数运行状态。

1) 在测试函数弹出框中，从测试模版中选择

COS 上传/删除文件测试代码

，默认测试数据将出现在窗口中。需要做以下修改：

- 把

cosBucket

中

name

更换成刚刚创建的

TestBucket

；

- 把

cosObject

中

key

的值更换成刚刚上传的文件

\testimage.jpeg

;

如下：

```
{
  "Records": [
    {
      "event": {
        "eventVersion": "1.0",
        "eventSource": "qcs::cos",
        "eventName": "cos:ObjectCreated:*",
        "eventTime": "1970-01-01T00:00:00.000Z",
        "eventQueue": "qcs:0:cos:gz:1251111111:cos",
        "requestParameters": {
          "requestSourceIP": "111.111.111.111",
          "requestHeaders": {
            "Authorization": "Example"
          }
        }
      },
      "cos": {
        "cosSchemaVersion": "1.0",
        "cosNotificationId": "设置的或返回的 ID",
        "cosBucket": {
          "name": "TestBucket", # Notice Here
          "appid": "1251111111",
          "region": "gz",
        },
      },
    }
  ]
}
```

```
"cosObject":{
  "key":"/testimage.jpg", # Notice Here
  "size":"1024",
  "meta":{
    "Content-Type": "text/plain",
    "x-cos-meta-test": "自定义的 meta",
    "x-image-test": "自定义的 meta"
  },
  "url": "访问文件的源站url"
}
}
}
]
```

2) 点击【运行】按钮，代码开始运行并将显示测试结果。其中：

- 函数返回值部分将显示运行结果，还将显示代码中

return

语句返回的函数执行结果。

- 运行信息部分将显示函数运行的时间、内存等信息。
- 日志部分将显示函数运行时生成的日志，包括用户代码中的打印语句、函数运行失败trace stack等，将会写入至日志模块。

3) 您可以多运行几次，并点击【日志】选项卡来查看每一次运行的日志信息。