

# Cloud Storage Python SDK develop guide

## 目录

1.	准备.....	- 2 -
1.1	环境.....	- 2 -
1.2	使用到的 Python module.....	- 2 -
1.3	下载 sndacspylib.....	- 2 -
1.4	安装 sndacspylib.....	- 2 -
2.	如何编写代码.....	- 3 -
2.1	创建连接.....	- 3 -
2.2	创建云存储服务对象.....	- 3 -
2.3	获取 Bucket 列表.....	- 3 -
2.4	添加新 Bucket.....	- 3 -
2.5	删除 Bucket.....	- 3 -
2.6	创建云存储数据对象.....	- 3 -
2.7	上传数据.....	- 4 -
2.8	获取数据 Meta 信息.....	- 4 -
2.9	下载数据.....	- 4 -
2.10	删除数据.....	- 4 -
2.11	生成 Bucket Policy 对象以及 xml 格式字符串.....	- 4 -
2.12	设置 Bucket Policy.....	- 5 -
2.13	获取 Bucket Policy.....	- 5 -
2.14	删除 Bucket Policy.....	- 5 -
2.15	列出未完成的 Multipart upload.....	- 5 -
2.16	初始化 Multiupload.....	- 5 -
2.17	取消 Multiupload.....	- 5 -
2.18	上传 Multiupload parts.....	- 5 -
2.19	完成 Multiupload.....	- 6 -
2.19	获取 Multiupload 对象列表.....	- 6 -
3	存储服务对象.....	- 6 -
3.19	SNDA_CS.....	- 6 -
3.20	SNDA_Bucket.....	- 6 -
3.21	SNDA_Object.....	- 7 -
4	代码示例.....	- 7 -

## 1. 准备

### 1.1 环境

使用盛大云存储的 Python SDK 进行开发需要具备 python 2.6 以上版本

### 1.2 使用到的 Python module

os, sys, httplib, urllib, socket, logging, time, base64, hmac, sha, md5, xml.sax, string

由于需要将存储服务返回的 RFC3339 日期格式转化成 datetime 数据类型，因此需要使用 rfc3339 这个 module

Linux 安装:

```
pip install -e git://github.com/tonyg/python-rfc3339.git#egg=rfc3339
```

Windows 安装:

- 1) 下载 <http://github.com/tonyg/python-rfc3339/tarball/master>
- 2) 解压缩后，执行 python setup.py install 安装

### 1.3 下载 sndacspylib

将 sndacspylib 下载到需要安装的目录，解压缩

### 1.4 安装 sndacspylib

- 1) 修改目录 sndacspylib/config/下，cs.properties 文件中的 AccessKey, SecretKey 的值为盛大云存储用户提供的相应 key 值；CheckHash 可接受的配置为 True 和 False，True 表示上传和下载数据时会本地校验数据的 MD5 值，False 表示不作本地校验；SecureComm 可接受的配置为 True 和 False，True 表示使用 HTTPS 连接，端口使用 443，False 表示使用 HTTP 连接，端口使用 80
- 2) 在 sndacspylib 目录下运行命令，python setup.py install

## 2. 如何编写代码

### 2.1 创建连接

```
import sndacpylib.snda_cs.cs_util as CSUtil
from sndacpylib.snda_cs_config import Config
connection = CSUtil.CS.SNDAAuthConnection(Config.CSProperties['AccessKey'],
                                           Config.CSProperties['SecretKey'],
                                           True)
```

### 2.2 创建云存储服务对象

```
service = CSUtil.SNDA_CS(ConnectionObject = connection)
```

### 2.3 获取 **Bucket** 列表

```
bucket_list = service.get_list_of_buckets()
for bucket in bucket_list:
    '''
    bucket instance has following attributes
    '''
    print bucket.name, bucket.creation_date, bucket.location
```

### 2.4 添加新 **Bucket**

第一个参数用以指定 bucket 的名字, 第二个参数为 bucket 的位置信息, 目前可选的 idc 位置有 huabei-1 和 huadong-1, 默认为 huabei-1

```
service.add_bucket("bucket_name", "huabei-1")
```

### 2.5 删除 **Bucket**

```
service.delete_bucket("bucket_name")
```

### 2.6 创建云存储数据对象

```
import sndacpylib.snda_cs.cs_util as CSUtil
```

```
object = CSUtil.SNDA_Object(ConnectionObject = connection, "bucket_name",  
"object_name")
```

## 2.7 上传数据

第一个参数指定需要上传的文件的路径，第二个参数指定需要指定的数据的 meta 信息，默认可以不传入此参数

```
object.put_object_from_file("filepath/file")
```

headers 为用户需要自定义的 HTTP Header 信息，例如，用户可以自定义 Content-Type 为值 XXX

## 2.8 获取数据 Meta 信息

```
infos = object.get_object_info()
```

## 2.9 下载数据

```
object.get_object_to_file("filepath/file")
```

## 2.10 删除数据

```
object.delete_object()
```

## 2.11 生成 Bucket Policy 对象以及 xml 格式字符串

```
from sndacpylib.snda_cs_model import *  
effect = Effects.Allow  
actions = Actions.AllActions  
resources = "*"   
conditions = {ConditionTypes.Bool: {AvailableKeys.SecureTransport: True}, \   
              ConditionTypes.IpAddress: {AvailableKeys.SourceIp: "192.168.0.24"}}  
statement = PolicyStatement(Sid = None,   
                            Effect = effect,   
                            Principal = None,   
                            Action = actions,   
                            Resource = resources,   
                            Condition = conditions)  
Statement.sid_regenerate()  
policy = BucketPolicy(Id = "your_id", Version = None, Statement = [statement])  
import json
```

```
policy_xml = json.dumps(policy.toDict())
```

## 2.12 设置 Bucket Policy

```
import sndacpylib.snda_cs.cs_util as CSUtil
bucket = CSUtil.SNDA_Bucket(connection, "bucket_name")
bucket.set_policy(policy_xml)
```

## 2.13 获取 Bucket Policy

```
bucket_policy_string = bucket.get_policy()
```

## 2.14 删除 Bucket Policy

```
bucket.delete_policy()
```

## 2.15 列出未完成的 Multipart upload

```
list_result = bucket.list_multipart_uploads(key_marker='key-marker',
                                             prefixDir='prefix',
                                             delimiter='delimiter',
                                             upload_id_marker='upload-id-marker')

for upload in list_result.uploads:
    print upload.key, upload.initiated
for common_prefix in list_result.common_prefixes:
    print common_prefix.prefix
```

## 2.16 初始化 Multiupload

```
object.initiate_multipart_upload()
```

## 2.17 取消 Multiupload

```
object.abort_multipart_upload(object.init_result.upload_id)
```

## 2.18 上传 Multiupload parts

```
part1 = object.upload_part_from_file(object.init_result.upload_id,
                                     '1',
```

```
        'filepath/file')
part2 = object.upload_part_from_data(object.init_result.upload_id,
        '2',
        'I am No.2.')
```

## 2.19 完成 Multiupload

```
from xml.dom.minidom import Document
complete_content = CompleteMultipartUpload([part1, part2])
document = Util.object_convert_to_xml(Document(), complete_content)
object.complete_multipart_upload(object.init_result.upload_id, document.toxml())
```

## 2.19 获取 Multiupload 对象列表

```
list_parts_result = object.list_parts(object.init_result.upload_id)
```

# 3 存储服务对象

## 3.19 SNDA\_CS

该类型封装了对盛大云存储 Bucket 数据类型的相关操作，其中包含：

- 1) 获取 bucket 列表: `get_list_of_buckets()`
- 2) 获取 bucket 对象: `get_bucket_name(bucket_name)`
- 3) 添加 bucket: `add_bucket(bucket_name)`
- 4) 删除 bucket: `delete_bucket(bucket_name)`

## 3.20 SNDA\_Bucket

该类型封装了对盛大云存储 Bucket 下数据进行同步以及 Bucket Policy 的相关操作，其中包含：

- 1) 获取 bucket 下文件列表: `get_list_of_keys_in_bucket()`
- 2) 设置 policy: `set_policy(policy)`
- 3) 获取 policy: `get_policy()`
- 4) 删除 policy: `delete_policy()`
- 5) 列出未完成 Multipart uploads: `list_multipart_uploads()`

### 3.21 SNDA\_Object

该类型封装了对盛大云存储 Object 数据类型的相关操作，其中包含：

- 1) 上传文件:  
`put_object_from_file(file_name, headers)`  
`put_object_from_stream(size, stream, headers, metadata)`  
`put_object_from_string(string, headers)`
- 2) 下载文件:  
`get_object_to_file(file_name)`  
`get_object_to_stream()`
- 3) 获取文件信息: `get_object_info()`
- 4) 删除文件: `delete_object()`
- 5) 初始化 Multiupload: `initiate_multipart_upload()`
- 6) 上传 Part:  
`upload_part_from_file(upload_id, part_number, file_name)`  
`upload_part_from_data(upload_id, part_number, data)`
- 7) 完成 Multipart upload: `complete_multipart_upload(upload_id, complete_parts)`
- 8) 终止 Multipart upload: `abort_multipart_upload(upload_id)`
- 9) 列出 Multipart upload 已上传的 part:  
`list_parts(upload_id, max_parts, part_number_marker)`

## 4 代码示例

Python IDLE 中执行下述代码：

```
>>>import sndacpylib.snda_cs.cs_util
>>>from sndacpylib.snda_cs_config import Config
>>>conn=sndacpylib.snda_cs.cs_util.CS.SNDAAuthConnection(Config.CSProperties['
AccessKey'], Config.CSProperties['SecretKey'],
(Config.CSProperties['SecureComm']==False))
>>>cloud_storage=sndacpylib.snda_cs.cs_util.SNDA_CS(ConnectionObject = conn)
>>>cloud_storage.get_list_of_buckets()
>>>for entry in cloud_storage.ListOfBuckets:
...     print '%s\t\t\t%s' % (entry.name, entry.creation_date)
...
>>>
```

完整 python 代码示例：

```
from sndacpylib.snda_cs_config import *
```

```
import sndacspylib.snda_cs.cs_rest as CSRest
import sndacspylib.snda_cs.cs_util as CSUtil

import uuid

# initialize connection
connection = CSRest.SNDAAuthConnection(Config.CSProperties['AccessKey'],
Config.CSProperties['SecretKey'], True)

# initialize service
service = CSUtil.SNDA_CS(ConnectionObject = connection)

# list buckets
bucket_list = service.get_list_of_buckets()

for item in bucket_list:
    print bucket_list[item]

bucket_name = str(uuid.uuid4())
# add bucket
service.add_bucket(bucket_name, 'huadong-1')

object_name = str(uuid.uuid4())
# initialize object
object = CSUtil.SNDA_Object(connection, bucket_name, object_name)

# add object
object.put_object_from_file("filepath/file")

# head object
infos = object.get_object_info()
print infos.metadata
print infos.size
print infos.last_modified

# get object
object.get_object_to_file("filepath/file.bak")

import commands
md5sum1 = commands.getoutput("md5sum filepath/file").split()[0]
md5sum2 = commands.getoutput("md5sum filepath/file.bak").split()[0]
print md5sum1
```



```
print md5sum2

# initialize bucket
bucket = CSUtil.SNDA_Bucket(connection, bucket_name)

# list object
object_list = bucket.get_list_of_keys_in_bucket("", "")
for item in object_list:
    print item

# add object from string
object.put_object_from_string('I am a string.')

# delete object
object.delete_object()

# delete bucket
service.delete_bucket(bucket_name)
```