背景:本文档定义了中间件 API 的说明和使用,中间件的提供方式为 C++ 动态链接库。

基础 API 列表

1)验证签发 token 接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “validate”
strJson 格式={“password”:“123456”,“email”:“messi@126.com”, “uuid”: “201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -X POST -d '{"password": "123456", "email":"messi@126.com"}' http://localhost:443/oauth/access_token

2)获取容器列表
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“containerList”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”: “201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -s http://IP:Port/v1/AUTH_ e8b18580e5b44cb79b10bd0f7a03bbdc -X GET -H "X-Auth-Token:22ec6a37ed6b4b23a435e6b9050f21a4"

3)创建容器
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“createContainer”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Container-Name”:“new_container”,“uuid”: “201506241010010”}
strResponse 为空
输出: strResponse 为服务器返回信息
CURL 示例:curl -s http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/new_container -X PUT -H "X-Auth-Token:22ec6a37ed6b4b23a435e6b9050f21a4"

4)删除容器
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“deleteContainer”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Container-Name”:“del_container”,“uuid”: “201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:

```
curl -s http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/del_container -X
DELETE -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

5)列出对象

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“listContainerObjects”
```

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Container-Name”:“my_container”,“uuid”: “201506241010010”}

```
strResponse = ?op=LISTDIR&recursive=<true|false>
```

输出:

strResponse 为服务器返回信息

CURL 示例:

```
curl -i
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/my_container?op=LISTDIR
&recursive=true" -H "X-Auth-Token:22ec6a37ed6b4b23a435e6b9050f21a4"
```

6)创建对象

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“createObject”
```

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Local-Object” :
“/home/jack/ufw.log”,“Container-Name”:“my_container”, “Object-Name”:
“ufw.log”,“uuid”:
“201506241010010”}

strResponse 为空

输出:strResponse 为服务器返回信息

CURL 示例

```
curl -X PUT -i -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4" -T ufw.log
http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/my_container/ufw.log
```

7)删除对象

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“deleteObject”
```

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Container-Name”:“my_container”, “Object-Name”: “ufw.log”,“uuid”: “201506241010010”}

strResponse 为空

输出:

strResponse 为服务器返回信息

CURL 示例:

```
curl -s
http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/my_container/ufw.log -
X DELETE -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

8)复制对象

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“copyObject”
```

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Container-Name”:“my_container”, “Object-Name”: “ufw.log”, “Destination” :
“your_container”,“uuid”:
“201506241010010”}

strResponse 为空

输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -s
http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/my_container/ufw.log -
X COPY -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4" -H "Destination:
/your_container/ufw.log"
```

9)读取对象
```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```
输入:
```
method=“readObject”
```
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Local-
Object” :“/home/jack/ufw.log”,“Container-Name”:“my_container”, “Object-Name”:
“ufw.log”,“uuid”:
“201506241010010”}
strResponse 为空
输出:strResponse 为服务器返回信息
CURL 示例:
```
curl -s
http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/my_container/ufw.log -
X GET -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

高级 API 列表

10)获取配额信息
```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```
输入:
```
method=“getQuotaInfo”
```
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:
“201506241010010”}
strResponse = ?op=info[&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -i -X GET
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/quota?op=info&type=NOR
MAL" -H "X-
Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

11)上传文件
```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```
输入:
```
method=“fileUpload”
```
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Source”:
“/home/jack/ufw.log”, “Destination”: “/temp/ufw.log”,“uuid”:
“201506241010010”,”time”=”2015-07-08T10:00:00”}
```
strResponse = ?op=CREATE
[&overwrite=<true|false>][&type=<NORMAL|PRIVATE|BACKUP>][&metadata=<STRING
>]
[&mode=<NORMAL|ENCRYPT|COMPRESS|ENCRYPT_COMPRESS>][&storetype=<USE
R|CORRESPONDING>]
```
输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -i -X PUT -T
```

```
ufw.log"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/temp/ufw.log?
op=CREATE
&overwrite=true&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

12)上传临时文件

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“fileUpload”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Source”:
“/home/jack/tempfile”, “Destination”: “/segment/tempfile”,“uuid”:
“201506241010010”,”time”=”2015-07-08T10:00:00”}
strResponse = ?op=CREATE
[&overwrite=<true|false>][&type=<NORMAL|PRIVATE|BACKUP>][&metadata=<STRING
>]
[&mode=<NORMAL|ENCRYPT|COMPRESS|ENCRYPT_COMPRESS>][&storetype=<USE
R|CORRESPONDING>]
```

输出:

```
strResponse 为服务器返回信息
CURL 示例:
curl -i -X PUT -T ufw1
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/segment/tempfile?
op=CREA
TE&overwrite=true&type=NORMA
L" -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

13)合并文件

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“mergeFile”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Merge-Info” :
[{"path":
"/segments/seg_test/1","etag":"9951ef01bc03745e6ebf3b50e990bc67","size_bytes":75
72},{"p
ath":"/segments/seg_
test/2","etag":"8dd16a3d50854caae6a23917d41688f3","size_bytes": 27312}],“uuid”:
“201506241010010”, “Destination”: “/normal/seg_test”,”X-Static-Large-
Object”:”true”}
strResponse = ?multipart-manifest=put
```

输出:

```
strResponse 为服务器返回信息
CURL 示例:
curl -i -X PUT -d '[{"path":
"/segments/seg_test/1","etag":"9951ef01bc03745e6ebf3b50e990bc67","size_bytes":75
72},{"p
ath":"/segments/seg_
test/2","etag":"8dd16a3d50854caae6a23917d41688f3","size_bytes": 27312}]'
"http://IP:port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/seg_test?
multipart-manifest=put"
-H "X-Auth-Token:6XyrIAUfRYpRGoRnrhJWILVbrqPergjDKde3e9Fi" -H "X-Static-
Large-Object: True"
```

14)创建目录

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“createDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Destination”:
“/normal/new_dir”,“uuid”: “201506241010010”}
```

strResponse = ?op=MKDIRS[&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i -X PUT
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/new_dir?op=MKDI
RS&type=NORMAL" -H "X-
Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

15)删除文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
输入:
method=“deleteFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Destination”:
“/normal/del_dir”,“uuid”: “201506241010010”}
strResponse
= ?op=DELETE[&ftype=<f|d>][&recursive=<true|false>][&type=<NORMAL|PRIVATE|BAC
KUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i -X DELETE
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/del_dir?op=DELET
E&ftype=f&recursive=true&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

16)批量删除文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
输入:method=“batchDeleteFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “list”:
[{“path” :
“/normal/myDir3”, “ftype”: “d”}, {“path” : “/normal/myDir2/log”, “ftype”:
“f”}],“uuid”:
“201506241010010”}
strResponse = ?op=DELETE&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i -X POST –d
' {"list":[{"path":"/normal/myDir3","ftype":"d"},
{" path ":"/normal/myDir2.log","ftype":"f"}]} "
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/batch?op=DELETE&type=
NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

17)移动文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
输入:
method=“moveFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Source” :
“/normal/myDir3/log”, “Destination” : “/temp”,“uuid”: “201506241010010”}
strResponse = ?op=MOVE[&ftype=f|d][&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i -X PUT
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/myDir3/log?op=MO

VE&ftype=f&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
-H "Destination: /temp/"

18)批量移动文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“batchMoveFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “list”:
[{“from” :
“/apps/album/a/b/c”, “to”: “/apps/album/b/b/c”,“ftype”: “f”}, {“from” :
“/apps/album/a/b/d”,
“to”: “apps/album/b/b/d”,“ftype”: “d”}],“uuid”: “201506241010010”}
strResponse = ?op=MOVE[&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息CURL 示例:
curl -i -X POST –d
" {"list":[
{"from":"/apps/album/a/b/c","to":"/apps/album/b/b/c",
“ftype”:”f”},{"from":"/apps/album/a/b/d",
"to":"/apps/album/b/b/d",“ftype”:”d”}]
}"
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/batch?op=MOVE&type=NO
RMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

19)复制文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“copyFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Source” :
“/normal/myDir3”, “Destination” : “/temp”,“uuid”: “201506241010010”}
strResponse = ?op=COPY[&ftype=<f|d>][&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i -X PUT
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/myDir3?op=COPY
&ftype=d&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
-H "Destination: /temp/"

20)批量复制文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“batchCopyFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “list”:
[{“from” :
“/apps/album/a/b/c”, “to”: “/apps/album/b/b/c”,“ftype”: “f”}, {“from” :
“/apps/album/a/b/d”,
“to”: “apps/album/b/b/d”,“ftype”: “d”}],“uuid”: “201506241010010”}
strResponse = ?op=COPY&type=<NORMAL|PRIVATE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i -X POST –d
"{"list":[
{

```
"from":"/apps/album/a/b/c",
"to":"/apps/album/b/b/c",
“ftype”:”f”
},
{
"from":"/apps/album/a/b/d",
"to":"/apps/album/b/b/d",
“ftype”:”d”
}
]
}"
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/batch?op=COPY&type=NO
RMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

21)读取文件

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“readFile”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“Source” :
“/home/jack/file.txt”, “Destination” : “/normal/file.txt”,“uuid”:
“201506241010010”}
strResponse
= ?op=OPEN[&offset=<LONG>][&length=<LONG>][&type=<NORMAL|PRIVATE|
BACKUP>][&version=<LATEST|指定版本
>][&mode=<NORMAL|ENCRYPT|COMPRESS|ENCRYPT_COMPRESS>]
```

输出:

```
strResponse 为服务器返回信息
CURL 示例:
curl -i -L
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/file.txt?
op=OPEN&f
type=f&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

22)获取文件历史

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“getFileHistory”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Destination” :
“/normal/file.txt”,“uuid”: “201506241010010”}strResponse = ?
op=GETHISTORY[&type=<NORMAL|PRIVATE|BACKUP>]
```

输出:

```
strResponse 为服务器返回信息
CURL 示例:
curl -i -L
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/file.txt?
op=GETHIS
TORY&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

23)获取用户操作历史

```
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
```

输入:

```
method=“getOperateHistory”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:
“201506241010010”}
strResponse = ?op=GET_OP_HISTORY[&recent=<INT>]
```

输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -i -L
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc?op=GET_OP_HISTORY&r
ecent=1"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

24)清除用户操作历史
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“deleteOperateHistory”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”: “201506241010010”}
strResponse = ?op=DELETE_HISTORY[&recent=<INT>]
输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -i -L
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc?op=DELETE_HISTORY&r
ecent=1"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

25)创建符号链接
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“createSymbolicLink”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Source” : “/normal/1.jpg”,“uuid”: “201506241010010”}
strResponse
= ?op=CREATESYMLINK&destination=<PATH>[&type=<NORMAL|PRIVATE|BACK
UP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -i -X PUT
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/1.jpg?op=CREATES
YMLINK&destination=/temp
/1.jpg&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
```

26)重命名文件,目录
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“renameFileDir”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Source” : “/picture/1.jpg”,“uuid”: “201506241010010”}
strResponse
= ?op=RENAME&destination=<PATH>[&ftype=<f|d>][&type=<NORMAL|PRIVA
TE|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
```
curl -i -X PUT
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/picture/1.jpg?op=RENAME
&destination=/picture/2.j
pg&ftype=f&type=NORMAL"
```

-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

27)获取文件属性

data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)

输入:method=“getFileAttribute”

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Destination” : “/picture/1.jpg”,“uuid”: “201506241010010”}

strResponse

= ?op=GETFILEATTR[&type=<NORMAL|PRIVATE|BACKUP>][&version=<LATEST|指定版本>]

输出:

strResponse 为服务器返回信息

CURL 示例:

curl -i "http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/picture/1.jpg?op=GETFILE ATTR&type=NORMAL" -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

28)获取回收站中文件列表

data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)

输入:

method=“getRecycle List ”

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”: “201506241010010”}

strResponse = ?op=GETRECYCLER[&start=<LONG>][&limit=<LONG>] [&type=<NORMAL|SECRET|BACKUP>]

输出:

strResponse 为服务器返回信息

CURL 示例:

curl -i "http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/recycle/user?op=GETRECYCLER&type=NORMAL" -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

29)从回收站恢复

data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)

输入:

method=“moveRecycle”

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,"list":[{"uuid":"vfgUc2hO-OKmCeg-l4OC","path":"/normal/myDir2.log","ftype":"f"},{"uuid":"wXQUs5DY-i0lrOH-2Bh8","path":"/normal/myDir1","ftype":"d"}],“uuid”: “201506241010010”}

strResponse = ?op=RECYCLER[&type=<NORMAL|PRIVATE|BACKUP>]

输出:strResponse 为服务器返回信息

CURL 示例:

curl -i -X POST -d '{"list":[{"uuid":"vfgUc2hO-OKmCeg-l4OC","path":"/normal/myDir2.log","ftype":"f"},{"uuid":"wXQUs5DY-i0lrOH-2Bh8","path":"/normal/myDir1","ftype":"d"}]}' "http://localhost:443/v1/AUTH_zhu__feng006163com/batch?op=MOVERECYCLE&type=NORMAL" -H "X-Auth-Token:6XyrIAUfRYpRGoRnrhJWILVbrqPergjDKde3e9Fi"

30)清空回收站

data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)

输入:

method=“cleanRecycle”

strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:
“201506241010010”}
strResponse = ?op=RECYCLER[&type=<NORMAL|SECRET|BACKUP>]
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -i –X POST
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/clearrecycle?op=RECYCLE
R&type=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

31)设置权限
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
输入:
method=“setPermission”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Destination” :
“/picture/1.jpg”,“uuid”: “201506241010010”}
strResponse
= ?op=SETPERMISSION[&permission=<OCTAL>][&type=<NORMAL|PRIVATE|BA
CKUP>]
输出:strResponse 为服务器返回信息
CURL 示例:
curl -i -X
PUT"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/picture/1.jpg?
op=SETPER
MISSION&permission=500&ty
pe=NORMAL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

32)获取用户文件列表
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
输入:
method=“getFileList”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “Destination” :
“/normal/myDir”,“uuid”: “201506241010010”}
strResponse = ?op=LISTDIR[&recursive=<true|false>][&ftype=<f|d>][&type=<NORMAL|
PRIVATE|BACKUP>]
输出:strResponse 为服务器返回信息
CURL 示例:
curl -i
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/normal/myDir?op=LISTDI
R&ftype=d&recursive=false&type=NORM
AL"
-H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

33)创建存储空间配额
data_pipeline(const std::string &method, const std::string &strJson, std::string
&strResponse)
输入:
method=“createStorageQuota”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,"X-Account-Meta-
Quota-Bytes”: “1000000000”,“uuid”: “201506241010010”}
strResponse 为空
输出:strResponse 为服务器返回信息
CURL 示例:
curl -i -X POST
"http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/quota?op=createstorage"
-H

"X-
Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"
-H "X-Account-Meta-Quota-Bytes: 1000000000"

34)用户注册初始化
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method=“userRegister”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:“201506241010010”}
strResponse 为空
输出:strResponse 为服务器返回信息
CURL 示例:
curl -i
-X PUT -H "X-Auth-Token:6XyrIAUfRYpRGoRnrhJWILVbrqPergjDKde3e9Fi"
http://IP:Port/v1/AUTH_e8b18580e5b44cb79b10bd0f7a03bbdc/register

服务器扩展 API

1)刷新 token 接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “refreshToken”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”, “uuid”:“201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -k -X POST -d '{}' https://10.3.3.236:443/oauth/token_refresh -H "X-Auth-Token:A26NY7CUtG5NnUS0dG3agGZa9mvAaQ16T7R4Tmek" --cacert /root/ssl/ca.crt

2)获取服务器版本接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “getCloudServVersion”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:“201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -k -i https://10.3.3.236:443/v1/AUTH_Jimmy2163com/cloud_server_version -H "X-Auth-
Token:Yznvej4nPm3VFKC4oBoHXpEifc7oqWpP9v38zPq6" --cacert /root/ssl/ca.crt

3)验证 token 接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “verifyToken”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:“201506241010010”}
strResponse 为空输出:
strResponse 为服务器返回信息

CURL 示例:
curl -i -X POST "https://10.3.3.236:443/oauth/verify_token" -H "X-Auth-Token: 22ec6a37ed6b4b23a435e6b9050f21a4"

4)注册网络帐号接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “networkRegister”
strJson 格式={“username”:“messi”,“password”:“123456”,“password_confirmation”:“123456”,“email”:“messi@163.com”,“name”:“zhang”,“uuid”: “201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -k -X POST -d '{"username":"messi","password_confirmation":"123456","password":"123456", "email":"messi@163.com","name":"zhang"}' https://10.3.3.236:443/v1/network_register

5)注册网络帐号+注册云服务接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “netCloudRegister”
strJson 格式={“username”:“messi”,“password”:“123456”,“password_confirmation”:“123456”,“email”:“messi@163.com”,“name”:“zhang”,“uuid”: “201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -k -X POST -d '{"username":"messi","password_confirmation":"123456","password":"123456", "email":"messi@163.com","name":"zhang"}' https://localhost:443/v1/user_register

6)验证网络帐号接口
data_pipeline(const std::string &method, const std::string &strJson, std::string &strResponse)
输入:
method = “validateNetworkAccout”
strJson 格式={“X-Auth-Token”:“22ec6a37ed6b4b23a435e6b9050f21a4”,“uuid”:“201506241010010”}
strResponse 为空
输出:
strResponse 为服务器返回信息
CURL 示例:
curl -k -X POST https://10.3.3.236:443/v1/token_validation
Token:22ec6a37ed6b4b23a435e6b9050f21a4"
-H "X-Auth-Token:22ec6a37ed6b4b23a435e6b9050f21a4"

版本信息 API

获取版本信息
bool getVersionInfo(std::string &version, std::string &info)
输入:无
输出:version 为版本号 ,info 为编译相关信息
返回值:bool

数据库查询相关 API

1)获取上传任务列表
bool getUploadTask(std::string &queryStr)
输入:无
输出:上传任务进度
返回值:bool

2)获取下载任务列表
bool getDownloadTask(std::string &queryStr)
输入:无
输出:下载任务进度
返回值:bool

3)查询任务执行结果
bool checkTaskStatus(const std::string &uuid, std::string &queryStr)
输入:uuid
输出:任务执行结果
返回值:bool