许一清

QQ：2023503307

接口文档

通过HTTP请求来配置服务器和管理播放器

Opera Audio接口文档

基于HTTP协议

# 

## 注意：

1. **大体上分为三个模块，配置服务器、播放器操作、更新操作。**
2. **每次访问的IP不确定，根据每次扫描结果而定。**
3. **端口为8888，个别端口是9000，我会特别注明。**
4. **个别请求会没有响应，比如重启服务器的请求，按照成功来处理即可。**
5. **所有请求都是GET请求，返回都是JSON。**
6. **JSON规范：成功：success:true,status:200；失败：success:false,status:500。**
7. **响应失败JSON：{"status":500,"success":false,”error”:”错误原因”}，可以显示出来用于调试。**

## 1 配置服务器模块：

### 1.1解码器列表：

url：/config/list/squeezelite

请求参数：无

响应示例：

{"data":{"DW-I2S-HDMI":"hw:CARD=DWI2SHDMI,DEV=0 - DW-I2S-HDMI, - Direct hardware device without any conversions","USB Audio OnBoard USB Audio":"hw:CARD=OnBoard,DEV=0 - USB Audio OnBoard, USB Audio - Direct hardware device without any conversions","USB Audio OnBoard USB Audio #1":"hw:CARD=OnBoard,DEV=1 - USB Audio OnBoard, USB Audio #1 - Direct hardware device without any conversions","USB Audio OnBoard USB Audio #2":"hw:CARD=OnBoard,DEV=2 - USB Audio OnBoard, USB Audio #2 - Direct hardware device without any conversions"},"status":200,"success":true}

解释：其中key1、key2……是给用户显示的，value第二个API要用到

### 1.2 配置解码器：

url：/config/modify/squeezelite

注意：用户能看到的是1.1中的key的列表，用户选中一个后出现两个按钮，两个按钮选项为DoP和Native，用户选择后根据请求参数传递过来。

请求参数：

output：截取1.1中value中类似” hw:CARD=Audio,DEV=0”这一段

dsd：只有两种选择：DoP、Native，注意区分大小写传递

响应示例：

{"status":200,"success":true}

### 1.3 无线网络列表：

url：/config/list/wifi

请求参数：无

响应示例：

{"data":["ChinaNet-nKzV","HW-LWG","xuyiqing","\u6781\u306e\u8336","CMCC-qL4a","ZCY","TP-LINK\_501","302\_5G","TP-LINK\_301","601","ChinaNet-CzUn","ChinaNet-sNYX","DIRECT-8a-BRAVIA"],"status":200,"success":true}

解释：data中每个字符串都是WIFI名称，注意把乱码（比如\u6781\u306e\u8336）解码成中文（Unicode编码解码，网上找写好的JS即可，不难）

### 1.4 检测当前连接的WIFI：

url：/config/check/wifi

请求参数：无

响应示例：

存在WIFI连接：{"ssid":"xuyiqing","status":200,"success":true}

不存在WIFI连接：{"status":200,"success":true}

解释：存在WIFI连接就会有ssid，这时候你在WIFI列表里面高亮显示已经连接上的WIFI即可，让用户知道他现在连接到的是哪个WIFI，这个接口用到的地方是：1.用户连接一个WIFI输入密码后，你间隔三秒后验证一下是否成功，给用户一个提示；2.用户已经连接成功后再次打开配置WIFI界面，高亮显示已经连接的WIFI。

### 1.5 断开WIFI连接：

url：/config/disc/wifi

请求参数：无

响应示例：无

解释：只要调用了断开连接接口，那么就会断开WIFI，然后机器会直接重启，用户等待启动好后，重新扫描IP即可。

### 1.6 连接WIFI：

url：/config/config/wifi

请求参数：

ssid：就是WIFI的名字，原样传过来，中文的直接传中文，不要编码

password：WIFI的密码

响应示例：

{"status":200,"success":true}

解释：无论密码是否正确，都会响应成功，验证是否真的成功是调用1.4的接口

### 1.7 重启：

url：/config/reboot

请求参数：无

响应示例：无

解释：机器直接重启了，没有相应，提示用户：请等待机器重启

### 1.8 关机：

url：/config/halt

请求参数：无

响应示例：无

解释：机器直接关机了，没有相应，提示用户：请等待机器重启（这个接口不要测试，机器关了有可能再打不开了）

### 1.9 挂载列表：

url：/config/mount/list

请求参数：无

响应示例：

{"data":["/dev/sda1 -> /mnt/music/sda1"],"status":200,"success":true}

解释：data的信息可能有多条，给用户按列表显示出来即可

### 1.10 挂载移动硬盘：

url：/config/mount/local

请求参数：无

响应示例：无

解释：挂载会直接重启，给用户一个提示，请等待重启

### 1.11 挂载网络硬盘

url：/config/mount/network

请求参数：

username：用户名

password：密码

path：挂载路径

注意：给用户三个input框输入即可，另外提醒用户，目标路径已经确定：/mnt/music/network

响应示例：无

解释：挂载会直接重启，给用户一个提示，请等待重启

### 1.12 卸载

url：/config/umount

请求参数：无

响应示例：无

解说：卸载后直接重启了，给用户一个提示即可

### 1.13 欧博音乐网桥服务状态检测

url：/config/status/squeezelite

请求参数：无

响应示例：

服务开启：{"active":true,"status":200,"success":true}

服务关闭：{"active":false,"status":200,"success":true}

### 1.14 开启欧博音乐网桥服务

url：/config/start/squeezelite

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功开启，都会响应成功，你可以调用1.13来验证

### 1.15 关闭欧博音乐网桥服务

url：/config/stop/squeezelite

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功关闭，都会响应成功，你可以调用1.13来验证

### 1.16 重启欧博音乐网桥服务

url：/config/restart/squeezelite

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功重启，都会响应成功，你可以调用1.13来验证

### 1.17 欧博音乐网桥服务开机是否自启动

url：/config/boot\_status/squeezelite

请求参数：无

响应示例：

开机自启动：{"enable":true,"status":200,"success":true}

开机不启动：{"enable":false,"status":200,"success":true}

### 1.18 设置欧博音乐网桥服务开机自启

url：/config/boot\_start/squeezelite

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功设置，都会响应成功，你可以调用1.17来验证

### 1.19 设置欧博音乐网桥服务开机不自启

url：/config/boot\_stop/squeezelite

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功设置，都会响应成功，你可以调用1.17来验证

### 1.20 RoonBridge服务状态检测

url：/config/status/roon

请求参数：无

响应示例：

服务开启：{"active":true,"status":200,"success":true}

服务关闭：{"active":false,"status":200,"success":true}

### 1.21 开启RoonBridge服务

url：/config/start/roon

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功开启，都会响应成功，你可以调用1.20来验证

### 1.22 关闭RoonBridge服务

url：/config/stop/roon

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功关闭，都会响应成功，你可以调用1.20来验证

### 1.23 重启RoonBridge服务

url：/config/restart/roon

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功重启，都会响应成功，你可以调用1.20来验证

### 1.24 RoonBridge服务开机是否自启动

url：/config/boot\_status/roon

请求参数：无

响应示例：

开机自启动：{"enable":true,"status":200,"success":true}

开机不启动：{"enable":false,"status":200,"success":true}

### 1.25 设置RoonBridge服务开机自启

url：/config/boot\_start/roon

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功设置，都会响应成功，你可以调用1.24来验证

### 1.26 设置RoonBridge服务开机不自启

url：/config/boot\_stop/roon

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功设置，都会响应成功，你可以调用1.24来验证

### 1.27 NAA服务状态检测

url：/config/status/naa

请求参数：无

响应示例：

服务开启：{"active":true,"status":200,"success":true}

服务关闭：{"active":false,"status":200,"success":true}

### 1.28 开启NAA服务

url：/config/start/naa

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功开启，都会响应成功，你可以调用1.27来验证

### 1.29 关闭NAA服务

url：/config/stop/naa

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功关闭，都会响应成功，你可以调用1.27来验证

### 1.30 重启NAA服务

url：/config/restart/naa

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功重启，都会响应成功，你可以调用1.27来验证

### 1.31 NAA服务开机是否自启动

url：/config/boot\_status/naa

请求参数：无

响应示例：

开机自启动：{"enable":true,"status":200,"success":true}

开机不启动：{"enable":false,"status":200,"success":true}

### 1.32 设置NAA服务开机自启

url：/config/boot\_start/naa

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功设置，都会响应成功，你可以调用1.31来验证

### 1.33 设置NAA服务开机不自启

url：/config/boot\_stop/naa

请求参数：无

响应示例：

{"status":200,"success":true}

注意：无论是否成功设置，都会响应成功，你可以调用1.31来验证

## 2 更新操作

### 2.1 更新列表：

url：/update/list

请求参数：无

响应示例：

{"data":["1.1","1.2"],"status":200,"success":true}

解释：data里面的1.1，1.2就是版本信息，用列表给用户展示出来

### 2.2 查看当前版本信息：

url：/update/current

请求参数：无

响应示例：

{"status":200,"success":true,"version":"1.0"}

解释：当前版本信息高亮显示在更新列表中即可

### 2.3 执行更新操作

url：/update/do

请求参数：

version：想要更新到的版本，用户在更新列表中选择版本后，把选择的版本号传过来即可

响应示例：无

解释：执行更新后机器直接重启了，没有相应，你那边需要给用户一个提示，版本号如果传错了，响应失败。

## 3 播放器

### 3.1 连接：

url：/player/connect

请求参数：

hostname：目标的IP地址

port：固定传给我9090即可

注意：这个IP地址就是扫描到的IP；注意，以下的所有操作都要基于连接，如果没有连接，那么无法访问以下所有接口，所以进入播放器页面的第一件事就是连接

响应示例：

{"server\_version":"7.9.2","status":200,"success":true}

解释：server\_version代表一个版本，无需关心，如果返回了它说明连接成功了

### 3.2 断开连接

url：/player/disconnect

请求参数：无

响应示例：

{"status":200,"success":true}

解释：不要轻易调用这个接口，因为断开后要重新连接，最好当作这个接口不存在

### 3.4 分页歌曲列表与查询

url：/player/list/songs

请求参数：

name：搜索名称，比如我要听《告白气球》，可以输入“告白”来搜索（注意，这个参数是可选的，如果不传，那么只是列表，不是搜索，后面也有类似的）

page\_size：分页大小，每页显示多少条数据，看情况来传，必须传的参数，后面有类似的不再详细介绍

page：页码，也就是第几页，从1开始，如果超出了则返回空列表，必须传的参数，后面有类似的不再详细介绍

响应示例：

{"status":200,"success":true,"tracks":[{"album":"\u4f9d\u7136\u8303\u7279\u897f","artist":"\u5468\u6770\u502b","bitrate":"893kbps VBR","duration":"256.373","id":"621","samplerate":"44100","samplesize":"16","title":"\u5343\u91cc\u4e4b\u5916","url":"file:///mnt/music/sda1/%E5%91%A8%E6%9D%B0%E4%BC%A6/%E5%8D%83%E9%87%8C%E4%B9%8B%E5%A4%96.flac"},{"album":"\u5468\u6770\u4f26\u7684\u5e8a\u8fb9\u6545\u4e8b","artist":"\u5468\u6770\u4f26","bitrate":"932kbps VBR","duration":"215.146","id":"624","samplerate":"44100","samplesize":"16","title":"\u544a\u767d\u6c14\u7403","url":"file:///mnt/music/sda1/%E5%91%A8%E6%9D%B0%E4%BC%A6/%E5%91%8A%E7%99%BD%E6%B0%94%E7%90%83.flac"},{"album":"\u65e0\u4e13\u8f91","artist":"\u6c6a\u82cf\u6cf7","bitrate":"912kbps VBR","duration":"196.885","id":"639","samplerate":"44100","samplesize":"16","title":"\u5c0f\u661f\u661f","url":"file:///mnt/music/sda1/%E5%96%9C%E7%88%B1/%E6%B1%AA%E8%8B%8F%E6%B3%B7-%E5%B0%8F%E6%98%9F%E6%98%9F.flac"}]}

注意：url你不用管，bitrate是比特率， duration是音乐时长，单位秒，samplerate是采样率，samplesize是抽样含量，里面的乱码是unicode，可以使用现成的JS转成中文，网上很多。

### 3.5 所有专辑信息

url：/player/list/albums

请求参数：

name：模糊搜索名称（可选，不传就不搜索）

page\_size：分页大小，每页显示多少条数据 ，必传

page：页码，也就是第几页，必传

响应示例：

{"albums":[{"id":"262","name":"\u5c71\u5ca1\u6643"},{"id":"273","name":"\u6c6a\u82cf\u6cf7"},{"id":"272","name":"\u6c6a\u82cf\u6cf7&By2"}],"status":200,"success":true}

解释：album是专辑名，id是专辑ID，注意把Unicode解码了。

### 3.6 所有艺术家信息

url：/player/list/artists

请求参数：

name：模糊搜索名称（可选，不传就不搜索）

page\_size：分页大小，每页显示多少条数据 ，必传

page：页码，也就是第几页，必传

响应示例：

{"artists":[{"id":"262","name":"\u5c71\u5ca1\u6643"},{"id":"273","name":"\u6c6a\u82cf\u6cf7"},{"id":"272","name":"\u6c6a\u82cf\u6cf7&By2"}],"status":200,"success":true}

解释：artist是艺术家名，id是艺术家ID，注意把Unicode解码了。

### 3.7 根据歌曲ID查询歌曲封面图片URL：

url：http://<server>:<port>/music/<track\_id>/cover.jpg

参数解释：

server就是扫描到的IP

port：固定9000

track\_id：歌曲ID

### 3.8 查看正在播放的歌曲的封面图片URL：

url：http://<server>:<port>/music/current/cover.jpg?player=<playerid>

参数解释：

server就是扫描到的IP

port：固定9000

playerid：播放器名称，暂时固定为Opera

### 3.9 播放器列表：

url：/player/list/players

请求参数：无

响应示例：

{"players":{“Opera”:” 04-D9-F5-64-8C-A4”},"status":200,"success":true}

解释：用户如果要进行播放等操作，必须要先选择播放器，选择播放器接口见3.8，Key-Value形式，Key为播放器名称，Value为播放器MAC地址，名称给用户看。

### 3.10 选择播放器：

url：/player/set/player

请求参数：

player：播放器名称

响应示例：

{"player\_name":”Opera”,"status":200,"success":true}

解释：如果有player\_name说明成功了

### 3.11 重新扫描曲目

url：/player/rescan

请求参数：无

响应示例：

{"status":200,"success":true}

解释：这个是服务器端重新扫描的接口，当服务器端扫描完后本地要重新调用；当用户重新挂载之后进行重新扫描

### 3.12 重新扫描进度

url：/playler/rescanprogress

请求参数：无

响应示例：

{"data":"0","status":200,"success":true}

解释：data如果是0代表服务器端扫描完成，如果是1代表服务器端正在扫描

### 3.13 播放

url：/player/play

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：播放器的播放开关，请求后开始播放音乐

### 3.14 暂停

url：/player/pause

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：播放器的播放开关，请求后暂停播放音乐

### 3.15 艺术家总数

url：/player/total/artists

请求参数：无

响应示例：

{"count":"17","status":200,"success":true}

解释：count就是艺术家总数

### 3.16 专辑总数

url：/player/total/albums

请求参数：无

响应示例：

{"count":"25","status":200,"success":true}

解释：count就是专辑总数

### 3.17 歌曲总数

url：/player/total/songs

请求参数：无

响应示例：

{"count":"71","status":200,"success":true}

解释：count就是歌曲总数

### 3.18 条件查询歌曲

url：/player/find/songs

请求参数：

track\_id：歌曲ID，用于精确查询某首歌

album：专辑完整的名称，用于查询某个专辑的歌曲

artist：艺术家完整的名称，用于查询某个艺术家的歌曲

注意：以上这三个参数只能同时传一个

page\_size：分页大小，每页显示多少条数据 ，必传

page：页码，也就是第几页，必传

响应示例：

### 3.19 根据歌曲ID播放

url：/player/play/song

请求参数：

track\_id：歌曲ID

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功播放，都会响应成功的。

### 3.20 当前播放歌曲的信息

url：/player/current/info

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{"data":{"album":"\u672a\u77e5","artist":"\u672a\u77e5","current\_time":"\u672a\u77e5","path":"/","player\_id":"0","title":"\u672a\u77e5","total\_time":"0"},"status":200,"success":true}

解释：album是专辑名，current\_time是当前播放的时间，path是歌曲的URL，player\_id是播放器ID，title是歌曲名，total\_time是歌曲总共的时间。这个接口的作用是当用户播放歌曲的时候，每隔几秒调用一次，用于同步歌曲播放进度条，利用当前时间和总时间做一个进度条。

### 3.21 设置音量

url：/player/set/volume

请求参数：

volume：音量，介于1-100，数字越大音量越大

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功设置，都会响应成功的。

### 3.22 获得当前音量

url：/player/get/volume

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{“volume”:100,status":200,"success":true}

解释：volume就是音量

### 3.23 播放列表添加歌曲

url：/player/playlist/add/song

请求参数：

track\_id：歌曲ID

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的，播放列表就是当前播放列表。

### 3.24 播放列表删除歌曲

url：/player/playlist/delete/song

请求参数：

track\_id：歌曲ID

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的，播放列表就是当前播放列表。

### 3.25 播放列表清空歌曲

url：/player/playlist/clear

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的，播放列表就是当前播放列表。

### 3.26 将某个专辑或艺术家添加到播放列表中

url：/player/playlist/add/album

请求参数：

album\_name：专辑名，可选参数

artist\_name：艺术家名，可选参数

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的，播放列表就是当前播放列表。

### 3.27 立即播放某个专辑或某个艺术家

url：/player/playlist/load/album

请求参数：

album\_name：专辑名，可选参数

artist\_name：艺术家名，可选参数

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的，播放列表就是当前播放列表。

### 3.28 播放下一首

url：/player/play/next

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的

### 3.29 播放上一首

url：/player/play/prev

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的

### 3.30 当前播放列表信息

url：/player/playlist/info

请求参数：无

注意：本接口必须要先调用3.10接口

响应示例：

待定，这里没有实际播放器，无法测试

### 3.31 查找所有播放文件夹

url：/player/folder/list

请求参数：

folder\_id：文件夹ID

响应示例：

什么参数都不传，那么就是根目录：

{"data":[{"filename":"network","id":"685","type":"folder"},{"filename":"sda1","id":"686","type":"folder"},{"filename":"sda2","id":"687","type":"folder"},{"filename":"sdb1","id":"688","type":"folder"},{"filename":"sdb2","id":"689","type":"folder"},{"filename":"sdc1","id":"690","type":"folder"},{"filename":"sdc2","id":"691","type":"folder"},{"filename":"sdd1","id":"692","type":"folder"},{"filename":"sdd2","id":"693","type":"folder"}],"status":200,"success":true}

传参数后，就是进入目录了：

{"data":[{"filename":"Taylor Swift - The Man.flac","id":"678","type":"track"},{"filename":"Taylor Swift - This Is Why We Can't Have Nice Things.flac","id":"679","type":"track"},{"filename":"Taylor Swift - You Need To Calm Down.flac","id":"680","type":"track"},{"filename":"Taylor Swift,Brendon Urie - ME! (Feat. Brendon Urie Of Panic! At The Disco).flac","id":"681","type":"track"},{"filename":"Taylor Swift,Dixie Chicks - Soon You\u2019ll Get Better.flac","id":"682","type":"track"},{"filename":"Taylor Swift,Ed Sheeran,Future - End Game.flac","id":"683","type":"track"}],"status":200,"success":true}

解释：filename就是文件或者文件夹名字，id是文件或文件夹名字，type的folder代表文件夹，track代表音乐文件

### 3.32 随机播放（直接替换当前播放列表）

url：/player/play/random

请求参数：

mode：tracks表示随机播放所有歌曲，albums表示随机播放所有专辑

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的

### 3.33 跳转播放进度

url：/player/play/seek

请求参数：

second：秒数，需要跳转到多少秒

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的

### 3.34 设置重复模式

url：/player/play/mode

请求参数：

mode：0表示播放一遍，1表示单曲循环，2表示循环播放列表

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的

### 3.35 获取本地电台

url：/player/radio/local/list

请求参数：无

注意：本接口必须要先调用3.10接口

响应结果：

{{"has\_items": "0","image": "/imageproxy/http%3A%2F%2Fcdn-radiotime-logos.tunein.com%2Fs116685q.png/image.png",

"is\_audio": "1","item\_id": "2213088a.0.23","name": "珠江经济广播电台 97.4 (经济)","type": "audio"}"status": 200,"success": true,"title": "中国"}

解释：

has\_items：如果是文件夹，应该拥有的子项（这里不会有文件夹，都是电台）

image：封面图片，访问9000端口拼接后即可获得

is\_audio：是否是电台，如果是文件夹就不是1（这里不会有文件夹，都是电台）

item\_id：播放歌曲需要传递的ID（注意：每次请求这个ID是不一样的）

name：电台名称

type：是文件夹还是电台

title:：是哪个国家的电台，目前是中国，如果换一个区域可能显示其他国家

### 3.36 播放本地电台

url：/player/radio/local/play

请求参数：

item\_id：3.35中的item\_id

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的

### 3.37 获取音乐电台

url：/player/radio/music/list

请求参数：

item\_id：文件夹ID，请注意不要错误传递为电台ID，不传递为返回顶级目录

注意：本接口必须要先调用3.10接口

响应示例：

{{"has\_items": "1","is\_audio": "0","item\_id": "8bd7f1c2.4","name": "嘻哈","type": "link"},”status”:200,”success”:true}

解释：参考3.35，这里有文件夹存在，本地电台的接口是没有文件夹存在。

### 3.38 播放音乐电台

url：/player/radio/music/play

请求参数：

item\_id：3.37的item\_id，注意不要错误传递为文件夹ID

注意：本接口必须要先调用3.10接口

响应示例：

{“status":200,"success":true}

解释：无论是否成功，都会响应成功的