Mulitiview Pro API 文档

Multiview Pro API 文档

序号	软件版本	API 版本	变更描述	修订人	日期	审核	日期
1	V _1.10	V _1.10	首次发行	陈利	2023-02-10		

1. 总体概述

1.1 HTTP API 介绍

NDI Multiview HTTP API 是为本型号网络解码和软件开发编程接口。NDI Multiview HTTP API 旨在通过简单的 HTTP 请求/响应机制，实现对设备的各种功能控制。软件开发者利用 NDI Multiview HTTP API，可以达到远程管理和控制设备的目的。API 的每一个功能，对应有一个 HTTP 请求路径。在当前版本的 API 中，这个路径的格式是：
本文档将列出每一个 API 功能对应的文件名(filename)及其功能描述。同时，在每个 API 文档中，会有一个测试接口，您可以根据请求参数的描述，在测试接口中填写您需要进行测试的请求参数，执行测试，可以在线验证 API 功能，十分方便。
API 的请求遵循标准的 HTTP 协议，开发人员可以使用任何符合 HTTP 标准的工具、软件开发库来完成 API 调用（例如 Web 浏览器，JavaScript 库，C/C++的 HTTP 协议库，JAVA，.NET 等）。参数的提交根据不同 API 的接口要求，可能为 POST 或 GET，或者两者均可，具体请参见 API 说明。API 的响应为 JSON 格式，JSON 对象中的每一个参数的意义，请参见每一个 API 的文档说明。
我们尽可能保持产品 API 接口的一致，但不可排除因产品特性差异而导致不同产品的 API 接口会存在部分差异。因此，请开发人员按照官方指导，区别和处理这些细节差异。

1.2 模块清单

序	模块	子模块

1	全局配置	①获取全局配置、②获取导航栏菜单
2	布局	①获取布局列表、 ②获取布局模板、 ③ 添加一个布局、 ④删除一个布局、 ⑤修改一个布局
3	窗口配置	①获取窗口列表、 ②新增一个窗口、 ③修改窗口名称、 ④关闭当前窗口、 ⑤获取屏幕列表、 ⑥获取当前窗口的位置、 ⑦设置当前窗口的位置
4	当个窗口的配置	①获取窗口信息、 ②设置窗口中每个窗格的参数、 ③移除当前窗口所有窗口的源、 ④获取当前窗口的工具栏参数、 ⑤设置当前窗口的工具栏、 ⑥获取当前窗口的边框参数、 ⑦设置当前窗口的边框参数、 ⑧获取窗口音频增益、 ⑨设置窗口音频增益
5	NDI 推流的配置	①获取 NDI 状态、 ②设置 NDI 状态、 ③获取 Take 配置参数、 ④设置 Take 配置参数、 ⑤应用 Take、 ⑥获取 NDI 输出配置参数、 ⑦提交 NDI 输出设置参数、 ⑧获取分辨率列表、 ⑨获取纵横比列表、 ⑩获取帧率列表
6	NDI 发现	①获取已选中的组 ②获取手动发现 IP 列表 ③提交 NDI 发现 配置
7	NDI 源	①获取 NDI 源列表 ②设置源是否预加载 ③通知后台打开网页 ④通知后台刷新 NDI 源
8	其他	①获取窗口的 SDP 信息

2. 基本规则

2.1 URL 规则

对于 HTTP 访问：

http://<host-ip>[:65532]/api/v2/<module-name>/<method-name>

HTTP 的服务端口默认为 81。<host-ip>是对应您要请求的设备的主机 IP 地址，这是必须的。<module-name> 是 HTTP API 对应的功能 module 名称，例如 config/general，discovery/scan...等等。详见文档中的具体描述。
<method-name>是对应 module 中的一个方法，例如：/discovery/scan/getGroups.json ，getGroups 是 scan 模块中的登录方法。

注：每一个 http 接口的请求标头里需要包含有名称为 app 的项，且其内容为必须包含有字段 username 和 token 的 json 数据结构，且用户名和令牌的值都是正确的才能正常调用接口。用户名和令牌通过登录接口获取。
例如：

```
app{
      "username":"admin",
      "alias":"admin",
      "changed":true,
      "role":"admin",
      "token":"c2c6c8e9ce9de77470e673c596eefcbe",
      "language":"zh",
      "software":"true"
      }
```

2.2 HTTP 响应和错误处理

除非设备的 Http server 在工作时出现了异常，否则设备总会给予您 HTTP 响应，同时在 HTTP Content 中包含一个 JSON object 描述对应 API 执行的结果。
如果设备未给您任何响应，这表示 Http server 出现了工作异常。请参考 HTTP 的错误代码、并进一步检测 HTTP 响应的错误消息获知详细的出错原因。
请注意：如果您的 HTTP API 请求地址正确，即使这个 API 执行失败，设备的 Http server 仍然会返回一个 JSON object 描述对应 API 执行的结果。您应该通过检查返回的 JSON object 中的字段来确定执行成功与否。
如果 HTTP API 执行成功，返回的 JSON object 格式如下：

```
{
    "result": "ok",
    "data": {
        ...
     }
}
```

执行失败时，返回的 JSON object 格式如下：

```
{
"result": "error",
   "msg": "..."
}
```

其中，**"result"** 是必定存在的字段，它的值**"ok"**表示请求的 HTTP API 执行成功。对于执行成功而言，**"msg"** 并非必定存在，您也可以完全忽略它的存在，它仅仅用于额外描述一些信息。

对于有返回结果的 HTTP API 来说，"data" 字段将会存在，而且它是一个 JSON object 或者 array，表示 API 的返回结果。这将视具体的 API 而定，请参考具体 API 的说明。
HTTP 响应的 Content-Type 必定为 application/json。
请注意：在本文档的后续 API 描述内容中，如非特殊情况的必要说明，将忽略关于 API 请求错误的说明。

2.3 CHARSET

目前版本的 HTTP API 仅仅支持 utf-8 字符集。

2.4 超时

在您处理 HTTP 请求/响应的超时设置时，请注意：

- 绝大多数的 API，从接受请求到完成执行，设备会在 <= 0.1s 时间内完成响应；但您需要根据实际的网络环境考虑传输延时等因素，合理设置超时值（一般来说，建议设置 5 秒超时或更高一些）。

3. 用户登录

说明：根据正确的用户名和密码登录设备并获取 token 值。

API URL

/api/v2/users/login.json

Request

Method：POST
格式（Example）：

```
{
    "username": "admin",
    "password": "Admin123"
}
```

Data 字段说明：

Field	Value	说明
username	[STRING]	用户名
password	[STRING]	密码

Response

格式（Example）：


```
{
    "data": {
        "alias": "admin",
        "changed": true,
        "role": "admin",
        "token": "8731f577dbc347671b708d41169413dd",
        "username": "admin"
    },
    "result": "ok"
}
```


Data 字段说明：

Field	Value	说明
alias	[STRING]	用户昵称
changed	[STRING]	token 是否发生更改
role	[STRING]	用户角色:管理员/普通用户
token	[STRING]	token 值
username	[STRING]	用户名

4. 全局配置

4.1 获取全局配置

说明：此处全局配置主要指的是关于设置界面上导航栏的一些配置，包括是否显示导航栏，语言切换，二维码地址等的设置。

API URL

/api/v2/config/general.json

Request

Method：GET

Response

格式（Example）：


```
{
  "result": "ok",
  "data": {
```

```
    "deviceName": "NDI Multiview",
    "languageShow": true,
    "websocketPort": 12345,
    "qrcode": "https://192.168.40.215:81/",
    "loginImgUrl": "./img/logo.png",
    "logoImgShow": true,
    "languageOptions": [
      {
        "value": "zh",
        "name": "简体中文",
        "default": true
      },
      {
        "value": "en",
        "name": "English"
      }
    ],
    "version": "V_1.10.0001"
  }
}
```


Data 字段说明：

Field	Value	说明
deviceName	[STRING]	设备名称
version	[STRING]	版本信息
logoImgShow	[BOOLEAN]	是否显示 logo
logoImgUrl	[STRING]	logo 路径
loginImgUrl	[STRING]	登录 logo 图标路径
qrcode	[STRING]	二维码地址
websocketPort	[INT]	websocket 端口
languageShow	[BOOLEAN]	是否显示语言切换图标
value	[STRING]	代表语言的字符串
name	[STRING]	语言名称
default	[BOOLEAN]	true 表示使用当前 Json Object 中的语言，false 或缺少这个字段则表示不使用此种语言

4.2 获取导航栏菜单

说明：主要是导航栏是否显示的设置

API URL

/api/v2/config/nav/nav

Request

Method：GET

Response

格式（Example）：

```
{
  "result": "ok",
  "data": {
    "main_nav": {
      "home": true
    }
  }
}
```

Data 字段说明：

Field	Value	说明
home	[BOOLEAN]	是否显示导航栏菜单

4.3 获取分辨率列表

说明：获取默认分辨率列表，包括常用的 720p，1080p，4k 等

API URL

/api/v2/resolution/list.json

Request

Method：GET

Response

格式（Example）：

```
{
  "result": "ok",
  "data": [
      {
        "value": 1,   // 分辨率参数值
        "label": "720p", // 前端显示
    "width": 1280, // 对应的宽度，用于前端切换 resolution 时绑值
        "height": 720
    },
    {
        "value": 2,
        "label": "1080p",
    "width": 1920,
        "height": 1080
    }
```

```
  ]
}
```

Data 字段说明：

Field	Value	说明
value	**[INT]**	分辨率列表 id
label	**[STRING]**	分辨率（用于显示）
width	**[INT]**	宽度
height	**[INT]**	高度

4.4 获取帧率列表

说明：获取默认帧率列表，包括常用的 60FPS，50FPS，30FPS 等

API URL

/api/v2/frameRate/list.json

Request

Method：GET

Response

格式（Example）：

```
{
  "result": "ok",
  "data": [ 60, 50, 40]
}
```

Data 字段说明：

Field	Value	说明
data	**[ARRAY]**	常用帧率列表

5. 制作

说明：每一个制作都是独立的，但共用发现列表，制作中有且仅有一个 PVW 和一个 PGM 窗口，制作名称不可重复。

5.1 获取分辨率列表

说明：主要是获取可选择的分辨率的列表。

API URL

/api/v2/resolution/list.json

Request

Method：GET

Response

格式（Example）：

```
{
   "result": "ok",
   "data": [
       {
         "id": 1,
         "label": "1280x720",
         "width": 1280,
         "height": 720
      },
      {
         "id": 2,
         "label": "1920x1080",
         "width": 1920,
         "height": 1080
      }
   ]
}
```

Data 字段说明：

Field	Value	说明
id	[INT]	代表此项分辨率的唯一标识
label	[STRING]	前端显示
width	[INT]	对应分辨率的宽度
height	[INT]	对应分辨率的高度

5.2 获取帧率列表

说明：获取默认可选的帧率项

API URL

/api/v2/frameRate/list.json

Request

Method：GET

Response

格式（Example）：

```
{
   "result": "ok",
   "data": [ 60, 50, 40]
}
```

Data 字段说明：

Field	Value	说明
data	[ARRAY]	提供的默认帧率列表

5.3 获取制作列表

说明：获取当前所有的制作项，包括制作 id，名称以及开关状态。

API URL

/api/v2/prrogram/list.json

Request

Method：GET

格式（Example）：

```
{
   "result": "ok",
   "data": [
     {
        "pro_id": "1", //制作 pro_id
        "name": "program A",
       "status": false,   // 当前制作状态为：开（true）/关(false)
     },
      {
       "pro_id": "2",
       "name": "programme B",
       "status": false,   // 当前制作状态为：开（true）/关(false)
     }
   ]
}
```

Data 字段说明：

Field	Value	说明
pro_id	[STRING]	制作 id
name	[STRING]	制作名称
status	[BOOLEAN]	当前制作状态为： 开（true） 关(false)

Response

格式（Example）：

```
{
"result":"ok"
}
```

5.4 修改制作名称

说明：可修改某一项制作的名称，当不可与其他制作名称重复，此名称同时被用于当前制作内窗口的 ndi 输出流的通道名称中。

API URL

/api/v2/program/rername.json

Request

Method：POST

格式（Example）：

```
{
    "pro_id": "1",
    "name": "programma A"
}
```

Data 字段说明：

Field	Value	说明
pro_id	[STRING]	布局 id
name	[STRING]	新的制作名称

Response

格式（Example）：

```
{
    "result":"ok"
}
```

5.5 添加一个制作

说明：添加一项制作，可设置初始名称，分辨率以及帧率值，制作的分辨率和帧率值仅应用于当前制作内的 PVW 和 PGM 窗口，其他普通窗口可单独设置。

API URL

/api/v2/program/add.json

Request

Method：POST

格式（Example）：

```
{
    "name": "programma A",
    "resolution": {
    "width": 1920,
      "height": 1080,
      "index": 2,
      "aspect": 1.777
    },
    "frameRate": 60
}
```

Data 字段说明：

Field	Value	说明
name	[STRING]	制作名称
width	[INT]	分辨率的宽度
height	[INT]	分辨率的高度
index	[INT]	分辨率的索引值
aspect	[FLOAT]	分辨率宽高比

Response

格式（Example）：

```
{
"result":"ok"
}
```

5.6 删除一个制作

说明：删除某一个制作，即删除当前制作下的所有窗口。

API URL

/api/v2/program/remove.json

Request

Method：POST

格式（Example）：

```
{
  "pro_id":"1"
}
```

Data 字段说明：

Field	Value	说明

pro_id	[INT]	制作 id

Response

格式（Example）：

```
{
"result":"ok"
}
```

5.7 开启/关闭一个制作

说明：开启/关闭制作即开启/停止当前制作内所有窗口的渲染。

API URL

/api/v2/program/set_status.json

Request

Method：POST

格式（Example）：

```
{
   "pro_id":"1",
   "status": true
}
```

Data 字段说明：

Field	Value	说明
pro_id	[INT]	制作 id
status	[BOOLEAN]	状态值： 开（true） 关（false）

Response

格式（Example）：

```
{
"result":"ok"
}
```

5.8 切换制作

说明：在多项制作间进行切换。

API URL

/api/v2/program/switch.json

Request

Method：POST

格式（Example）：

```
{
   "pro_id":"1"
}
```

Data 字段说明：

Field	Value	说明
pro_id	[INT]	制作 id

Response

格式（Example）：

```
{
   "result":"ok"
}
```

5.9 获取制作参数

说明：获取当前制作的名称，分辨率和帧率值，此处设置的分辨率和帧率仅应用于当前制作内的 PVW 和 PGM 窗口。

API URL

/api/v2/program/get_param.json

Request

Method：GET

格式（Example）：

```
/api/v2/prgram/get_param.json?pro_id=1
```

Response

格式（Example）：

```
{
"data":{
 "name": "programma A",
      "resolution": {
         "width": 1920,
            "height": 1080,
```

```
            "index": 2,
          "aspect": 1.777
        },
      "frameRate": 60
},
"result": "ok"
}
```

Data 字段说明：

Field	Value	说明
name	[STRING]	制作名称
width	[INT]	分辨率的宽度
height	[INT]	分辨率的高度
index	[INT]	分辨率的索引值
aspect	[FLOAT]	分辨率宽高比
frameRate	[FLOAT]	帧率值

5.10 修改制作参数

说明：修改当前制作的名称，分辨率或帧率值，此处设置的分辨率和帧率仅应用于当前制作内的 PVW 和 PGM 窗口。

API URL

/api/v2/program/set_param.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
 "name": "programma A",
      "resolution": {
         "width": 1920,
            "height": 1080,
            "index": 2,
            "aspect": 1.777
       },
      "frameRate": 60
}
```

Data 字段说明：

Field	Value	说明
pro_id	[STRING]	制作 id
name	[STRING]	制作名称
width	[INT]	分辨率的宽度

height	[INT]	分辨率的高度
index	[INT]	分辨率的索引值
aspect	[FLOAT]	分辨率宽高比
frameRate	[FLOAT]	帧率值

Response

格式（Example）：

```
{
   "result":"ok"
}
```

5.11 语言切换

说明：目前仅支持中英文的语言切换。

API URL

/api/v2/lang/modify.json

Request

Method：POST

格式（Example）：

```
{
   "lang":"en"
 }
```

Data 字段说明：

Field	Value	说明
lang	[STRING]	切换的语言： 中文（zh） 英文（en）

Response

格式（Example）：

```
{
   "result":"ok"
}
```

6. 源发现

说明：主要包括自动和手动发现源列表的添加，删除，修改操作。

6.1 获取源发现列表

API URL

/api/v2/source/list.json

Response

Method：GET
格式（Example）：


```
{
"result": "ok",
"data": [
{
        "disc_id": "1",
   "disc_name": "自动发现 1",
        "sources": [
        {
              "id": "KILOVIEW (Chan 1  涔辩爜)",
              "name": "KILOVIEW (Chan 1  涔辩爜)",
              "ip": "192.168.2.33",
              "url": "192.168.2.33:8080",
              "web_url": ""
           },
           {
              "id": "N4-19162010064 (Channel-1)",
              "name": "N4-19162010064 (Channel-1)",
              "ip": "192.168.2.33",
              "url": "" ,
              "web_url": ""
           }
  ]
},
      {
   "disc_id": "2",
   "disc_name": "手动发现 2",
   "sources": [
    {
        "id": "KILOVIEW (Chan 1  涔辩爜)",
         "name": "KILOVIEW (Chan 1  涔辩爜)",
        "ip": "192.168.2.33",
        "url": "192.168.2.33:8080",
                  "web_url": ""
     },
     {
       "id": "N4-19162010064 (Channel-1)",
       "name": "N4-19162010064 (Channel-1)",
       "ip": "192.168.2.33",
       "url": "",
                  "web_url": ""
     }
  ]
```

```
    }
  ]
}
```

Data 字段说明：

Field	Value	说明
disc_id	[STRING]	源发现的 id
disc_name	[STRING]	源发现的名称
id	[STRING]	源发现中某一项源的 id
name	[STRING]	源发现中某一项源的名称
ip	[STRING]	源发现中某一项源的地址
url	[STRING]	源发现中某一项源的地址与端口
web_url	[STRING]	源发现中某一项源的网页链接地址

6.2 添加一项分组

说明：添加一项分组，名称不可与其他分组重复。

API URL

/api/v2/source/add.json

Request

Method：POST

格式（Example）：

自动发现

```
{
"disc_name": "自动发现 1",
   "disc_type": "Auto",
   "discovery_enable": true,
   "discovery_ip": "192.168.1.1",
   "group": ["kiloview", "NDI","test"]
}
```

手动发现

```
{
"disc_name": "手动添加 1",
   "disc_type": "Manual",
   "group": ["kiloview", "NDI","test"],
   "ip": ["192.168.40.11", "192.168.40.12"]
}
```

Data 字段说明：

Field	Value	说明
disc_name	[STRING]	源发现名称
disc_type	[STRING]	源发现类型：分为自动和手动两种类型： 自动（Auto） 手动（Manual）
discovery_enable	[BOOLEAN]	是否开启发现服务器
discovery_ip	[STRING]	发现服务器地址
group	[ARRAY]	发现组字符串列表
ip	[ARRAY]	ip 字符串列表

Response

格式（Example）：

```
{
   "result":"ok"
}
```

6.3 修改分组参数

说明：修改当前源发现的名称，类型，发现服务器，发现组列表或者 ip 列表。

API URL

/api/v2/source/modify.json

Request

Method：POST

格式（Example）：

自动发现

```
{
"disc_id": "1",
"disc_name": "自动发现 1",
   "disc_type": "Auto",
   "discovery_enable": true,
   "discovery_ip": "192.168.1.1",
   "group": ["kiloview", "NDI","test"]
}
```

自动发现

```
{
"disc_id": "1",
"disc_name": "手动发现 1",
   "disc_type": "Manual",
```

```
   "group": ["kiloview", "NDI","test"]，
   "ip": ["192.168.40.11", "192.168.40.12"]
}
```

Response

格式（Example）：

```
{
   "result":"ok"
}
```

5.4 获取分组参数

说明：获取当前分组的参数。

API URL

/api/v2/source/get_param.json

Request

Method：GET

格式（Example）：

```
/api/v2/source/get_param.json?disc_id=1
```

Response

格式（Example）：

自动发现

```
{
"data":{
"disc_id": "1",
"disc_name": "自动添加 1",
   "disc_type": "Manual" ,
   "group": ["kiloview", "NDI","test"],
   "ip": ["192.168.40.11", "192.168.40.12"]
   },
"result": "ok"
}
```

手动发现

```
{
"data":{
"disc_id": "1",
"disc_name": "手动添加 1",
   "disc_type": "Auto" ,
   "discovery_enable": true,
```

```
   "discovery_ip": "192.168.1.1",
   "group": ["kiloview", "NDI","test"]
   },
"result": "ok"
}
```

5.5 删除一项分组

API URL

/api/v2/source/remove.json

Request

Method：POST

格式（Example）：

```
{
      "disc_id": "1"
}
```

Field	Value	说明
disc_id	**[STRING]**	源发现 id

Response

格式（Example）：

```
{
   "result":"ok"
}
```

5.6 打开 ndi 源的网页地址

API URL

/api/v2/webSite/set.json

Request

Method：POST

格式（Example）：

```
{
   url:"http://192.168.40.9/"
}
```

Data 字段说明：

Field	Value	说明
url	**[STRING]**	NDI 源网址

Response

格式（Example）：

```
{
"result":"ok"
}
```

5.7 刷新发现列表

API URL

/api/v2/source/refresh.json

Request

Method：POST

Response

格式（Example）：

```
{
   "refresh":true
}
```

Data 字段说明：

Field	Value	说明
refresh	**[BOOLEAN]**	是否刷新一次发现列表

Response

格式（Example）：

```
{
"result":"ok"
}
```

6. 窗口

6.1 获取当前制作的窗口列表

API URL

/api/v2/window/list.json

Request

Method：GET/POST
格式（Example）：

```
/api/v2/window/list.json?pro_id=1
```

Response

格式（Example）：

```
{
   "result": "ok",
   "data": [
     {
       "id": "1",
       "name": "Main",
        "layout_id": "1",
        "modified": false,
        "resolution": {
          "width": 1920,
             "height": 1080,
             "index": 2,
             "aspect": 1.777
        },
        "frameRate": 60,
     },
     {
       "id": "pvw",
       "name": "PVW",
        "layout_id": "1"
   "modified": false,
        "resolution": {
        "width": 1920,
           "height": 1080,
           "index": 2,
           "aspect": 1.777
       },
       "frameRate": 60
     },
     {
       "id": "pgm",
       "name": "PGM",
        "layout_id": "1",
```

```
        "modified": false,
         "resolution": {
         "width": 1920,
            "height": 1080,
            "index": 2,
            "aspect": 1.777
        },
       "frameRate": 60
      }
   ]
}
```

Data 字段说明：

Field	Value	说明
id	**[STRING]**	窗口 id
name	**[STRING]**	窗口名称
layout_id	**[STRING]**	窗口使用的预设 id
modified	**[BOOLEAN]**	窗口中预设是否有更改
width	[INT]	窗口分辨率的宽度
height	[INT]	窗口分辨率的高度
index	[INT]	窗口分辨率的索引值
aspect	[FLOAT]	窗口分辨率宽高比
frameRate	[FLOAT]	窗口帧率值

6.2 切换窗口

说明：在当前制作有多个窗口的情况下，可进行不同窗口间的切换操作。

API URL

/api/v2/window/switch.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
      "win_id": "2",
      "session_id": "1"
}
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
win_id	**[STRING]**	窗口 id
session_id	**[STRING]**	webrtc 的会话 id

Response

格式（Example）：

```
{
"result":"ok"
}
```

6.3 添加窗口

API URL

/api/v2/window/add.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
   "win_id": "1",
   "name": "window_1",
   "resolution": {
      "width": 1920,
         "height": 1080,
         "index": 2,
         "aspect": 1.777
      },
    "frameRate": 60.0,
    "number_used": true,
    "number_range": {
   "min": 1,
      "max": 16
   }
}
```

Data 字段说明：

Field	Value	说明
pro_id	[STRING]	制作 id
win_id	[STRING]	窗口 id
name	[STRING]	窗口名称
frameRate	[FLOAT]	窗口帧率
width	[INT]	窗口分辨率的宽度
height	[INT]	窗口分辨率的高度
index	[INT]	窗口分辨率的索引值
aspect	[FLOAT]	窗口分辨率宽高比

number_used	[BOOLEAN]	是否开启编号
min	[INT]	最小编号值
max	[INT]	最大编号值

Response

格式（Example）：

```
{
   "result":"ok"
}
```

6.4 修改窗口配置参数

说明：修改窗口的名称，分辨率和帧率。

API URL

/api/v2/window/set_param.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
   "win_id": "1",
   "name": "window_1",
   "resolution": {
      "width": 1920,
         "height": 1080,
         "index": 2,
         "aspect": 1.777
      },
    "frameRate": 60,
    "number_used": true,
    "number_range": {
   "min": 1,
      "max": 16
   }
}
```

Data 字段说明：

参见上面 6.3.

Response

格式（Example）：

```
{
"result":"ok"
}
```

6.5 删除窗口

说明：删除某个制作中的某个窗口。

API URL

/api/v2/window/remove.json

Request

Method：POST
格式（Example）：

```
{
"pro_id": "1",
  "win_id": "1"
}
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
win_id	**[STRING]**	窗口 id

Response

格式（Example）：

```
{
"result":"ok"
}
```

6.6 获取当前编码输出列表

说明：获取当前可用的编码方式的列表。

API URL

/api/v2/window/get_output_list.json

Request

Method：GET
格式（Example）：

```
/window/get_output_list.json
```

Response

格式（Example）：

```
{
"result": "ok",
   "data": [
   {
      "id": 1,
         "label": "NDI"
      }
   ]
}
```

6.7 获取可用屏幕列表

说明：获取当前服务器已连接的所有显示器列表。

API URL

/api/v2/window/get_screen_list.json

Request

Method：GET

格式（Example）：

```
/window/get_screen_list.json
```

Response

格式（Example）：

```
{
 "result":"ok",
  "data":[
     {
        "id":1,
        "label":"Screen 1"
     },
     {
        "id":2,
        "label":"Screen 2"
     },
     {
        "id":3,
        "label":"Screen 3"
     },
     {
        "id":4,
        "label":"Screen 4"
     }
```

```
  ]
}
```

Data 字段说明：

Field	Value	说明
id	**[INT]**	屏幕 id
label	**[STRING]**	用于前端显示的标签

6.8 获取窗口内容

说明：获取窗口的所有设置参数以及加载的预设的参数。

API URL

/api/v2/window/get.json

Request

Method：GET

格式（Example）：

```
/api/v2/window/get.json?pro_id=1&win_id=1
```

Response

格式（Example）：

```
{
"result": "ok",
      "data": {
      "layout_id": "3",
      "layout_count": 1,
            "layout_name": "4",
            "resolution": {
              "aspect": 0,
              "height": 1080,
              "index": 2,
              "width": 1920
            }
            "layout": [
                  {
                  "id": 1,
                        "number": 1,
                        "x": 0,
                        "y": 0,
                        "z_index": 0
                        "width": 100,
                        "height": 100,
                        "type": "",
                        "aspectRatio": "",
                        "frameRate": 0,
```

```
                    "stream_id": "",
                    "stream_name": "",
                    "pattern_name": "Camera_1",
                    "audio_mix": {
                        "mix_type": "close",
                        "pgm_enable": false,
                        "pvw_enable": false
                    },
                }
            ],
        }
}
```


Data 字段说明：

Field	Value	说明
layout_id	**[STRING]**	预设 id
layout_count	**[INT]**	预设中的窗格数量
layout_name	**[STRING]**	预设名称
id	**[STRING]**	窗格 id
number	**[STRING]**	窗格编号
x	**[FLOAT]**	窗格的 x 位置
y	**[FLOAT]**	窗格的 y 位置
z_index	**[FLOAT]**	窗格的 z 位置，即叠层层级
width	**[FLOAT]**	窗格的宽度
height	**[FLOAT]**	窗格的高度
type	**[STRING]**	窗格的类型：分为三种， 普通窗格 type="" pvw 窗格 type="pvw" pgm 窗格 type="pgm"
aspectRatio	**[STRING]**	窗格中的视频源分辨率
frameRate	**[FLOAT]**	窗格中的视频源帧率
stream_id	**[STRING]**	窗格中的视频源 id
stream_name	**[STRING]**	窗格中的视频源名称
pattern_name	**[STRING]**	窗格名称，可自定义
mix_type	**[STRING]**	窗格中视频源的音频设置：分为三种， 静音（close） 锁定（always） 跟随（follow）
pgm_enable	**[BOOLEAN]**	当前视频源是否被设置到 pgm 窗口中
pvw_enable	**[BOOLEAN]**	当前视频源是否被设置到 pvw 窗口中

6.9 获取窗口在屏幕上的显示状态

说明：获取当前窗口是否显示以及显示在哪一个屏幕上。

API URL

/api/v2/window/get_position.json

Request

Method：GET

格式（Example）：

```
/api/v2/window/get_position.json?pro_id=1&win_id=1
```

Response

格式（Example）：

```
{
   "result":"ok",
   "data":{
      "select": 1
   }
}
```

Data 字段说明：

Field	Value	说明
select	**[INT]**	当前窗口所在屏幕的索引，select <= 0 时关闭屏幕输出

6.10 设置窗口输出到屏幕

说明：设置当前窗口在哪一个屏幕上显示。

API URL

/api/v2/window/set_position.json

Request

Method：POST

格式（Example）：

```
{
 "pro_id": "1",
    "win_id":"1",
    "select":1
 }
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
win_id	**[STRING]**	窗口 id
select	**[INT]**	屏幕索引值（即选择第几个屏幕

		作为输出）

Response

格式（Example）：

```
{
"result": "ok"
}
```

6.11 获取窗口编码输出状态

说明：获取当前窗口是否开启 ndi 编码输出。

API URL

/api/v2/window/get_output_type.json

Request

Method：GET

格式（Example）：

```
/api/v2/window/get_output_type.json?pro_id=1&win_id=1
```

Response

格式（Example）：

```
{
"result":"ok",
"data":{
        "select": "NDI"
}
}
```

Data 字段说明：

Field	Value	说明
select	**[STRING]**	编码输出类型选择， select=""时表示关闭编码输出； select="NDI"时表示开启 ndi 推流。

6.12 设置编码输出

说明：主要是指开启/关闭 NDI 推流的设置。

API URL

/api/v2/window/set_output_type.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
    "win_id":"1",
    "select": "NDI"
}
```

Response

格式（Example）：

```
{
     "result":"ok"
}
```

6.13 设置 NDI 编码输出参数

说明：设置 NDI 编码输出方式，分为三种：默认，禁用 tcp，组播。

API URL

/api/v2/window/ndi/set_param.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
      "win_id": "1",
      "connection_type": "Default",
      "netprefix": "",
      "netmask": "",
      "ttl": 1,
      "group": ["public"],
      "discovery_enable": true,
      "discovery_ip": "192.168.1.1"
}
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
win_id	**[STRING]**	窗口 id
connection_type	**[STRING]**	传输方式：分为三种 默认（Default）； 禁用多 tcp（no_multi_tcp）；

		组播（multicast）
netprefix	**[STRING]**	组播地址
netmask	**[STRING]**	子网掩码
ttl	**[INT]**	TTL
group	**[ARRAY]**	添加的组名称列表
discovery_enable	**[BOOLEAN]**	是否开启发现服务器
discovery_ip	**[STRING]**	发现服务器地址

Response

格式（Example）：

```
{
"result":"ok"
}
```

6.14 获取 NDI 编码参数

API URL

/api/v2/window/ndi/get_param.json

Request

Method：GET

格式（Example）：

```
/api/v2/window/ndi/get_param.json?pro_id=1&win_id=1
```

Response

格式（Example）：

```
{
"result": "ok",
"data":{
"pro_id": "1",
      "win_id": "1",
      "connection_type": "Default",
      "netprefix": "",
      "netmask": "",
      "ttl": 1,
      "group": ["public"],
      "discovery_enable": true,
      "discovery_ip": "192.168.1.1"
      }
}
```

6.15 删除窗口中所有视频源

API URL

/api/v2/window/delete_sources.json

Request

Method：POST

格式（Example）：

```
{
    "pro_id": "1"
    "win_id": "1" //窗口 id
}
```

Response

格式（Example）：

```
{
  "result":"ok"
}
```

6.16 获取工具栏状态

说明：获取工具栏的音柱，安全框，中心线，名称显示，字体，边框，音量增益的状态。

API URL

/api/v2/window/get_state_control.json

Request

Method：GET

格式（Example）：

```
/api/v2/window/get_state_control?pro_id=1&win_id=1
```

Response

格式（Example）：

```
{
"data"{
 "name_enable": false,
      "safe_area": false,
      "center_cross": false,
      "vu_meter": false,
      "font": {
         "color": "#ddd",
```

```
        "size": 12
      },
      "border": {
        "enable": false,
        "color": "#fff"
      },
      "audio_gain": 10
},
"result": "ok"
}
```

Data 字段说明：

Field	Value	说明
name_enable	**[BOOLEAN]**	是否显示名称
safe_area	**[BOOLEAN]**	是否显示安全框
center_cross	**[BOOLEAN]**	是否显示中心线
vu_meter	**[BOOLEAN]**	是否显示音柱
audio_gain	**[INT]**	音量增益值
color	**[STRING]**	字体颜色/边框颜色
size	**[INT]**	字体大小
enable	**[BOOLEAN]**	是否显示边框

6.17 设置工具栏状态

说明：设置工具栏的音柱，安全框，中心线，名称显示，字体，边框，音量增益的状态/大小。

API URL

/api/v2/window/set_state_control.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
"win_id": "1",
"type":"font",//类型:name_enable,safe_area,center_cross,vu_meter,border,audio_gain 等
"config": {
"color":"#fff",
"size":12,
"enable":false/true,
"value": 10 (音频增益值)
} //根据需要增加或删减 config 中字段
```

```
}
```

Response

格式（Example）：

```
{
"result": "ok"
}
```

6.18 设置窗口预设

说明：设置当前窗口的预设。

API URL

/api/v2/window/set_layout.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
   "win_id": "1",
   "layout_id": "1"
}
```

Response

格式（Example）：

```
{
"result": "ok"
}
```

6.19 重新加载窗口预设

说明：重新初始化当前窗口的内容为当前所选预设模板。

API URL

/api/v2/window/reload_layout.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
   "win_id": "1",
   "layout_id": "1"
}
```

Response

格式（Example）：

```
{
"result": "ok"
}
```

6.20 设置窗格中的视频源

说明：选择并设置 NDI 视频源到选中窗格中。

API URL

/api/v2/window/set_source.json

Request

Method：POST

格式（Example）：

```
{
   "from": {
   "pro_id": "1",
   "win_id": "1",
      "pos_id": 1
   },
   "to": {
   "pro_id": "1",
   "win_id": "2",
      "pos_id": 2,
      "stream_id": ""
   }
}
```

Data 字段说明：

Field	Value	说明
from：pro_id	**[STRING]**	视频源来源的制作 id
from:　win_id	**[STRING]**	视频源来源的窗口 id
from：pos_id	**[STRING]**	视频源来源的窗格 id

to: pro_id	**[STRING]**	视频源去向的制作 id
to: win_id	**[STRING]**	视频源去向的窗口 id
to：pos_id	**[STRING]**	视频源去向的窗格 id
to: stream_id	**[STRING]**	视频源名称

Response

格式（Example）：

```
{
"result": "ok"
}
```

6.21 删除窗格中的视频源

说明：删除选中窗格中的视频源。

API URL

/api/v2/window/remove_source.json

Request

Method：POST

格式（Example）：

```
{
   "pro_id": "1",
   "win_id": "2",
      "pos_id": 2
}
```

Response

格式（Example）：

```
{
"result": "ok"
}
```

6.22 获取单个窗格的高级参数

说明：删除选中窗格中的视频源。

API URL

/api/v2/window/get_single_layout.json

Request

Method：POST

格式（Example）：

```
/window/get_single_layout.json?pro_id=1&win_id=1&pos_id=1
```

Response

格式（Example）：

```
{
   "result": "ok",
   "data": {
      "pattern_name": "",
      "safe_area": true,
      "vu_meter": true,
      "center_cross": false,
       "name_enable": false
   }
}
```

Data 字段说明：

Field	Value	说明
pattern_name	**[STRING]**	窗格别名
safe_area	**[BOOLEAN]**	是否开启安全框
vu_meter	**[BOOLEAN]**	是否开启音柱
center_cross	**[BOOLEAN]**	是否开启中心线
name_enable	**[BOOLEAN]**	是否开启名称显示

6.23 获取单个窗格的高级参数

说明：删除选中窗格中的视频源。

API URL

/api/v2/window/set_single_layout.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
```

```
  "win_id": "1",
  "pos_id": 1,
  "type": "name_enable", //name_enable,safe_area,center_cross,vu_meter,audio_enable,name,
  "config": {
  "enable": true/false,
    "pattern_name": "标题名称"
  }
}
```

Response

格式（Example）：

```
{
  "result": "ok",
}
```

6.24 设置混音输出状态

说明：设置某个窗格的混音输出状态为锁定/跟随/关闭。

API URL

/api/v2/window/set_audio_type.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
  "win_id": "1",
  "pos_id": 1,
  "mix_type": "always"/"follow"/"close"
}
```

Response

格式（Example）：

```
{
  "result": "ok",
}
```

7. 预设

说明：主要包括预设布局的增加，删除，修改，保存，重命名等。

7.1 获取预设模板列表

API URL

/api/v2/layout/template.json

Request

Method：GET

Response

格式（Example）：

```
{
"result": "ok",
   "count": 1000,
   "data": [
     {
        "template_id": "1", //固定布局模板的 id
        "template_count": 2,
       "name": "picInPic",
       "position": [
         {
           "y": 0,
           "x": 0,
           "height": 100,
           "width": 100,
              "type": "pvw"/"pgm"/"",
              "number": "1",
              "id": 1,
              "z_index": 0
         },
         {
           "y": 69,
           "x": 69,
           "height": 30,
           "width": 30,
              "type": "pvw"/"pgm"/"",
              "number": "2", //编号
              "id": 2, //唯一标识
              "z_index": 0 //层级：从 0 开始
              }
       ],
     },
   ]
}
```

Data 字段说明：

Field	Value	说明
template_id	**[STRING]**	模板 id
template_count	**[INT]**	模板中窗格个数

name	**[STRING]**	模板名称
id	**[INT]**	窗格 id
x	**[INT]**	窗格坐标 x 值
y	**[INT]**	窗格坐标 y 值
width	**[INT]**	窗格宽度
height	**[INT]**	窗格高度
z_index	**[INT]**	窗格层级
type	**[STRING]**	窗格类型
number	**[STRING]**	窗格编号

7.2 获取预设列表

说明：获取当前指定制作的预设列表。

API URL

/api/v2/layout/list.json

Request

Method：GET

格式（Example）：

```
/api/v2/layout/list.json?pro_id=1
```

Response

格式（Example）：


```
{
  "result": "ok",
  "data": [
    {
      "layout_id": "1",
      "layout_count": 2,
      "name": "画中画",
       "used": true,
      "position": [
        {
          "x": 0,
          "y": 0,
          "width": 100,
          "height": 100,
             "type": "",
              "number": "1",
             "id": 1,
             "z_index": 0
          },
        {
          "x": 50,
          "y": 50,
          "width": 48,
```

```
            "height": 48,
                "type": "",
                "number": "2",
                "id": 2,
                "z_index": 0
            }
        ]
    },
    ]
}
```


Data 字段说明：

Field	Value	说明
layout_id	**[STRING]**	预设 id
layout_count	**[INT]**	预设中窗格个数
name	**[STRING]**	预设名称
used	**[BOOLEAN]**	当前预设是否正在使用中
id	**[INT]**	窗格 id
x	**[INT]**	窗格坐标 x 值
y	**[INT]**	窗格坐标 y 值
width	**[INT]**	窗格宽度
height	**[INT]**	窗格高度
z_index	**[INT]**	窗格层级
type	**[STRING]**	窗格类型
number	**[STRING]**	窗格编号

7.3 添加预设

说明：在指定制作的预设列表中添加一项新的预设。

API URL

/api/v2/layout/add.json

Request

Method：POST

格式（Example）：


```
{
     "pro_id": "1",
    "name": "123",
     "template_id": "1"
}
```


Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
name	**[STRING]**	预设名称

template_id	**[STRING]**	预设模板 id

Response

格式（Example）：

```
{
   "result": "ok",
}
```

7.4 预设重命名

API URL

/api/v2/layout/rename.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
   "layout_id": "1",
   "name": ""
}
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
layout_id	**[STRING]**	预设 id
name	**[STRING]**	预设名称

Response

格式（Example）：

```
{
   "result": "ok",
}
```

7.5 编辑布局

API URL

/api/v2/layout/modify.json

Request

Method：POST

格式（Example）：

```
{
      "pro_id": "1",
     "win_id": "1",
      "layout_id": "1",
      "layout_count": 2,
     "name": "123",
     "position": [
         {
              "y": 0,
              "x": 0,
              "height": 100,
              "width": 100 ,
                  "type": ""/"pvw"/"pgm",
                    "number": "1",
               "id": 1,
               "z_index": 0
           },
         {
              "y": 69,
              "x": 69,
              "height": 30,
              "width": 30,
                  "type": ""/"pvw"/"pgm",
                     "number": "2",
               "id": 2,
               "z_index": 0
         }
     ]
}
```

Response

格式（Example）：

```
{
   "result": "ok",
}
```

7.6 删除预设

API URL

/api/v2/layout/remove.json

Request

Method：POST

格式（Example）：

```
{
   "pro_id": "1",
```

```
  "layout_id":"2"
}
```

Response

格式（Example）：

```
{
  "result": "ok",
}
```

7.7 当前预设另存为

API URL

/api/v2/layout/resave.json

Request

Method：POST

格式（Example）：

```
{
    "pro_id": "1",
    "layout_id": "1",
    "name": ""
}
```

Response

格式（Example）：

```
{
  "result": "ok",
}
```

8. NDI 发现

说明：主要包括 NDI 源自动发现列表的获取，刷新，添加跨网段视频源等

8.1 获取已选中的组

说明：获取当前已添加的 NDI 的组名称列表

API URL

/api/discovery/scan/groupChecked.json

Request

Method：GET/POST

Response

格式（Example）：

```
{
"result": "ok",
"data": ["kiloview", "NDI","test"]
}
```

8.2 获取手动发现 IP 列表

说明：获取当前已添加的跨网段的 ip 列表

API URL

/api/discovery/scan/getManualIps.json

Request

Method：GET/POST

Response

格式（Example）：

```
{
   "result": "ok",
   "data":["192.168.2.52","192.168.2.33"]
}
```

8.3 提交 NDI 发现配置

说明：在设置完 NDI 组以及跨网段 ip 地址列表后，通过以下请求即可提交进行 NDI 源发现的设置

API URL

/api/discovery/scan/addManualIp.json

Request

Method：GET/POST

格式（Example）：

```
{
   "group_name": [
```

```
      "kiloview",
"newtek"
   ],
   "ip": [
      "192.168.2.52",
      "192.168.2.33"
   ]
}
```

Data 字段说明：

Field	Value	说明
group_name	**[ARRAY]**	NDI 组名列表
ip	**[ARRAY]**	NDI 源跨网段 ip 地址列表

Response

格式（Example）：

```
{
"result":"ok"
}
```

9. 其他

9.1 获取窗口的 SDP 信息

API URL

/api/v2/webrtc/get_sdp.json

Request

Method：GET

```
/api/v2/webrtc/get_sdp.json?pro_id=1&win_id=1&ip=192.168.42.156
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
win_id	**[STRING]**	窗口 id
ip	**[STRING]**	视频源 ip 地址

Response

格式（Example）：

```
{
"result":"ok",
"data":{
"sdp":"v=0 o=- 5443219974135798586 2 IN IP4 127.0.0.1 s=- t=0 0 m=video 9 UDP/TLS/RTP/SAVPF 96 97 c=IN IP4
0.0.0.0 a=rtcp:9 IN IP4 0.0.0.0 a=ice-ufrag:MUZf a=ice-pwd:4QhikLcmGXnCfAzHDB++ZjM5 a=ice-options:trickle
a=fingerprint:sha-256
2A:5A:B8:43:66:05:B3:6A:E9:46:36:DF:DF:20:11:6A:F6:11:EA:D9:4E:26:E3:CE:5A:3A:C6:8D:03:49:7B:DE
a=setup:active a=mid:video a=ssrc:3587783331 cname:INxZnBV2Sty1zlmN a=ssrc:3587783331
msid:uiZ7cB0hsFDRGgTIMNp6TajUK9dOoHi43HVs a3b297e7-cdbe-464e-a32c-347465ace055".
      "session_id": "1"
}
}
```


Data 字段说明：

Field	Value	说明
sdp	**[STRING]**	sdp 信息
session_id	**[STRING]**	webrtc 链接 id

SDP: Session Description Protocol 会话描述协议（Session Description Protocol 或简写 SDP）描述的是流媒体的初始化参数。

9.2 PTZ 设置相关

API URL

/api/v2/ptz/set.json

Request

Method：POST

```
{
"pro_id": "1",
   "win_id": "pvw",
   "pos_id": 1,
"type": "ptz_zoom"/"ptz_zoom_speed"/"ptz_pan_tilt"/
   "ptz_pan_tilt_speed"/"ptz_store_preset"/
                "ptz_recall_preset"/"ptz_auto_focus"/
                "ptz_focus"/"ptz_focus_speed"/
                "ptz_white_balance_auto"/"ptz_white_balance_indoor"/
                "ptz_white_balance_outdoor"/"ptz_white_balance_oneshot"/
                "ptz_white_balance_manual"/"ptz_exposure_auto"/
                "ptz_exposure_manual"/"ptz_is_supported",
 "data": {
 "zoom_value": 0.5    //0.0 (zoomed in) ... 1.0 (zoomed out)
      "zoom_speed": -1.0        //-1.0 (zoom outwards) ... +1.0 (zoom inwards)
      "pan_value": -1.0 //-1.0 (left) ... 0.0 (centered) ... +1.0 (right)
      "tilt_value": -1.0 //-1.0 (bottom) ... 0.0 (centered) ... +1.0 (top)
      "pan_speed": -1.0 //-1.0 (moving right) ... 0.0 (stopped) ... +1.0 (moving left)
      "tilt_speed": -1.0 //-1.0 (down) ... 0.0 (stopped) ... +1.0 (moving up)
      "preset_no": 0 //0 ... 99
```

```
    "speed": 0.0 //0.0(as slow as possible) ... 1.0(as fast as possible)
    "focus_value": 0.0 //0.0 (focused to infinity) ... 1.0 (focused as close as possible)
    "focus_speed": -1.0 //-1.0 (focus outwards) ... +1.0 (focus inwards)
    "red": 0.0 //0.0(not red) ... 1.0(very red)
    "blue": 0.0 //0.0(not blue) ... 1.0(very blue)
    "exposure_level": 0.0 //0.0(dark) ... 1.0(light)
 }
}
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
win_id	**[STRING]**	窗口 id
ip	**[STRING]**	视频源 ip 地址

Response

格式（Example）：

```
{
"result": "ok"/"error"
}
```

9.3 转场特效类型列表

API URL

/api/v2/take/transition/list.json

Request

Method：GET

Response

格式（Example）：

```
{
 "result":"ok",
  "data":[
    {
      "id":1,            // 特效 id
      "label":"fade"       // 特效名称
    },
    {
      "id":2,
      "label":"pinwheel"
    },
    {
      "id":3,
      "label":"angular"
```

```
    },
    {
      "id":4,
      "label":"wipeRight"
    },
    {
      "id":5,
      "label":"wipeLeft"
    },
    {
      "id":6,
      "label":"wipeDown"
    },
    {
      "id":7,
      "label":"wipeUp"
    }
  ]
}
```

Data 字段说明：

Field	Value	说明
id	**[INT]**	类型 id
label	**[STRING]**	类型名称

9.4 Take 设置

API URL

/api/v2/take/set.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
  "time": 10,  //take 的时间间隔(s)，T bar 设置时 time 默认为 0
  "progress": 1.0,  //auto 时，默认为 1.0；T bar 设置时则为 Tbar 设置的值
  "select": 1 (默认选第一个效果)
}
```

Data 字段说明：

Field	Value	说明
pro_id	**[STRING]**	制作 id
time	**[INT]**	take 转场效果的时间(s)
progress	**[INT]**	默认为 1.0
select	**[INT]**	默认选第一个效果，值为 1

Response

格式（Example）：

```
{
  "result":"ok"
}
```

9.5 获取 Take 设置参数

API URL

/api/v2/take/get.json

Request

Method：POST

格式（Example）：

```
/api/v2/take/get.json?pro_id=1
```

Response

格式（Example）：

```
{
"result": "ok",
   "data" {
"time": 10,
   "select": 1
   }
}
```

9.6 设置 Take 参数

API URL

/api/v2/take/set_params.json

Request

Method：POST

格式（Example）：

```
{
"pro_id": "1",
   "time": 3,
   "select": 1
}
```

Response

格式（Example）

```
{
"result": "ok"
}
```

注：所有 api 接口的调用都需要在请求标头中带 app 字段，其中需要包括有效的 username 和 token 参数，例如: app**{"username":"admin","token":"2c5ff2da51b87dea33ba1faeb987b562"}**