

SenseNebula-M

星云 M 智能边缘节点

接口参考

文档编号：IDEA-PMO-AR303

文档版本：04

发布时间：2020-06-23

产品名称：智能边缘节点

产品系列：星云 M4s/M8s

系统版本：

SenseNebula-M V2.1.0



坚持原创，
让AI引领人类进步

目录

1 使用说明.....	7
2 登录相关接口.....	8
2.1 用户登录	8
2.2 用户登出	9
3 人像库相关接口.....	10
3.1 查询所有人像库.....	10
3.2 新建人像库.....	11
3.3 删除人像库.....	12
3.4 修改人像库.....	13
3.5 人像库条件查询.....	14
3.6 单张人脸图片查询.....	16
3.7 单张人脸图片入库.....	17
3.8 单张人脸图片删除.....	19
3.9 单张人脸图片属性修改.....	20
3.10 人脸图片条件查询	21
3.11 人脸图片分页导出	23
3.12 人脸图片选择导出	25
3.13 抓拍图片条件查询	26
3.14 抓拍图片分页导出	28
3.15 抓拍图片选择导出	30
3.16 告警图片条件查询	31
3.17 告警图片分页导出	34
3.18 告警图片选择导出	35
3.19 人脸图片批量入库	36
3.20 人脸图片批量删除	38
3.21 全量删除抓拍记录	39
3.22 人体抓拍图片条件查询.....	39
3.23 人体抓拍图片分页导出.....	42
3.24 抓拍图片选择导出	43
3.25 单张图片人脸检测	44
3.26 单张图片多人脸检测	46
3.27 单张人脸图片(base64 编码)入库	47
3.28 人脸图片(base64 编码)批量入库	49
3.29 单张图片(base64 编码)人脸检测	50
4 相机相关接口.....	52

4.1 查询相机	52
4.2 添加相机	57
4.3 修改相机	64
4.4 删除相机	72
4.5 查询 SIP 服务器 ID	73
5 人脸功能相关接口	74
5.1 1:N 人脸图片比对	74
5.2 1:1 人脸图片比对	78
5.3 1:N 人脸图片(base64)比对	79
5.4 1:1 人脸图片(base64)比对	84
6 系统配置接口	85
6.1 查询 NTP 配置信息	85
6.2 修改 NTP 配置信息	86
6.3 版本信息查询	87
6.4 存储策略查询	88
6.5 存储策略修改	89
6.6 系统时间查询	90
6.7 系统时间设置	91
6.8 网络查询	92
6.9 网络设置	94
6.10 默认网口设置	96
6.11 软件升级	97
6.12 系统重启	98
6.13 Device ID 查询	98
6.14 Device ID 修改	99
6.15 恢复默认配置	100
6.16 开启 web 访问功能	101
6.17 关闭 web 访问功能	102
6.18 查询 CPU 信息	102
6.19 查询内存信息	103
7 HTTP 配置接口	104
7.1 HTTP 配置信息查询	104
7.2 HTTP 配置信息修改	105
7.3 HTTP 心跳信息推送接口	106
7.4 HTTP 布控结果推送接口	107
8 HTTPS 配置接口	110
8.1 上传 HTTPS 证书和密钥	110
8.2 删除 HTTPS 证书和密钥	111
8.3 查询 HTTPS 证书和密钥	112

8.4 切换为 HTTPS 或 HTTP.....	113
8.5 查询当前为 HTTPS 或 HTTP.....	114
9 WebSocket 布控推送相关接口.....	115
9.1 WebSocket 查询布控推送密钥.....	115
9.2 WebSocket 布控结果推送接口.....	116
10 事件管理接口.....	119
10.1 查询相机与继电器绑定关系.....	119
10.2 添加相机与继电器绑定关系.....	121
10.3 删除相机与继电器绑定关系.....	122
附录 A：错误码.....	123
附录 B：人脸属性说明.....	125
附录 C：人体属性说明.....	126
附录 D：Http/Websocket 布控推送接收示例.....	129
.....	134

关于本手册

概述

本文适用于 SenseNebula 智能边缘节点 M 系列产品（以下简称 SenseNebula-M），内容包括各接口的接口描述、参数列表、参考示例。

读者对象

本文档主要适用于以下工程师：

- 售前工程师
- 技术支持工程师

版权声明

北京市商汤科技开发有限公司及其关联公司（以下并称“SenseTime”或“本公司”）对其发行的或与合作公司共同发行的包括但不限于产品或服务的全部内容拥有版权等知识产权，且受法律保护。

未经本公司书面许可，任何单位及个人不得以任何方式（手写、电子或机械的方式，包括通过复印、录音、录音笔录或信息收集系统）或理由对上述产品、服务、信息、材料的任何部分进行使用、复制、修改、抄录、传播、发行或与其它产品捆绑使用、销售。已经本公司授权使用相关内容的，应在授权范围内使用，并按照本公司要求注明来源。凡侵犯本公司版权等知识产权的，本公司必依法追究其法律责任。

商标声明

“SenseTime”等商标为北京市商汤科技开发有限公司或关联公司（以下并称“本公司”）的注册商标，受法律保护，侵权必究。

未经本公司书面许可，任何单位及个人不得以任何方式或理由对该商标进行使用、复制、修改、传播或其它产品捆绑使用、销售。凡侵犯本公司商标专用权的，我公司必依法追究其法律责任。

不保证声明

所有资料、信息、产品、软件、程序和服务均“按现状”提供，不附有任何形式的保证或担保。在法律许可的最大范围内，北京市商汤科技开发有限公司及其关联公司（以下并称“本公司”）明确声明免除所有明示的、暗示的、法定的和其他保证、担保、声明或承诺，包括但不限于有关适销、适用于某种特定用途以及有关所有权和知识产权的非侵权的保证。在不受任何限制的情况下，本公司不保证，此文档将及时、安全或无错误。

修订记录

文档版本	系统版本	发布日期	修订说明
01	V2.0.0	2019-11-21	第一次正式发布。
02	V2.0.1	2020-01-20	根据版本 V2.0.1 修改。
03	V2.0.2	2020-04-30	根据版本 V2.0.2 修改。
04	V2.1.0	2020-06-23	根据版本 V2.1.0 修改。

1 使用说明

- 1. 接口请求 url 的通用格式：http://\${ip}:\${port}/api/json ，端口号为 80。
- 2. 当切换至 https 时，接口 url 更改为 https:// \${ip}:\${port}/api/json，端口号为 443。
- 3. 用户登录会返回用户会话 id。用户登录后，后续所有 api 请求都需要在 http 请求 header 中以 ‘sessionid’ 带上会话 id。若用户登出或在登录后的 30 分钟未做任何操作，会话 id 将会过期自动失效，用户需重新登录。

1.1 用户登录

POST

Send Save

Params Authorization Headers (1) Body Pre-request Script Tests

Cookies Code Comments (0)

none form-data x-www-form-urlencoded raw binary JSON (application/json)

Beautify

```
1 {
2   "msg_id": "257",
3   "user_name": "test0104_3",
4   "user_pwd": "123456"
5 }
```

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 33 ms Size: 336 B Save Download

Pretty Raw Preview JSON

```
1 {
2   "code": 0,
3   "data": "5219561440865967480",
4   "msg": ""
5 }
```

2.2 新建目标库

POST

Send Save

Params Authorization Headers (2) Body Pre-request Script Tests

Cookies Code Comments (0)

Content-Type application/json

sessionid 5219561440865967480

Body Cookies Headers (8) Test Results

Status: 200 OK Time: 3022 ms Size: 354 B Save Download

Pretty Raw Preview JSON

```
1 {
2   "code": 0,
3   "data": {
4     "lib_id": 5,
5     "lib_name": "test0104_3"
6   },
7   "msg": ""
8 }
```

2 登录相关接口

2.1 用户登录

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	用户登录请求

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“257”	是	
user_name	string	用户名, 可以为大写字母、小写字母、数字和特殊字符(@ and .-_), 长度小于 50	是	
user_pwd	string	密码信息, 应该包含大写字母、小写字母、特殊字符、数字中的三种, 并且密码长度大于等于 8, 小于 32	是	

响应参数

参数	类型	描述
code	int	结果码 (0 即为 OK)
data	string	用户会话 id, 登录后的所有 api 请求都需要在 http 请求 header 中以 ‘sessionid’ 带上会话 id。若用户在登录后的 30 分钟未做任何操作, 会话 id 将会过期自动失效, 用户需重新登录。
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"257",
  "user_name":"admin",
  "user_pwd":"newpassword"
}
```

响应示例

```
{
  "code": 0,
  "data": "10538985380704462465",
  "msg": ""
}
```

2.2 用户登出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	用户登出请求

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“258”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	string	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"258",
}
```

响应示例

```
{
  "code": 0,
```

```
"data": "",
"msg": ""
}
```

3 人像库相关接口

3.1 查询所有人像库

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询所有人像库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1028”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
create_time	string	创建时间
lib_id	int	库 id
lib_name	string	库名称
lib_type	int	库类型（1：黑名单，2：白名单）
picture_no	int	库中图片数目
update_time	string	更新时间

参考示例

请求示例

```
{
  "msg_id": "1028"
}
```

```
}
```

响应示例

```
{
  "code": 0,
  "data": [
    {
      "create_time": "2019-01-23 10:54:13",
      "lib_id": 1,
      "lib_name": "test01",
      "lib_type": 1,
      "picture_no": 6,
      "update_time": "2019-01-23 10:54:13"
    }
  ],
  "msg": ""
}
```

3.2 新建人像库

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	新建人像库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1025”	是	
lib_name	string	新建库名称	是	
lib_type	int	库类型（1：黑名单，2：白名单）	是	2

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
----	----	----

lib_id	int	新建库 id
lib_name	string	新建库名称

参考示例

请求示例

```
{
  "msg_id": "1025",
  "lib_name": "test",
  "lib_type": 1
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "lib_id": 2,
    "lib_name": "test"
  },
  "msg": ""
}
```

3.3 删除人像库

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	删除指定人像库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1026”	是	
lib_id	int	需删除库 id	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1026",
  "lib_id":23
}
```

响应示例

```
{
  "code":0,
  "data": "",
  "msg":""
}
```

3.4 修改人像库

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	修改指定人像库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1027”	是	
lib_id	int	修改的库 id	是	
lib_name	string	库名称	是	
lib_type	int	库类型（1：黑名单，2：白名单）	是	1

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
lib_id	int	库 id
lib_name	string	库名称

参考示例

请求示例

```
{
  "msg_id": "1027",
  "lib_id": 1,
  "lib_name": "test",
  "lib_type": 1
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "lib_id": 1,
    "lib_name": "test"
  },
  "msg": ""
}
```

3.5 人像库条件查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	人像库条件查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1034”	是	
start_no	int	起始位置	否	默认 1
qry_len	int	查询数量，最大 50	否	默认 10
lib_id	int	库 id	否	
lib_name	string	库名称	否	
lib_type	int	库类型（1：黑名单，2：白名单）	否	1

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）

data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
record	json list	返回结果
result_num	int	查询结果条数
total_num	int	总条数

字段信息(record)

参数	类型	描述
create_time	string	创建时间
lib_id	int	库 id
lib_name	string	库名称
lib_type	int	库类型（1：黑名单，2：白名单）
picture_no	int	图片数量
update_time	string	更新时间

备注：人像库默认排序方式按照创建时间排序。

参考示例

请求示例

```
{
  "msg_id":"1034",
  "start_no":1,
  "qry_len":1
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "record": [
      {
        "create_time": "2019-01-23 10:54:13",
        "lib_id": 1,
        "lib_name": "test",
        "lib_type": 1,
        "picture_no": 6,
        "update_time": "2019-01-23 10:54:13"
      }
    ]
  }
}
```

```
    ],
    "result_num": 1,
    "total_num": 1
  },
  "msg": ""
}
```

3.6 单张人脸图片查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	单张人脸图片查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1032”	是	
img_id	string	待查询的人脸图片 id	是	
lib_id	int	待查询的库 id	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
create_time	string	创建时间
img	string	图片（base64 转码）
img_feat	string	图片特征
img_id	string	人脸图 id
img_path	string	图片路径
lib_id	int	所在库 id
person_addr	string	地址
person_age	string	年龄，范围 1~120
person_gender	string	性别（0：女，1：男）
person_idcard	string	身份标识码

person_name	string	姓名
-------------	--------	----

参考示例

请求示例

```
{
  "msg_id": "1032",
  "lib_id": 8,
  "img_id": "test"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "create_time": "2019-01-23 11:34:52",
    "img": "\ufffd\u001c\u0014Q_\uffd",
    "img_feat": "2442,501,111,179,482,239,414,486,49,111,143,169,192,247",
    "img_id": "test",
    "img_path": "img/1_a2d7ae51-f551-43c0-9bf5-af12ff32c630.png",
    "lib_id": 1,
    "person_addr": "shanghai",
    "person_age": "14",
    "person_gender": "1",
    "person_idcard": "123",
    "person_name": "renlian1"
  },
  "msg": ""
}
```

3.7 单张人脸图片入库

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	添加单张人脸图片到指定人脸库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1029”	是	
lib_id	int	图片所入的库 id	是	

img	file	需入库的人脸图片	是	
img_id	string	需入库的人脸图片 id, 代表图片的唯一标识, 长度限制为 48 位 (由字母、数字、短横线 “-” 组成), 用户可自定义, 但不可重复	否	
person_idcard	string	身份标识码	否	
person_name	string	姓名	否	
person_gender	string	性别 (0: 女, 1: 男)	否	
person_age	string	年龄, 范围 1~120	否	
person_addr	string	地址	否	

响应参数

参数	类型	描述
code	int	结果码 (0 即为 OK)
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
img_id	string	图片 id

备注 1: 图片文件只支持 jpg / jpeg / png/bmp/tif 格式。

备注 2: 入库图片 img_id 为图片唯一标识不允许重复; 字段未指定或为空, 则由系统自动分配。

参考示例

请求示例

```
Input Files
{
  'img': name='15427121547955542.jpg'
}
"img_id":"test",
"lib_id":11,
"msg_id":"1029"
```

响应示例

```
{
  "code": 0,
  "data": {
```

```
{
  "img_id": "test"
},
"msg": ""
}
```

3.8 单张人脸图片删除

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	单张人脸图片删除

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1031”	是	
img_id	string	图片 id	是	
lib_id	int	图片对应的人脸库 id	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1031",
  "img_id": "test_3",
  "lib_id": 8
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

3.9 单张人脸图片属性修改

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从指定人脸库中修改人脸图片属性

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1033”	是	
img_id	string	图片 id	是	
lib_id	int	所属人脸库 id	是	
person_idcard	string	身份标识码	否	
person_name	string	姓名	否	
person_gender	string	性别 (0: 女, 1: 男)	否	
person_age	string	年龄, 范围 1~120	否	
person_addr	string	地址	否	

响应参数

参数	类型	描述
code	int	结果码 (0 即为 OK)
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
img_id	string	图片 id
lib_id	int	所属人脸库 id

参考示例

请求示例

```
{
  "msg_id": "1033",
  "img_id": "test",
  "lib_id": 8,
  "person_age": "90"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "img_id": "test",
    "lib_id": 8
  },
  "msg": ""
}
```

3.10 人脸图片条件查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从指定人脸库中按条件查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1035”	是	
lib_id	int	图片对应的人脸库 id	是	
start_no	int	起始位置	否	默认 1
qry_len	int	查询数量，最大 50	否	默认 10
person_idcard	string	身份标识码（精确查询项）	否	
person_name	string	姓名（精确查询项）	否	
person_age	string	年龄，范围 1~120（精确查询项）	否	
person_gender	string	性别（精确查询项）（0：女，1：男）	否	
person_addr	string	地址（精确查询项）	否	
create_time	string	创建时间	否	
fuzzy_key	string	模糊查询项，表示可以按“姓名”或“性别”或“年	否	

		<p>龄”或“人员证件号”或“地址”来进行查询，关键字匹配到上面任一项即可进行查询，其中 fuzzy_key 字符串的最大长度为 40。注：</p> <p>fuzzy_key 查询与上述任一精确查询项互斥，即不可同时填写。性别只能输入“0”或“1”。即当该项输入“男”时，不会产生性别为“男”的结果。</p>		
--	--	--	--	--

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
record	json list	返回数据
result_num	int	查询结果条数
total_num	int	总条数

字段信息(record)

参数	类型	描述
person_idcard	string	身份标识码
person_addr	string	地址
person_age	string	年龄，范围 1~120
person_gender	string	性别（0：女，1：男）
img_id	string	图片 id
person_name	string	姓名
img_path	string	图片路径
lib_id	int	所在库 id

create_time	string	创建时间
-------------	--------	------

备注：默认排序方式按照创建时间排序。

参考示例

请求示例

```
{
  "msg_id": "1035",
  "lib_id": 1,
  "person_addr": "wuhan"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "record": [
      {
        "create_time": "2019-01-23 12:00:19",
        "img_id": "test_4",
        "img_path": "img/1_692198ba-e9ea-4ca3-bcf8-2b7a56fd734b.png",
        "lib_id": 1,
        "person_addr": "wuhan",
        "person_age": "14",
        "person_gender": "1",
        "person_idcard": "130182",
        "person_name": "wang"
      }
    ],
    "result_num": 1,
    "total_num": 1
  },
  "msg": ""
}
```

3.11 人脸图片分页导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	人脸图片分页导出

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1038”	是	
lib_id	int	库 id	是	
start_no	int	起始索引	否	默认 1
qry_len	int	导出数量，最大 500	否	默认 10

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
result_num	int	导出图片数量
total_num	int	人像库中图片总数

备注：导出人脸数据时，导出内容中不要包含逗号，否则会导出失败。

参考示例

请求示例

```
{
  "msg_id": "1038",
  "lib_id": 1
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/face_2019-01-23-12-04-27.zip",
    "result_num": 10,
    "total_num": 16
  },
  "msg": ""
}
```


3.12 人脸图片选择导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从指定人脸库中导出指定的人脸图片

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1041”	是	
lib_id	int	库 id	是	
img_ids	string	图片 id 集合，查询可得。用逗号隔开，最多 50 个，最少 1 个	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
failed_img_ids	string	导出图片失败的 img_id
result_num	int	导出图片数量
total_num	int	人像库中图片总数

备注：导出人脸数据时，导出内容中不要包含逗号，否则会导出失败。

参考示例

请求示例

```
{
  "msg_id": "1041",
  "lib_id": 1,
  "img_ids": "test,test_4"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/face_2019-01-23-12-06-19.zip",
    "failed_img_ids": "",
    "result_num": 2,
    "total_num": 2
  },
  "msg": ""
}
```

3.13 抓拍图片条件查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从抓拍图片中按条件查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1037”	是	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	否	
lib_name	string	人脸库名称	否	
lib_type	int	库类型（1：黑名单，2：白名单）	否	
start_time	string	开始时间	否	yyyy-MM-dd HH:mm:ss
stop_time	string	结束时间	否	
start_no	int	起始位置	否	默认 1
qry_len	int	查询数量，最大 50	否	默认 10

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
record	json	返回数据
result_num	int	查询结果条数
total_num	int	总条数

字段信息(record)

参数	类型	描述
camera_name	string	相机名称
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]
lib_id	int	比中的库 id
lib_name	string	库名称
lib_type	int	库类型（1：黑名单，2：白名单）
position	string	相机位置
ranking	int	值为 1，代表比对结果中的最高名次
similarity	int	相似度分数，范围：[0,100]
snap_id	string	抓拍图片 id
snap_path	string	抓拍图片路径
threshold	int	阈值，范围：[0,100]
trigger	string	抓拍时间
face_attr	json	人脸属性，详见附录 B

备注：默认排序方式按照抓拍时间排序。

参考示例

请求示例

```
{
  "msg_id":"1037",
  "start_no":1,
  "qry_len":1
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "record": [
      {
        "camera_name": "no1",
```

```

"channel": 3,
"face_attr": {
  "cap_style": "hat_style_type_none",
  "gender_code": "female",
  "glass_style": "glasses_style_type_none",
  "mustache_style": "mustache_style_type_none",
  "respirator_color": "color_type_none",
  "st_age": "st_adult",
  "st_age_value": "26.000000",
  "st_expression": "st_angry"
},
"lib_id": 1,
"lib_name": "test",
"lib_type": 1,
"position": "",
"ranking": 1,
"similarity": 32,
"snap_id": "9448f6e5-0d8b-4fd0-a13f-98b1a05500ec",
"snap_path": "record/9448f6e5-0d8b-4fd0-a13f-98b1a05500ec.jpg",
"threshold": 100,
"trigger": "2019-01-23 12:07:53"
}
],
"result_num": 1,
"total_num": 105
},
"msg": ""
}

```

3.14 抓拍图片分页导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从抓拍图片中按条件导出图片

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1040”	是	

start_no	int	起始索引	否	默认 1
qry_len	int	导出数量，最大 500	否	默认 10
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	否	
lib_name	string	人脸库名称	否	
lib_type	int	库类型（1：黑名单，2：白名单）	否	1
start_time	string	开始时间	否	
stop_time	string	结束时间	否	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
result_num	int	导出图片数量
total_num	int	人像库中图片总数

参考示例

请求示例

```
{
  "msg_id": "1040"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/catch_2019-01-23-12-09-42.zip",
    "result_num": 10,
    "total_num": 108
  },
  "msg": ""
}
```

3.15 抓拍图片选择导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从抓拍图片中选择导出图片

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1043”	是	
snap_ids	string	图片 id 集合，查询可得。用逗号隔开，最多 50 个，最少 1 个	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
failed_img_ids	string	导出图片失败的 img_id
result_num	int	导出图片数量
total_num	int	人像库中图片总数

参考示例

请求示例

```
{
  "msg_id": "1043",
  "snap_ids": "74a199aa-dad7-4d3a-95f0-f74dbb818974,f12ed623-87d6-49e0-939a-97262a0407e9"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/catch_2019-01-23-12-44-37.zip",

```

```

    "failed_img_ids": "",
    "result_num": 5,
    "total_num": 5
  },
  "msg": ""
}

```

3.16 告警图片条件查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从告警图片中按条件查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1036”	是	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	否	
lib_name	string	人脸库名称	否	
lib_ids	string	id 集合	否	
lib_type	int	库类型（1：黑名单，2：白名单）	否	
start_time	string	开始时间	否	
stop_time	string	结束时间	否	
start_no	int	起始位置	否	默认 1
qry_len	int	查询数量，最大 50	否	默认 10

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
record	json list	返回数据
result_num	int	查询结果条数

total_num	int	总条数
-----------	-----	-----

字段信息(record)

参数	类型	描述
camera_name	string	相机名称
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]
create_time	string	创建时间
img_id	string	图片 id
img_path	string	图片路径
lib_id	int	库 id
lib_name	string	库名称
lib_type	int	库属性（1：黑名单，2：白名单）
person_addr	string	地址
person_age	string	年龄，范围：1~120
person_gender	string	性别（0：女，1：男）
person_idcard	string	身份标识码
person_name	string	姓名
position	string	相机位置
ranking	int	值为 1，代表比中结果中的最高名次
similarity	int	相似度分数，范围：[0,100]
snap_id	string	抓拍图片 id
snap_path	string	抓拍图片路径
threshold	int	阈值，范围：[0,100]
trigger	string	抓拍时间
face_attr	json	人脸属性，详见附录 B

备注：人像库默认排序方式按照抓拍时间排序。

参考示例

请求示例

```
{
  "msg_id":"1036",
  "start_no":1,
  "qry_len":1
}
```

响应示例

```
{
  "code": 0,
```



```

"data":{
  "record":[
    {
      "camera_name": "no2",
      "channel": 2,
      "face_attr": {
        "cap_style": "hat_style_type_none",
        "gender_code": "female",
        "glass_style": "glasses_style_type_none",
        "mustache_style": "mustache_style_type_none",
        "respirator_color": "color_type_none",
        "st_age": "st_adult",
        "st_age_value": "26.000000",
        "st_expression": "st_angry"
      },
      "create_time": "2019-01-23 11:37:26",
      "img_id": "71f11a77-94c0-43c9-ab23-6e9076f6127a",
      "img_path": "img/1_71f11a77-94c0-43c9-ab23-6e9076f6127a.jpg",
      "lib_id": 1,
      "lib_name": "test",
      "lib_type": 1,
      "person_addr": "wang",
      "person_age": "14",
      "person_gender": "",
      "person_idcard": "wang",
      "person_name": "1101060000000061481",
      "position": "aaa",
      "ranking": 1,
      "similarity": 35,
      "snap_id": "1d9abf3d-09e1-4ef2-8832-f3d025ae0f91",
      "snap_path": "record/1d9abf3d-09e1-4ef2-8832-f3d025ae0f91.jpg",
      "threshold": 10,
      "trigger": "2019-01-23 14:29:22"
    }
  ],
  "result_num": 1,
  "total_num": 297
},
"msg": ""
}

```

3.17 告警图片分页导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	告警图片分页导出

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1039”	是	
start_no	int	起始索引	否	默认 1
qry_len	int	导出数量，最大 500	否	默认 10
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	否	
lib_name	string	人脸库名称	否	
lib_type	int	库属性（1：黑名单，2：白名单）	否	
start_time	string	开始时间	否	
stop_time	string	结束时间	否	
lib_ids	string	id 集合	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
result_num	int	导出图片数量
total_num	int	人像库中图片总数

参考示例

请求示例

```
{
```

```
"msg_id": "1039",
"lib_ids": "1"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/alarm_2019-01-23-14-31-51.zip",
    "result_num": 10,
    "total_num": 298
  },
  "msg": ""
}
```

3.18 告警图片选择导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从告警图片中选择导出图片

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1042”	是	
snap_ids	string	图片 id 集合，查询可得。用逗号隔开，最多 50 个，最少 1 个	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
failed_img_ids	string	导出图片失败的 img_id

result_num	int	导出图片数量
total_num	int	人像库中图片总数

参考示例

请求示例

```
{
  "msg_id": "1042",
  "snap_ids": "91f3705f-539b-4474-8f8b-fffb86d8ffa7"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/alarm_2019-01-23-14-35-29.zip",
    "failed_img_ids": "",
    "result_num": 1,
    "total_num": 1
  },
  "msg": ""
}
```

3.19 人脸图片批量入库

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	人脸图片批量入库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1030”	是	
lib_id	int	库 id	是	
img_*	file list	人脸图片。如 img_1、img_2 等。	是	
uuid	string	批次标识码	是	

响应参数

参数	类型	描述
----	----	----

code	int	结果码（0 即为 OK）
data	json	返回失败信息
msg	string	结果描述

字段信息(data)

参数	类型	描述
url	string	入库信息 csv 文件
err_count	int	失败数量

- 备注 1：入库文件名包含的内容为：姓名，img_id，证件号，性别，年龄，地址。
- 备注 2：批量入库文件名的格式为：姓名_img_id_证件号_性别_年龄_地址。
- 备注 3：img_id 为一个整体，代表图片的唯一标识，长度限制为 48 位（由字母、数字、短横线“-”组成），用户可自定义，但不可重复。
- 备注 4：性别只能填写 0 或 1，0 代表女性，1 代表男性。
- 备注 5：请严格按照规则中属性的顺序命名文件名，不同属性之间用下划线隔开，系统会根据规则自动获取图片的各项属性。若某个属性为空，请用一个“#”填充代替。

参考示例

请求示例

```
Input Files
{
  'img_1': name='zhangsan_asdqwe12345_32011111_0_23_beijing.jpg',
  'img_2': name='lisi_abcdse0987_32011111_1_41_shanghai.jpg' ,
  ...
}
"msg_id": "1030",
"lib_id": 11,
"uuid": "test"
```

响应示例

```
{
  "code": 0,
  "data": {
    "err_count": 0,
    "url": "tmp/image_.csv"
  },
  "msg": ""
}
```

3.20 人脸图片批量删除

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	人脸图片批量删除

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1044”	是	
img_ids	string	图片 id,多张图片用逗号隔开,最多 50 个 id	是	
lib_id	int	图片对应的人脸库 id	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1044",
  "img_ids":"f30c0f4b-d783-44a5-b118-b4db9ea2b8c4,85b728f8-ed96-42c3-a7e2-f811cf7a1481",
  "lib_id":2
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

3.21 全量删除抓拍记录

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	全量删除抓拍记录

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1045”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1045"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

3.22 人体抓拍图片条件查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从抓拍图片中按条件查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1046”	是	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	否	
start_time	string	开始时间	否	yyyy-MM-dd HH:mm:ss
stop_time	string	结束时间	否	
start_no	int	起始位置	否	默认 1
qry_len	int	查询数量，最大 50	否	默认 10

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
record	json list	返回数据
result_num	int	查询结果条数
total_num	int	总条数

字段信息(record)

参数	类型	描述
camera_name	string	相机名称
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]
position	string	相机位置
quality	int	质量分数，范围：[0,100]
snap_id	string	抓拍图片 id
snap_path	string	抓拍图片路径
trigger	string	抓拍时间
body_attr	json	人体属性，详见附录 C

备注：默认排序方式按照抓拍时间排序。

参考示例

请求示例

```
{
```



```
"msg_id": "1046",
"start_no": 1,
"qry_len": 1,
"channel": 1,
"start_time": "2019-10-16 13:20:00",
"stop_time": "2019-10-16 18:00:00"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "record": [
      {
        "body_attr": {
          "bag_style": "bag_style_type_none",
          "cap_style": "hat_style_type_none",
          "coat_color": "black",
          "coat_length": "long_sleeve",
          "coat_style": "t_shirt",
          "gender_code": "female",
          "hair_color": "black",
          "hair_style": "long",
          "shoes_color": "black",
          "shoes_style": "sandal",
          "st_age": "st_adult",
          "st_bag": "st_bag",
          "st_coat_pattern": "st_pure",
          "st_hat": "st_hat",
          "st_hold_object_in_front": "st_hold_object_in_front",
          "st_pedestrian_angle": "st_front",
          "st_respirator": "st_respirator",
          "st_trousers_pattern": "st_pure",
          "st_umbrella": "st_umbrella",
          "trousers_color": "black",
          "trousers_len": "st_skirt"
        },
        "camera_name": "c1",
        "channel": 1,
        "position": "",
        "quality": 67,

```

```

      "snap_id": "cb19f0a6-0e55-4b86-8079-d22d530ad14e",
      "snap_path": "record/17/body/cb19f0a6-0e55-4b86-8079-d22d530ad14e.jpg",
      "trigger": "2019-10-16 14:01:41"
    }
  ],
  "result_num": 1,
  "total_num": 371
},
"msg": ""
}

```

3.23 人体抓拍图片分页导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从人体抓拍图片中按条件导出图片

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1048”	是	
start_no	int	起始索引	否	默认 1
qry_len	int	导出数量，最大 500	否	默认 10
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	否	
start_time	string	开始时间	否	
stop_time	string	结束时间	否	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径

result_num	int	导出图片数量
total_num	int	人像库中图片总数

参考示例

请求示例

```
{
  "msg_id": "1048"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/body_2019-10-23-12-09-42.zip",
    "result_num": 10,
    "total_num": 108
  },
  "msg": ""
}
```

3.24 抓拍图片选择导出

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	从人体抓拍图片中选择导出图片

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1047”	是	
snap_ids	string	图片 id 集合，查询可得。用逗号隔开，最多 50 个，最少 1 个	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
export_url	string	导出文件路径
failed_img_ids	string	导出图片失败的 img_id
result_num	int	导出图片数量
total_num	int	选择导出图片总数

参考示例

请求示例

```
{
  "msg_id":"1047",
  "snap_ids":"74a199aa-dad7-4d3a-95f0-f74dbb818974,f12ed623-87d6-49e0-939a-97262a0407e9"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "export_url": "tmp/body_2019-01-23-12-44-37.zip",
    "failed_img_ids": "",
    "result_num": 2,
    "total_num": 2
  },
  "msg": ""
}
```

3.25 单张图片人脸检测

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	单张图片人脸检测

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“769”	是	
img	file	人脸图片	是	

响应参数

参数	类型	描述
----	----	----

code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
quality	int	人脸质量分数
coordinate	string	人脸坐标，例如 “x1,y1,x2,y2” ，分别表示矩形的左、上、右和底边界的坐标。
attribute	json	人脸属性，详见附录 B

参考示例

请求示例

```
{
  "msg_id": "769",
  "img": "test.jpg"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "attribute": {
      "cap_style": "hat_style_type_none",
      "gender_code": "male",
      "glass_style": "glasses_style_type_none",
      "mustache_style": "mustache_style_type_none",
      "respirator_color": "color_type_none",
      "st_age": "st_adult",
      "st_age_value": "24.000000",
      "st_expression": "st_calm"
    },
    "quality": 67,
    "coordinate": "56,10,432,678"
  },
  "msg": ""
}
```

3.26 单张图片多人脸检测

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	单张图片多人脸检测

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“782”	是	
img	file	人脸图片	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json list	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
quality	int	人脸质量分数
coordinate	string	人脸坐标，例如 “x1,y1,x2,y2”，分别表示矩形的左、上、右和底边界的坐标。
img_path	string	人脸图路径
attribute	json	人脸属性，详见附录 B

参考示例

请求示例

```
{
  "msg_id": "782",
  "img": "test.jpg"
}
```

响应示例

```
{
  "code": 0,
  "data": [
    {
      "attribute": {
```

```

    "cap_style": "hat_style_type_none",
    "gender_code": "male",
    "glass_style": "glasses_style_type_none",
    "mustache_style": "mustache_style_type_none",
    "respirator_color": "color_type_none",
    "st_age": "st_adult",
    "st_age_value": "24.000000",
    "st_expression": "st_calm"
  },
  "quality": 67,
  "img_path": "tmp/943f95bc-c7df-408f-bae8-c3285557f3ef.jpg",
  "coordinate": "56,10,432,678"
}
],
"msg": ""
}

```

3.27 单张人脸图片(base64 编码)入库

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	添加单张人脸图片到指定人脸库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1051”	是	
lib_id	int	图片所入的库 id	是	
img	json	需入库的人脸图片	是	
img_id	string	需入库的人脸图片 id, 代表图片的唯一标识, 长度限制为 48 位 (由字母、数字、短横线 “-” 组成), 用户可自定义, 但不可重复	否	
person_idcard	string	身份标识码	否	
person_name	string	姓名	否	
person_gender	string	性别 (0: 女, 1: 男)	否	

		男)		
person_age	string	年龄，范围 1~120	否	
person_addr	string	地址	否	

字段信息(img)

参数	类型	描述
filename	string	图片文件名
data	string	base64 编码的图片数据

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
img_id	string	图片 id

备注 1：图片文件只支持 jpg / jpeg / png/bmp/tif 格式。

备注 2：入库图片 img_id 为图片唯一标识不允许重复；字段未指定或为空，则由系统自动分配。

参考示例

请求示例

```
Input Files
{
  "msg_id": "1051",
  "lib_id": 1,
  "img": {
    "filename": "haha.jpg",
    "data": ""
  }
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "img_id": "test"
  },
  "msg": ""
}
```


3.28 人脸图片(base64 编码)批量入库

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	人脸图片批量入库

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1052”	是	
lib_id	int	库 id	是	
imgs	json	人脸图片列表, img 列表	是	
uuid	string	批次标识码	是	

字段信息(img)

参数	类型	描述
filename	string	图片文件名
data	string	base64 编码的图片数据

响应参数

参数	类型	描述
code	int	结果码 (0 即为 OK)
data	json	返回失败信息
msg	string	结果描述

字段信息(data)

参数	类型	描述
url	string	入库信息 csv 文件
err_count	int	失败数量

备注 1: 入库文件名包含的内容为: 姓名, img_id, 证件号, 性别, 年龄, 地址。

备注 2: 批量入库文件名的格式为: 姓名_img_id_证件号_性别_年龄_地址。

备注 3: img_id 为一个整体, 代表图片的唯一标识, 长度限制为 48 位 (由字母、数字、短横线 “-” 组成), 用户可自定义, 但不可重复。

备注 4: 性别只能填写 0 或 1, 0 代表女性, 1 代表男性。

备注 5: 请严格按照规则中属性的顺序命名文件名, 不同属性之间用下划线隔开, 系统会根据规则自动获取图片的各项属性。若某个属性为空, 请用一个 “#” 填充代替。

参考示例

请求示例

```
Input Files
{
  "msg_id": "1052",
  "lib_id": 1,
  "uuid": "xyyzz",
  "imgs": [
    {
      "filename": "haha.jpg",
      "data": ""
    }
  ]
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "err_count": 0,
    "url": "tmp/image_.csv"
  },
  "msg": ""
}
```

3.29 单张图片(base64 编码)人脸检测

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	单张图片人脸检测

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“779”	是	
img	json	人脸图片	是	

字段信息(img)

参数	类型	描述
filename	string	图片文件名

data	string	base64 编码的图片数据
------	--------	----------------

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
quality	int	人脸质量分数
coordinate	string	人脸坐标，例如“x1,y1,x2,y2”，分别表示矩形的左、上、右和底边界的坐标。
attribute	json	人脸属性，详见附录 B

参考示例

请求示例

```
{
  "msg_id": "779",
  "img": {
    "filename": "haha.jpg",
    "data": ""
  }
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "attribute": {
      "cap_style": "hat_style_type_none",
      "gender_code": "male",
      "glass_style": "glasses_style_type_none",
      "mustache_style": "mustache_style_type_none",
      "respirator_color": "color_type_none",
      "st_age": "st_adult",
      "st_age_value": "24.000000",
      "st_expression": "st_calm"
    },
    "quality": 67,
  }
}
```

```
"coordinate": "56,10,432,678"
},
"msg": ""
}
```

4 相机相关接口

4.1 查询相机

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询相机

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“516”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
camera	json list	相机
computing_power_capacity	int	系统最大算力；1 路抓拍摄像机算力值为 1，1 路网络摄像机算力值为 2，M4s 的系统最大算力为 8，M8s 的系统最大算力为 16。
used_computing_power	int	系统当前算力

字段信息(camera[RTSP])

参数	类型	描述
camera_name	string	相机名称
channel	int	通道；M4s 的范围：[1,8]，M8s

		的范围：[1,16]
decoder	string	协议解码方式
camera_mode	string	相机模式（VIDEO：视频流，适用于网络摄像机，IMAGE：图片流，适用于抓拍摄像机）
description	string	相机描述
lib_ids	string	VIDEO 模式返回相机布控库 ids（可返回多个）
position	string	相机位置
protocol	string	相机协议
status	int	相机的实时状态（1：在线，0：不在线）
threshold	int	阈值，范围：[0,100]
url	string	相机 URL
camera_roi	string	相机感兴趣区域(“x1,y1,x2,y2”，分别表示感兴趣区域的左、上、右、下 4 个边界的坐标值)
roi_max	int	相机感兴趣区域设置允许的最大面积（计算方法：(x2 - x1) * (y2 - y1)）
snap_mode	int	相机的抓拍策略（1：精准，2：定时，3：实时）
facesize_min	int	最小人脸像素值范围： [60,facesize_max]，值无效时默认为 60；例：120，人脸需要大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围： [facesize_min, 1000]，值无效时默认为 1000；例：800，人脸需要小于 800 * 800 像素

字段信息(camera[onvif])

参数	类型	描述
camera_name	string	相机名称
camera_pwd	int	相机密码
camera_user	string	相机用户
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]
decoder	string	解码方式
camera_mode	string	相机模式

description	string	相机描述
ip	string	相机 ip
lib_ids	string	VIDEO 模式返回相机布控库 ids (可返回多个)
port	int	相机端口号
position	string	相机位置
protocol	string	相机协议
status	int	相机的实时状态 (1: 在线, 0: 不在线)
threshold	int	阈值, 范围: [0,100]
camera_roi	string	相机感兴趣区域(“x1,y1,x2,y2”, 分别表示感兴趣区域的左、上、右、下 4 个边界的边界值)
roi_max	int	相机感兴趣区域设置允许的最大面积 (计算方法: $(x2 - x1) * (y2 - y1)$)
snap_mode	int	相机的抓拍策略 (1: 精准, 2: 定时, 3: 实时)
facesize_min	int	最小人脸像素值范围: [60,facesize_max], 值无效时默认为 60; 例: 120, 人脸需要大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围: [facesize_min, 1000], 值无效时默认为 1000; 例: 800, 人脸需要小于 800 * 800 像素

字段信息(camera[GB28181])

参数	类型	描述
channel	int	通道; M4s 的范围: [1,8], M8s 的范围: [1,16]
camera_mode	string	相机模式
ip	string	SIP 服务器 IP
port	int	SIP 服务器端口, 通常是 5060
server_sip_id	string	SIP 服务器 SIP ID, 最长 20 位
camera_sip_id	string	相机 SIP ID, 最长 20 位
camera_name	string	相机名称
camera_pwd	string	GB28181 鉴权密码
description	string	相机描述
lib_ids	string	VIDEO 模式返回相机布控库 ids

		(可返回多个)
position	string	相机位置
protocol	string	相机协议，内容为“GB28181”
status	int	相机的实时状态（1：在线，0：不在线）
threshold	int	阈值，范围：[0,100]
snap_mode	int	相机的抓拍策略（1：精准，2：定时，3：实时）
facesize_min	int	最小人脸像素值范围： [60,facesize_max]，值无效时默认为 60；例：120，人脸需要大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围： [facesize_min, 1000]，值无效时默认为 1000；例：800，人脸需要小于 800 * 800 像素

字段信息(camera[image])

参数	类型	描述
camera_name	string	相机名称
camera_pwd	string	相机登录密码
camera_user	string	相机登录用户名
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]
description	string	相机描述
ip	string	相机 IP
lib_ids	string	相机布控库 id（返回多个）
port	int	相机端口号
position	string	相机位置
product	string	版本型号
status	int	相机的实时状态（1：在线，0：不在线）
threshold	int	阈值，范围：[0,100]
vendor	string	相机厂商
camera_mode	string	相机模式

参考示例

请求示例

```
{
  "msg_id": "516"
```

```
}
```

响应示例

```
{
```

```
"code": 0,
"data": {
  "camera": [
    {
      "camera_mode": "IMAGE",
      "camera_name": "123",
      "camera_pwd": "1",
      "camera_user": "1",
      "channel": 4,
      "description": "",
      "ip": "1.1.1.1",
      "lib_ids": "22",
      "port": 1,
      "position": "",
      "product": "",
      "status": 0,
      "threshold": 85,
      "vendor": "dahua"
    },
    {
      "camera_mode": "VIDEO",
      "camera_name": "rtsp12",
      "camera_roi": "",
      "channel": 7,
      "decoder": "",
      "description": "",
      "facesize_max": 1000,
      "facesize_min": 60,
      "lib_ids": "221",
      "position": "",
      "protocol": "rtsp",
      "roi_max": 2073600,
      "snap_mode": 3,
      "status": 1,
      "threshold": 75,
      "url": "rtsp://admin:admin123@10.151.116.112:554"
```



```
    }
  ],
  "computing_power_capacity": 16,
  "used_computing_power": 3
},
"msg": ""
}
```

当系统无相机设备时，响应报文如下：

响应示例

```
{
  "code": 0,
  "data": {
    "camera": [],
    "computing_power_capacity": 16,
    "used_computing_power": 0
  },
  "msg": ""
}
```

4.2 添加相机

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	添加相机

a. rtsp VIDEO 模式相机添加请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“513”	是	
protocol	string	相机支持的协议 (rtsp)	是	
decoder	string	相机支持的解码方式	否	
camera_mode	string	相机模式	是	
url	string	需添加相机的 URL	是	
position	string	需添加的相机位置	否	
lib_ids	string	VIDEO 模式需添加 的相机布控库 ids (可输入多个)	否	

description	string	需添加的相机描述	否	
threshold	int	阈值，范围： [0,100]	是	
camera_name	string	相机名称	是	
snap_mode	int	相机的抓拍策略 (1: 精准, 2: 定时, 3: 实时)	是	
facesize_min	int	最小人脸像素值范围： [60,facesize_max]， 值无效时默认为 60	否	例：120 人脸需要 大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围： [facesize_min, 1000]， 值无效时默认为 1000	否	例：800 人脸需要 小于 800 * 800 像素

响应参数

参数	类型	描述
code	int	结果码 (0 即为 OK)
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "camera_mode": "VIDEO",
  "msg_id": "513",
  "protocol": "rtsp",
  "url": "rtsp://admin:admin123@10.5.4.178:554",
  "position": "test",
  "lib_ids": "10",
  "threshold": 1,
  "camera_name": "test",
  "snap_mode": 1,
  "facesize_min": 60,
```

```
"facesize_max":1000
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "channel": 1
  },
  "msg": ""
}
```

b. onvif VIDEO 模式相机添加请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“513”	是	
ip	string	需添加的相机 IP	是	
port	int	需添加的相机端口号	是	
camera_mode	string	相机模式	是	
position	string	需添加的相机位置	否	
camera_user	string	需添加的相机登录用户名	是	
camera_pwd	string	需添加的相机登录密码	是	
protocol	string	相机视频流协议 (onvif)	是	
decoder	string	相机支持的解码方式	否	
lib_ids	string	VIDEO 模式需添加的相机布控库 ids (可输入多个)	否	
description	string	相机描述	否	
threshold	int	阈值，范围：[0,100]	是	
camera_name	string	相机名称	是	
snap_mode	int	相机的抓拍策略(1：精准；2 定时；3 实时)	是	
facesize_min	int	最小人脸像素值范围：[60,facesize_max]，值无效时默认为 60	否	例：120 人脸需要大于 120 * 120 像素

facesize_max	int	最大人脸像素值范围：[facesize_min, 1000]，值无效时默认为 1000	否	例：800 人脸需要小于 800 * 800 像素
--------------	-----	---	---	---------------------------

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "camera_mode": "VIDEO",
  "description": "test",
  "ip": "10.5.4.178",
  "lib_ids": "18",
  "port": 80,
  "position": "test",
  "camera_pwd": "admin123",
  "camera_user": "admin",
  "decoder": "onvif",
  "protocol": "onvif",
  "threshold": 1,
  "msg_id": "513",
  "camera_name": "test",
  "snap_mode": 1,
  "facesize_min": 60,
  "facesize_max": 1000
}
```

响应示例

```
{
  "code": 0,
  "data": {
```

```
"channel": 2
},
"msg": ""
}
```

c. GB28181 VIDEO 模式相机添加请求参数列表：

当星云 M 作为 SIP 服务端时，鉴权密码是 sensetime.

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“513”	是	
ip	string	需添加的相机 IP	是	
port	int	需添加的相机端口号	是	
camera_mode	string	相机模式	是	
server_sip_id	string	SIP 服务器 SIP ID	是	最长为 20
camera_sip_id	string	相机 SIP ID	是	最长为 20
position	string	需添加的相机位置	否	
camera_pwd	string	GB28181 鉴权密码	否	
protocol	string	相机视频流协议 (GB28181)	是	
lib_ids	string	VIDEO 模式需添加 的相机布控库 ids (可输入多个)；	否	
description	string	相机描述	否	
threshold	int	阈值，范围： [0,100]	是	
camera_name	string	相机名称	是	
snap_mode	int	相机的抓拍策略(1： 精准；2 定时；3 实 时)	是	
facesize_min	int	最小人脸像素值范 围： [60,facesize_max]， 值无效时默认为 60	否	例：120 人脸需要 大于 120 * 120 像 素
facesize_max	int	最大人脸像素值范 围：[facesize_min, 1000]，值无效时默 认为 1000	否	例：800 人脸需要 小于 800 * 800

响应参数

参数	类型	描述
----	----	----

code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "camera_mode":"VIDEO",
  "description":"test",
  "ip":"10.5.4.178",
  "lib_ids":"18",
  "port":80,
  "position":"test",
  "camera_pwd":"admin123",
  "server_sip_id":"34020000002000000001",
  "camera_sip_id":"34020000001320000001",
  "protocol": "gb28181",
  "threshold":1,
  "msg_id":"513",
  "camera_name":"test",
  "snap_mode":"1",
  "facesize_min":60,
  "facesize_max":1000
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "channel": 2
  },
  "msg": ""
}
```

d. IMAGE 模式相机添加请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“513”	是	
ip	string	需添加的相机 IP	是	
port	int	需添加的相机端口号	是	
camera_mode	string	相机模式	是	
position	string	需添加的相机位置	否	
camera_user	string	需添加的相机登录用户名	是	
camera_pwd	string	需添加的相机登录密码	是	
vendor	string	需添加的相机生产厂商	否	字段可填 hikvision、sensedlc11、sensedlct、sensedlcd 和 dahua。不填或填错都是 dahua。
product	string	需添加的相机型号	否	
lib_ids	string	需添加的相机布控 ids（可输入多个库 Id）	否	
description	string	需添加的相机描述	否	
threshold	int	阈值，范围：[0,100]	是	
camera_name	string	相机名称	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "camera_mode":"IMAGE",
  "description":"test",
  "ip":"10.5.4.178",
  "lib_ids":"18",
  "port":37777,
  "position":"test",
  "camera_pwd":"admin123",
  "camera_user":"admin",
  "vendor": "dahua",
  "product": "01",
  "threshold":1,
  "msg_id":"513",
  "camera_name":"test"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "channel": 2
  },
  "msg": ""
}
```

备注 1：视频流相机目前只支持 RTSP、onvif 和 GB28181 协议，decoder 和 protocol 暂时为保留字段。

备注 2：人像库 Id 参数格式为“lib1,lib2,lib3,...”，库 Id 间用“,”进行分隔。

备注 3：channel 字段, 星云 M4 该字段范围为 1~8, 星云 M8 该字段范围为 1~16。

备注 4：当添加的相机 ip 填写错误或无效时，需要耐心等待服务响应。

备注 5：vendor 厂商字段，可填 hikvision、sensedlc11、sensedlct、sensedlcd 和 dahua/sensedlcaa，分别代表不同类型的抓拍机。

4.3 修改相机

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	修改相机

a. rtsp 格式码流 VIDEO 模式相机修改请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“515”	是	
url	string	修改后的相机 URL	否	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	是	
position	string	修改后的相机位置	否	
lib_ids	string	VIDEO 模式修改后的相机布控库 ids（可输入多个）；	否	
camera_roi	string	修改后的相机感兴趣区域	否	
description	string	修改后的相机描述	否	
threshold	int	修改后的相似度阈值，范围：[0,100]	否	
camera_name	string	修改后的相机名称	否	
snap_mode	int	相机的抓拍策略(1：精准；2 定时；3 实时)	否	
facesize_min	int	最小人脸像素值范围：[60,facesize_max]，值无效时默认为 60	否	例：120 人脸需要大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围：[facesize_min, 1000]，值无效时默认为 1000	否	例：800 人脸需要小于 800 * 800 像素

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "msg_id": "515",
  "url": "rtsp://admin:admin123@10.5.4.178:554",
  "channel": 1,
  "position": "test",
  "lib_ids": "1",
  "threshold":1,
  "camera_name":"test",
  "description":"home",
  "camera_roi":"0,180,1920,900",
  "snap_mode":1,
  "facesize_min":60,
  "facesize_max":1000
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "channel": 1
  },
  "msg": ""
}
```

b. onvif VIDEO 模式相机修改请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“515”	是	
ip	string	修改后的相机 IP	否	
port	int	修改后的相机端口号	否	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	是	
position	string	修改后的相机位置	否	
camera_user	string	修改后的相机登录用户名	否	
camera_pwd	string	修改后的相机登录密码	否	
lib_ids	string	VIDEO 模式修改后	否	

		的相机布控库 ids (可输入多个)		
camera_roi	string	修改后的相机感兴趣区域	否	
description	string	修改后的相机描述	否	
threshold	int	修改后的相似度阈值，范围：[0,100]	否	
camera_name	string	修改后的相机名称	否	
snap_mode	int	相机的抓拍策略(1：精准；2 定时；3 实时)	否	
facesize_min	int	最小人脸像素值范围： [60,facesize_max]， 值无效时默认为 60	否	例：120 人脸需要大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围：[facesize_min, 1000]，值无效时默认为 1000	否	例：800 人脸需要小于 800 * 800 像素

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "channel":2,
  "description":"test",
  "ip":"10.5.4.178",
  "lib_ids":"1",
  "port":80,
  "position":"test",
```

```

"camera_pwd":"admin123",
"camera_user":"admin",
"threshold":1,
"msg_id":"515",
"camera_name":"test",
"camera_roi":"0,130,1020,700",
"snap_mode":1,
"facesize_min":60,
"facesize_max":1000
}

```

响应示例

```

{
  "code": 0,
  "data": {
    "channel": 2
  },
  "msg": ""
}

```

c. GB28181 VIDEO 模式相机修改请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“515”	是	
ip	string	SIP 服务器 IP	否	
port	int	SIP 服务器端口	否	
server_sip_id	string	SIP 服务器 SIP ID	否	最长为 20
camera_sip_id	string	相机 SIP ID	否	最长为 20
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	是	
position	string	相机位置	否	
camera_pwd	string	GB28181 鉴权密码	否	
lib_ids	string	VIDEO 模式需修改的相机布控库 ids（可输入多个）；	否	
description	string	相机描述	否	
threshold	int	阈值，范围：[0,100]	否	
camera_name	string	相机名称	否	
snap_mode	int	相机的抓拍策略(1：	否	

		精准；2 定时；3 实时)		
facesize_min	int	最小人脸像素值范围： [60,facesize_max]， 值无效时默认为 60	否	例：120 人脸需要 大于 120 * 120 像素
facesize_max	int	最大人脸像素值范围：[facesize_min, 1000]，值无效时默认为 1000	否	例：800 人脸需要 小于 800 * 800 像素

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "channel":2,
  "description":"test",
  "ip":"10.5.4.178",
  "lib_ids":"18",
  "port":80,
  "position":"test",
  "camera_pwd":"admin123",
  "server_sip_id":"34020000002000000001",
  "camera_sip_id":"34020000001320000001",
  "protocol":"gb28181",
  "threshold":1,
  "msg_id":"515",
  "camera_name":"test",
  "snap_mode":1,
  "facesize_min":60,
```

```
"facesize_max":1000
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "channel": 2
  },
  "msg": ""
}
```

d. IMAGE 模式相机修改请求参数列表：

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“515”	是	
ip	string	修改后的相机 IP	否	
port	int	修改后的相机端口号	否	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	是	
position	string	修改后的相机位置	否	
camera_user	string	修改后的相机登录用户名	否	
camera_pwd	string	修改后的相机登录密码	否	
lib_ids	string	修改后的相机布控 ids（可输入多个库 id）	否	
description	string	修改后的相机描述	否	
threshold	int	修改后的相似度阈值，范围：[0,100]	否	
vendor	string	需修改的相机生产厂商	否	字段可填 hikvision、sensedlc11、sensedlct、sensedlcd 和 dahua。不填或填错都是 dahua。
camera_name	string	修改后的相机名称	否	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

参考示例

请求示例

```
{
  "channel":1,
  "description":"test",
  "ip":"10.5.4.177",
  "lib_ids":"1",
  "port":80,
  "position":"test",
  "camera_pwd":"admin123",
  "camera_user":"admin",
  "threshold":1,
  "vendor": "dahua",
  "msg_id":"515",
  "camera_name":"test"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "channel": 1
  },
  "msg": ""
}
```

备注 1：视频流相机目前只支持 RTSP、onvif 和 GB28181 协议，decoder 和 protocol 暂时为保留字段，不支持跨协议修改相机。

备注 2：人像库 Id 参数格式为“lib1,lib2,lib3,...”，库 Id 间用“,”进行分隔。

备注 3：channel 字段, 星云 M4 该字段范围为 1~8, 星云 M8 该字段范围为 1~16。

备注 4：当修改的相机 ip 填写错误或无效时，需要耐心等待服务响应。

备注 5：若非必填字段不填，会保留原有信息。

备注 6：vendor 厂商字段，可填 hikvision、sensedlc11、sensedlct、sensedlcd 和 dahua/sensedlcaa，分别代表不同类型的抓拍机。

4.4 删除相机

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	删除相机

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“514”	是	
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "514",
  "channel": 16
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```


4.5 查询 SIP 服务器 ID

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询 SIP 服务器 ID

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1310”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

字段信息(data)

参数	类型	描述
server_sip_id	string	SIP 服务器 ID

参考示例

请求示例

```
{
  "msg_id": "1310"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "server_sip_id": "34020000002000000001"
  },
  "msg": ""
}
```

5 人脸功能相关接口

5.1 1:N 人脸图片比对

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	1:N 人脸图片比对，输入单张图片与指定人脸库进行比对，支持返回 Top1-Top50，支持返回人脸坐标和人脸质量分数，以及 N 个人的属性(姓名，年龄，性别，ID)。

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“772”	是	
threshold	int	比对阈值，范围：[0,100]，默认 0	否	
img	file	需要比对的人脸图片格式 “xxx.jpg “	是	
lib_ids	string	比对目标人脸库,可输入多个，最多 50 个，库 id 用 “,” 进行分隔	是	
topk	int	需返回的 topn 比对结果数量，最多可返回 top50，范围：[1,50]，默认 1	否	
n_topk	int	返回 topn 的第 n 个结果，范围：[1,topk]，默认 1	否	

响应参数

参数	类型	描述
code	int	结果码（0 即为成功）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
n_topn	json	topn 个比对结果中的第 n 个，由请求参数 n_topk 决定
topn	json	前 topn 个比对结果
ln_pic	json	输入人脸在原图属性

字段信息(n_topn)

参数	类型	描述
img_id	string	名次对应的底库人脸图 ID
img_path	string	名次对应的底库人脸图路径
lib_id	int	名次对应的库 id
lib_name	string	名次对应的库名
lib_type	int	名次对应的人脸库类型
ranking	int	比对名次
similarity	int	名次对应的人脸图比对相似度系数[0,100]
id	string	名次对应的底库人脸图 ID
name	string	名次对应的底库人脸图姓名
gender	string	名次对应的底库人脸图性别（0：女，1：男）
age	string	名次对应的底库人脸图年龄
address	string	名次对应的底库人脸图地址

字段信息(topn)

参数	类型	描述
img_id	string	名次对应的底库人脸图 imgid
img_path	string	名次对应的底库人脸图路径
lib_id	int	名次对应的库 id
lib_name	string	名次对应的库名
lib_type	int	名次对应的人脸库类型
ranking	int	比对名次
similarity	int	名次对应的人脸图比对相似度系数[0,100]
id	string	名次对应的底库人脸图 ID
name	string	名次对应的底库人脸图姓名
gender	string	名次对应的底库人脸图性别（0：女，1：男）
age	string	名次对应的底库人脸图年龄

address	string	名次对应的底库人脸图地址
---------	--------	--------------

字段信息(In_pic)

参数	类型	描述
pos_top	int	输入人脸在原图中最上坐标
pos_bottom	int	输入人脸在原图中最下坐标
pos_left	int	输入人脸在原图中最左坐标
pos_right	int	输入人脸在原图中最右坐标
quality	int	输入人脸的质量分数

参考示例

请求示例

```
Input : files
{
  'img': name='1542712163434643.jpg'
}
"lib_ids":"1",
"threshold":0,
"topk":3,
"msg_id":"772",
"n_topk":1
```

响应示例

```
{
  "code": 0,
  "data": {
    "in_pic": {
      "pos_top": 255,
      "pos_bottom": 400,
      "pos_left": 100,
      "pos_right": 300,
      "quality": 95,
    },
    "n_topn": {
      "img_id": "deab1032-b90a-4d02-9f9f-a1a25907b467",
      "img_path": "img/1_deab1032-b90a-4d02-9f9f-a1a25907b467.jpg",
      "lib_id": 1,
      "lib_name": "blacklib",
      "lib_type": 1,
      "ranking": 1,
      "similarity": 36,
```

```

    "id":4259541481123111,
    "name":张三,
    "gender":0,
    "age":18,
    "address":地址,
  },
  "topn": [
    {
      "img_id": "deab1032-b90a-4d02-9f9f-a1a25907b467",
      "img_path": "img/1_deab1032-b90a-4d02-9f9f-a1a25907b467.jpg",
      "lib_id": 1,
      "lib_name": "blacklib",
      "lib_type": 1,
      "ranking": 1,
      "similarity": 36,
      "id":4259541481123111,
      "name":张三,
      "gender":0,
      "age":18,
      "address":地址 1
    },
    {
      "img_id": "a068b878-95cb-4aa7-aba6-af95cd5b36b3",
      "img_path": "img/1_a068b878-95cb-4aa7-aba6-af95cd5b36b3.jpg",
      "lib_id": 1,
      "lib_name": "blacklib",
      "lib_type": 1,
      "ranking": 2,
      "similarity": 33,
      "id":4259541481123111,
      "name":李四,
      "gender":0,
      "age":18,
      "address":地址 2
    },
    {
      "img_id": "fb7dcd8b-bfee-4e47-b7cd-5cc4e6d09d60",
      "img_path": "img/1_fb7dcd8b-bfee-4e47-b7cd-5cc4e6d09d60.jpg",
      "lib_id": 1,
      "lib_name": "blacklib",

```

```

        "lib_type": 1,
        "ranking": 3,
        "similarity": 31,
        "pos_top": 255,
        "pos_bottom": 400,
        "pos_left": 100,
        "pos_right": 300,
        "quality": 95,
        "id": 4259541481123111,
        "name": "王五",
        "gender": 0,
        "age": 18,
        "address": "地址 3"
    }
]
},
"msg": ""
}

```

5.2 1:1 人脸图片比对

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	单张人脸与单张人脸图片比对

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“771”	是	
img_1	file	需要输入的第一张人脸	是	
img_2	file	需要输入的第二张人脸	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为成功）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
Facelmg1to1Rsp_Score	int	返回的两个人脸的相似度值

参考示例

请求示例

```
Input : files
{
  "msg_id": "771"
  'img_1': name='1542712163434643.jpg'
  'img_2': name='54646644544635.jpg'
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "Facelmg1to1Rsp_Score": 30
  },
  "msg": ""
}
```

5.3 1:N 人脸图片(base64)比对

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	1:N 人脸图片比对，输入单张图片与指定人脸库进行比对，支持返回 Top1-Top50，支持返回人脸坐标和人脸质量分数，以及 N 个人的属性(姓名，年龄，性别，ID)。

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“780”	是	
threshold	int	比对阈值，范围：[0,100]，默认 0	否	
img	json	需要比对的人脸图片	是	
lib_ids	string	比对目标人脸库,可	是	

		输入多个，最多 50 个，库 id 用 “,” 进行分隔		
topk	int	需返回的 topn 比对结果数量，最多可返回 top50，范围：[1,50]，默认 1	否	
n_topk	int	返回 topn 的第 n 个结果，范围：[1,topk]，默认 1	否	

字段信息(img)

参数	类型	描述
filename	string	图片文件名
data	string	base64 编码的图片数据

响应参数

参数	类型	描述
code	int	结果码（0 即为成功）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
n_topn	json	topn 个比对结果中的第 n 个，由请求参数 n_topk 决定
topn	json	前 topn 个比对结果
ln_pic	json	输入人脸在原图属性

字段信息(n_topn)

参数	类型	描述
img_id	string	名次对应的底库人脸图 img_id
img_path	string	名次对应的底库人脸图路径
lib_id	int	名次对应的库 id
lib_name	string	名次对应的库名
lib_type	int	名次对应的人脸库类型
ranking	int	比对名次
similarity	int	名次对应的人脸图比对相似度系数[0,100]
id	string	名次对应的底库人脸图证件号

name	string	名次对应的底库人脸图姓名
gender	string	名次对应的底库人脸图性别（0：女，1：男）
age	string	名次对应的底库人脸图年龄
address	string	名次对应的底库人脸图地址

字段信息(topn)

参数	类型	描述
img_id	string	名次对应的底库人脸图 imgid
img_path	string	名次对应的底库人脸图路径
lib_id	int	名次对应的库 id
lib_name	string	名次对应的库名
lib_type	int	名次对应的人脸库类型
ranking	int	比对名次
similarity	int	名次对应的人脸图比对相似度系数[0,100]
id	string	名次对应的底库人脸图 ID
name	string	名次对应的底库人脸图姓名
gender	string	名次对应的底库人脸图性别（0：女，1：男）
age	string	名次对应的底库人脸图年龄
address	string	名次对应的底库人脸图地址

字段信息(In_pic)

参数	类型	描述
pos_top	int	输入人脸在原图中最上坐标
pos_bottom	int	输入人脸在原图中最下坐标
pos_left	int	输入人脸在原图中最左坐标
pos_right	int	输入人脸在原图中最右坐标
quality	int	输入人脸的质量分数

参考示例

请求示例

```
Input : files
{
  "msg_id":"780",
  "lib_ids":"1",
  "threshold":0,
  "topk":3,
  "n_topk":1,
```

```
"img": {
  "filename": "haha.jpg",
  "data": ""
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "in_pic": {
      "pos_top": 255,
      "pos_bottom": 400,
      "pos_left": 100,
      "pos_right": 300,
      "quality": 95,
    },
    "n_topn": {
      "img_id": "deab1032-b90a-4d02-9f9f-a1a25907b467",
      "img_path": "img/1_deab1032-b90a-4d02-9f9f-a1a25907b467.jpg",
      "lib_id": 1,
      "lib_name": "blacklib",
      "lib_type": 1,
      "ranking": 1,
      "similarity": 36,
      "id": 4259541481123111,
      "name": "张三",
      "gender": 0,
      "age": 18,
      "address": "地址",
    },
    "topn": [
      {
        "img_id": "deab1032-b90a-4d02-9f9f-a1a25907b467",
        "img_path": "img/1_deab1032-b90a-4d02-9f9f-a1a25907b467.jpg",
        "lib_id": 1,
        "lib_name": "blacklib",
        "lib_type": 1,
        "ranking": 1,
        "similarity": 36,
        "id": 4259541481123111,
      }
    ]
  }
}
```

```

    "name":张三,
    "gender":0,
    "age":18,
    "address":地址 1
  },
  {
    "img_id": "a068b878-95cb-4aa7-aba6-af95cd5b36b3",
    "img_path": "img/1_a068b878-95cb-4aa7-aba6-af95cd5b36b3.jpg",
    "lib_id": 1,
    "lib_name": "blacklib",
    "lib_type": 1,
    "ranking": 2,
    "similarity": 33,
    "id":4259541481123111,
    "name":李四,
    "gender":0,
    "age":18,
    "address":地址 2
  },
  {
    "img_id": "fb7dcd8b-bfee-4e47-b7cd-5cc4e6d09d60",
    "img_path": "img/1_fb7dcd8b-bfee-4e47-b7cd-5cc4e6d09d60.jpg",
    "lib_id": 1,
    "lib_name": "blacklib",
    "lib_type": 1,
    "ranking": 3,
    "similarity": 31,
    "pos_top":255,
    "pos_bottom":400,
    "pos_left":100,
    "pos_right":300,
    "quality":95,
    "id":4259541481123111,
    "name":王五,
    "gender":0,
    "age":18,
    "address":地址 3
  }
]
},

```

```
"msg": ""  
}
```

5.4 1:1 人脸图片(base64)比对

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	单张人脸与单张人脸图片比对

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“781”	是	
img_1	json	需要输入的第一张人脸 img 格式	是	
img_2	json	需要输入的第二张人脸 img 格式	是	

字段信息(img)

参数	类型	描述
filename	string	图片文件名
data	string	base64 编码的图片数据

响应参数

参数	类型	描述
code	int	结果码 (0 即为成功)
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
FaceImg1to1Rsp_Score	int	返回的两个人脸的相似度值

参考示例

请求示例

```
Input : files  
{  
  "msg_id": "781",  
  "img_1": {
```

```

    "filename": "haha1.jpg",
    "data": ""
  }
  "img_2": {
    "filename": "haha2.jpg",
    "data": ""
  }
}

```

响应示例

```

{
  "code": 0,
  "data": {
    "Facelmg1to1Rsp_Score": 30
  },
  "msg": ""
}

```

6 系统配置接口

6.1 查询 NTP 配置信息

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	该接口用于查询 NTP 时间同步配置信息

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1291”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
ntp_cycle	string	同步周期，默认 10 分钟
ntp_disable	string	是否禁用 NTP，0 不禁用，1 禁用
port	string	NTP 服务器端口
url	string	NTP 服务器 url

参考示例

请求示例

```
{
  "msg_id":"1291"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "ntp_cycle": "10",
    "ntp_disable": "0",
    "port": "123",
    "url": "edu.ntp.org.cn"
  },
  "msg": ""
}
```

6.2 修改 NTP 配置信息

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	该接口用于修改 NTP 时间同步配置信息

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1290”	是	
url	string	NTP 服务器 url， 默认 pool.ntp.org	否	
ntp_cycle	string	同步周期，默认 10 分钟	否	10

ntp_disable	string	是否禁用 NTP，0 不禁用，1 禁用，默认 1	否	
-------------	--------	--------------------------	---	--

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1290",
  "ntp_disable":"0"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.3 版本信息查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	版本信息查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1281”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
device_id	string	设备 id
serial_id	string	序列号
version	string	服务器版本信息
web_version	string	Web 版本信息

参考示例

请求示例

```
{
  "msg_id": "1281"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "device_id": "666",
    "serial_id": "STSNMAWX08A118110001",
    "version": "SenseNebula-M-V1.0.2-20190122",
    "web_version": "SenseNebulaWeb-M-V1.0.2-web-20190122"
  },
  "msg": ""
}
```

6.4 存储策略查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	存储策略查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1288”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据

msg	string	结果描述
-----	--------	------

字段信息(data)

参数	类型	描述
cur_storage	int	已占用存储
max_storage	int	最大存储
storage_strategy	int	存储策略（1：满覆盖，2：满即停）

参考示例

请求示例

```
{
  "msg_id":"1288"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "cur_storage": 0,
    "max_storage": 2048,
    "storage_strategy": 1
  },
  "msg": ""
}
```

6.5 存储策略修改

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	存储策略修改

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1287”	是	
storage_strategy	int	存储策略（1：满覆盖，2：满即停）	是	1 或 2

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1287",
  "storage_strategy":1
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.6 系统时间查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	系统时间查询

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1283”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
sys_time	string	系统时间

time_format	string	时间格式
time_zone	string	时区

参考示例

请求示例

```
{
  "msg_id": "1283"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "sys_time": "2019-01-23 14:51:18",
    "time_format": "yyyy-MM-dd HH:mm:ss",
    "time_zone": "GMT+08:00"
  },
  "msg": ""
}
```

6.7 系统时间设置

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	系统时间设置

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1282”	是	
sys_time	string	系统时间	是	2019-01-14 12:23:45
time_zone	string	时区	是	GMT+08:00
time_format	string	时间格式，支持三种格式（年-月-日/日-月-年/月-日-年）	否	yyyy-MM-dd HH:mm:ss

响应参数

参数	类型	描述
----	----	----

code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1282",
  "sys_time": "2019-01-14 12:23:45",
  "time_zone": "GMT+08:00",
  "time_format": "yyyy-MM-dd HH:mm:ss"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.8 网络查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	网络查询接口

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1297”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
default_info	json	返回数据

ethernet	json	网络配置信息
----------	------	--------

字段信息(default_info)

参数	类型	描述
dns_method	string	dns 获取方式
ethname	string	网卡名
first_dns	string	首选 dns
second_dns	string	备选 dns

字段信息(ethernet)

参数	类型	描述
ethname	string	网卡名
gateway	string	网关
hw_addr	string	MAC 地址
ip	string	IP
ip_method	string	IP 获取方式 (static/dhcp)
ip_version	string	IP 版本
netmask	string	子网掩码

参考示例

请求示例

```
{
  "msg_id": "1297"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "default_info": {
      "dns_method": "static",
      "ethname": "eth3",
      "first_dns": "10.5.4.207",
      "second_dns": "10.5.4.205"
    },
    "ethernet": [
      {
        "ethname": "eth0",
        "gateway": "10.5.2.1",
        "hw_addr": "00:04:4b:a5:4d:b5",
        "ip": "10.5.2.56",
```

```

        "ip_method": "static",
        "ip_version": "IPv4",
        "netmask": "255.0.0.0"
    },
    {
        "ethname": "eth1",
        "gateway": "",
        "hw_addr": "00:07:32:4e:68:44",
        "ip": "192.168.2.10",
        "ip_method": "static",
        "ip_version": "IPv4",
        "netmask": "255.255.255.0"
    },
    {
        "ethname": "eth2",
        "gateway": "",
        "hw_addr": "00:07:32:4e:68:45",
        "ip": "192.168.3.10",
        "ip_method": "static",
        "ip_version": "IPv4",
        "netmask": "255.255.255.0"
    },
    {
        "ethname": "eth3",
        "gateway": "",
        "hw_addr": "00:07:32:4e:68:46",
        "ip": "192.168.4.10",
        "ip_method": "static",
        "ip_version": "IPv4",
        "netmask": "255.255.255.0"
    }
]
},
"msg": ""
}

```

6.9 网络设置

接口描述

接口 url

http://\${ip}:\${port}/api/json

请求方式	POST
请求参数格式	JSON
接口描述	网络设置接口

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1296”	是	
ethname	string	网卡名	是	
ip	string	ip 地址	是	
ip_method	string	ip 获取方式	是	static 或 dhcp
netmask	string	子网掩码	是	
gateway	string	网关	是	
ip_version	string	IP 版本	是	IPv4 或 IPv6

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1296",
  "ethname": "eth3",
  "ip": "192.168.133.120",
  "ip_method": "static",
  "netmask": "255.255.255.0",
  "gateway": "192.168.133.1",
  "ip_version": "IPv4"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.10 默认网口设置

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	默认网口设置接口

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1295”	是	
dns_method	string	DNS 获取方式	是	static 或 dhcp
first_dns	string	首选 DNS	否	
second_dns	string	备用 DNS	否	
ethname	string	网卡名	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1295",
  "dns_method": "static",
  "first_dns": "10.5.4.207",
  "second_dns": "10.5.4.205",
  "ethname": "eth0"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```


6.11 软件升级

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	软件升级

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1292”	是	
file	file	文件	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

备注：软件升级成功后，系统重启，请稍等。

参考示例

请求示例

```
POST /form HTTP/1.1
Host: 10.5.2.53:8080
Content-Type: multipart/form-data; boundary=----WebKitFormBoundary7MA4YWxkTrZu0gW
cache-control: no-cache
Postman-Token: 37201329-0164-414c-85d3-65e821456ea2

Content-Disposition: form-data; name="file";
filename="/home/SENSETIME/liuyang1_vendor/Downloads/install/nebula.tar.bz2
Content-Disposition: form-data; name="msg_id"; msg_id = "1292"
-----WebKitFormBoundary7MA4YWxkTrZu0gW--
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.12 系统重启

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	系统重启

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1294”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1294"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.13 Device ID 查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询 device id

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1301”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
device_id	string	设备 id

参考示例

请求示例

```
{
  "msg_id": "1301"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "device_id": "test_device"
  },
  "msg": ""
}
```

6.14 Device ID 修改

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	修改 device id

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1300”	是	
device_id	string	设备 id	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1300",
  "device_id":"test_device"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.15 恢复默认配置

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	恢复默认配置

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1317”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	string	NULL
msg	string	结果描述

参考示例

请求示例

```
{
```

```
"msg_id":"1317"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.16 开启 web 访问功能

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	开启 web 页面访问功能

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1315”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	string	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1315"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.17 关闭 web 访问功能

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	关闭 web 页面访问功能

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1316”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	string	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1316"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

6.18 查询 CPU 信息

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	请求查询 cpu 信息

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1313”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
cpu_type	string	cpu 类型
cpu_usage	double	cpu 使用百分比
kernel_num	int	cpu 核数量

参考示例

请求示例

```
{
  "msg_id": "1313"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "cpu_type": "ARMv8 Processor rev 3 (v8l)",
    "cpu_usage": 14.450867052023122,
    "kernel_num": 6
  },
  "msg": ""
}
```

6.19 查询内存信息

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	请求查询内存信息

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1314”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
memory_usage	double	内存使用百分比
total_memory	int	内存总大小

参考示例

请求示例

```
{
  "msg_id": "1314"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "memory_usage": 78.615106141826942,
    "total_memory": 8232378368
  },
  "msg": ""
}
```

7 HTTP 配置接口

7.1 HTTP 配置信息查询

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON

接口描述	查询 HTTP 配置信息
------	--------------

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1299”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
key	string	Http Key
url	string	Http Url

参考示例

请求示例

```
{
  "msg_id": "1299"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "key": "db504129-85e0-49ca-8626-f639b8d09846",
    "url": "http://sgdzipic.3322.org:8088/agbox/device/snap"
  },
  "msg": ""
}
```

7.2 HTTP 配置信息修改

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	修改 HTTP 配置信息

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1298”	是	
url	string	http url	是	google.com
key	string	http key	是	google-key

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1298",
  "url":"google.com",
  "key":"google-key"
}
```

响应示例

```
{
  "code":0,
  "data": "",
  "msg": ""
}
```

7.3 HTTP 心跳信息推送接口

接口描述

接口 url	用于接收推送心跳信息的 HTTP 服务器 URL
接口描述	该接口用于将心跳信息推送到用户指定 ip 地址的服务器。

响应参数

参数	类型	描述
msg_id	string	“775”
data	json	返回数据

字段信息(data)

参数	类型	描述
----	----	----

trigger	string	心跳信息对应的时间
device_id	string	心跳信息对应的设备 id

响应示例

```
{
  "data": {
    "device_id": "test_device",
    "trigger": "2019-01-24 15:05:05"
  },
  "msg_id": 775
}
```

7.4 HTTP 布控结果推送接口

接口描述

接口 url	用于接收推送结果的 HTTP 服务器 URL
接口描述	该接口用于将布控结果推送到用户指定 ip 地址的服务器。

响应参数

参数	类型	描述
key	string	对接设备验证时使用
json	json	返回数据
snap	file	抓拍人脸/人体小图
img	file	比中的底库图片
snap_frame	file	人脸/人体小图对应的场景大图

字段信息(json)

参数	类型	描述
msg_id	string	“774”
data	json	返回数据

字段信息(data)

参数	类型	描述
camera_name	string	相机名称
device_id	string	设备 ID
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]
position	string	相机位置
threshold	int	比对阈值（与相机绑定），范围：[0,100]；

img_id	string	比中的库内人脸图片 id
img_path	string	底库图片路径（当前为保留字段，指向图片名称）下载图片需拼接 url： http://\${ip}:\${port}/ws/img_path
lib_id	int	库 id
lib_name	string	库名称
lib_type	int	库类型（1：黑名单，2：白名单）
person_addr	string	地址
person_age	string	年龄
person_gender	string	性别（0：女，1：男）
person_idcard	string	身份标识码
person_name	string	姓名
snap_id	string	抓拍图片 id
similarity	int	与底库图片相似度分数，范围： [0,100]；默认值为-1
quality	int	质量分数，范围:[0,100]
pos_top	int	人脸坐标左上 Y
pos_bottom	int	人脸坐标右下 Y
pos_left	int	人脸坐标左上 X
pos_right	int	人脸坐标右下 X
snap_feat	string	抓拍图片特征
snap_path	string	抓拍图片路径（当前为保留字段，指向图片名称）；下载图片需拼接 url： http://\${ip}:\${port}/ws/snap_path
trigger	string	触发时间（抓拍时间）
obj_label	int	标识（1：人脸，2：人体）
face_attr	json	obj_label 为 1 时，人脸属性，详见附录 B
body_attr	json	obj_label 为 2 时，人体属性，详见附录 C

备注 1：布控结果推送接收之前，请先配置好相应的 httpserver。

备注 2：该推送结果中，img、snap、snap_frame 为 file 文件。下载底库图片请根据 img_path 进行下载链接拼接，下载抓拍图片请根据 snap_path 进行下载链接拼接。

响应示例

```
{
  "key": "",
  "snap": "",
```

```

"img": "",
"snap_frame": "",
"json": {
  "data": {
    "face_attr": {
      "cap_style": "hat_style_type_none", "gender_code": "male",
      "glass_style": "glasses_style_type_none", "mustache_style": "mustache_style_type_none",
      "respirator_color": "color_type_none",
      "st_age": "st_adult",
      "st_age_value": "24.000000", "st_expression": "st_calm"
    },
    "camera_name": "test", "device_id": "13",
    "channel": 19,
    "img_id": "fa48b56f-9aed-43bc-ad20-9ee164873473",
    "img_path": "img_pic",
    "lib_id": 3,
    "lib_name": "公司人像库",
    "lib_type": 2,
    "obj_label": 1,
    "person_addr": "",
    "person_age": "",
    "person_gender": "",
    "person_idcard": "",
    "person_name": "娄松亚_00000000000000000000_1",
    "position": "",
    "similarity": 96,
    "quality": 1,
    "pos_top": 1000,
    "pos_bottom": 100,
    "pos_left": 100,
    "pos_right": 1000,
    "snap_feat":
    "124.000000,391.000000,280.000000,327.000000,239.000000,239.000000,242.000000,244.000000,",
    "snap_id": "92ed1a18-2aaf-4362-b5b4-1fd6750e1dd9",
    "snap_path": "snap_pic",
    "threshold": 30,
    "trigger": "2019-01-24 13:43:32",
  },
  "msg_id": 774
}

```

```
}
```

8 HTTPS 配置接口

8.1 上传 HTTPS 证书和密钥

接口描述

接口 url	http://\${ip}:\${port}/api/form
请求方式	POST
请求参数格式	FORM
接口描述	上传 HTTPS 证书和密钥

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“x”	是	
file	file	https 文件	是	
file_type	string	https 文件类型： crt:证书；key:密钥	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
Input Files
{
  'file':name='server.crt'
}
data
{
  "msg_id":"1302",
  "file_type":"crt"
}
```

响应示例

```
{
```

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

参考示例

请求示例

```
Input Files
{
  'file':name='server.key'
}
data
{
  "msg_id":"1302",
  "file_type":"key"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

8.2 删除 HTTPS 证书和密钥

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	删除 HTTPS 证书和密钥

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1303”	是	
file_type	string	https 文件类型 crt: 证书; key:密钥	是	“crt” or “key”

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）

data	json	NULL
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id": "1303",
  "file_type": "crt"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

参考示例

请求示例

```
{
  "msg_id": "1303",
  "file_type": "key"
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

8.3 查询 HTTPS 证书和密钥

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询 HTTPS 证书和密钥

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1304”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
https_cert	string	https 认证后的证书
https_key	string	https 私用密钥

参考示例

请求示例

```
{
  "msg_id": "1304"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "https_cert": server.crt,
    "https_key": server.key
  }
  "msg": ""
}
```

8.4 切换为 HTTPS 或 HTTP

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	切换为 HTTPS 或 HTTP

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1305”	是	
prot_mode	string	切换模式 https 或 http	是	“http” or “https”

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	NULL
msg	string	结果描述

备注：HTTPS/HTTP 切换后，系统自动重启，请稍等。

参考示例

请求示例

```
{
  "msg_id": "1305",
  "prot_mode": "http"
}
```

响应示例

```
{
  "code": 0,
  "data": ""
  "msg": ""
}
```

8.5 查询当前为 HTTPS 或 HTTP

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询当前为 HTTPS 或 HTTP

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1306”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回结果
msg	string	结果描述

字段信息(data)

参数	类型	描述
----	----	----

https_stat	string	http 或 https
------------	--------	--------------

参考示例

请求示例

```
{
  "msg_id": "1306"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "https_stat": "http"
  },
  "msg": ""
}
```

9 WebSocket 布控推送相关接口

9.1 WebSocket 查询布控推送密钥

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	WebSocket 查询布控推送密钥

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1286”	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
----	----	----

key	string	订阅密钥
-----	--------	------

参考示例

请求示例

```
{
  "msg_id": "1286"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "key": "04af251e-91d4-43bd-b8b0-8c8c6b3245bb"
  },
  "msg": ""
}
```

9.2 WebSocket 布控结果推送接口

接口描述

接口 url	ws://{ip}:{port}/ws/
接口描述	WebSocket 布控结果推送接口

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“776”	是	
key	string	订阅密钥	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg_id	string	“777”（推图消息唯一识别码）
msg	string	结果描述

字段信息(data)

参数	类型	描述
camera_name	string	相机名称
device_id	string	设备 ID
channel	int	通道；M4s 的范围：[1,8]，M8s 的范围：[1,16]

threshold	int	比对阈值（与相机绑定），范围：[0,100]；
img_id	string	比中的库内人脸图片 id
img	string	底库图片(base64 转码)
img_path	string	底库图片路径，下载图片需拼接 url： http://\${ip}:\${port}/ws/img_path
lib_id	int	库 id
lib_name	string	库名称
lib_type	int	库类型（1：黑名单，2：白名单）
person_addr	string	地址
person_age	string	年龄
person_gender	string	性别（0：女，1：男）
person_idcard	string	身份标识码
person_name	string	姓名
position	string	相机位置
ranking	int	比对相似度排名
similarity	int	相似度分数，范围：[0,100]；默认值为-1
quality	int	质量分数，范围：[0,100]
pos_top	int	人脸坐标左上 Y
pos_bottom	int	人脸坐标右下 Y
pos_left	int	人脸坐标左上 X
pos_right	int	人脸坐标右下 X
snap_id	string	抓拍图片 id
snap_buf	string	抓拍人脸/人体小图(base64 转码)
snap_feat	string	抓拍图片特征, 当 obj_label 为 1/2 分别表示人脸/人体特征
snap_path	string	抓拍图片路径，下载图片需拼接 url： http://\${ip}:\${port}/ws/snap_path
snap_frame	string	抓拍人脸/人体小图对应的场景大图(base64 转码)
trigger	string	触发时间
obj_label	int	标识（1：人脸，2：人体）
face_attr	json	当 obj_label 为 1 时。人脸属性，详见附录 B
body_attr	json	当 obj_label 为 2 时。人体属性，详见附录 C

备注：该推送结果中，img、snap、snap_frame 为 base64 转码。下载底库图片请根据 img_path 进行下载链接拼接，下载抓拍图片请根据 snap_path 进行下载链接拼接。

参考示例

请求示例

```
{
  "key": "7410342e-e4e8-47de-a92f-2fd918eb3d6a",
  "msg_id": "776"
}
```

响应示例

```
{
  "code": 0,
  "msg": ""
}
{
  "code": 0,
  "data": {
    "camera_name": "test",
    "device_id": "123",
    "channel": 1,
    "img_id": "3c370627-d082-4c53-b156-e9380163d22f",
    "img_path": "img/12_3c370627-d082-4c53-b156-e9380163d22f.jpg",
    "lib_id": 12,
    "lib_name": "Marry",
    "lib_type": 1,
    "person_addr": "",
    "person_age": "",
    "person_gender": "",
    "person_idcard": "",
    "person_name": "1201020000000589932",
    "position": "",
    "ranking": 1,
    "similarity": 38,
    "quality": 1,
    "pos_top": 1000,
    "pos_bottom": 100,
    "pos_left": 100,
    "pos_right": 1000,
    "snap_buf": "",
    "snap_feat": "315.000000,32.000000,"
  }
}
```

```

"snap_id": "5672fb61-74ab-4037-8a37-a3c0abf417a3",
"snap_path": "record/5672fb61-74ab-4037-8a37-a3c0abf417a3.jpg",
"threshold": 60,
"trigger": "2019-01-24 14:09:21",
"snap_frame": "",
"obj_label": 1,
"face_attr": {
  "cap_style": "hat_style_type_none",
  "gender_code": "female",
  "glass_style": "glasses_style_type_none",
  "mustache_style": "mustache_style_type_none",
  "respirator_color": "color_type_none",
  "st_age": "st_adult",
  "st_age_value": "26.000000",
  "st_expression": "st_angry"
},
},
"msg": "",
"msg_id": "777"
}

```

10 事件管理接口

10.1 查询相机与继电器绑定关系

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	查询相机与继电器绑定关系

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	"1537"	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）

data	json list	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
id	int	记录 id，相机与继电器绑定关系的唯一标识
camera_channel	int	相机通道
camera_name	string	相机名称
dev_ip	string	继电器 ip
dev_port	int	继电器 port
dev_name	string	继电器名称
dev_tag	string	预留字段
dev_type	string	设备类型，继电器对应 'relay'
work_status	int	继电器在线状态（0：离线；1：在线）

参考示例

请求示例

```
{
  "msg_id": "1537"
}
```

响应示例

```
{
  "code": 0,
  "data": [
    {
      "camera_channel": 1,
      "camera_name": "subway1_1.264",
      "dev_ip": "127.0.0.2",
      "dev_name": "测试 3",
      "dev_port": 5679,
      "dev_tag": "1",
      "dev_type": "relay",
      "work_status": 1,
      "id": 2
    }
  ],
  "msg": ""
}
```


10.2 添加相机与继电器绑定关系

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	添加相机与继电器绑定关系

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1538”	是	
camera_channel	int	相机通道	是	
dev_ip	string	继电器 ip	是	10.0.0.2
dev_port	int	继电器 port 0-65535	是	1234
dev_name	string	继电器名称	否	
dev_tag	string	预留字段	否	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	json	返回数据
msg	string	结果描述

字段信息(data)

参数	类型	描述
id	int	记录 id
camera_channel	int	相机通道
camera_name	string	相机名称
dev_ip	string	继电器 ip
dev_port	int	继电器端口号
dev_name	string	继电器名称
dev_tag	string	预留字段
dev_type	string	设备类型，继电器对应 'relay'

参考示例

请求示例

```
{
  "msg_id": "1538",
```

```
"camera_channel":1,
"dev_ip":"10.0.0.2",
"dev_port":1234,
"dev_name":"测试 3"
}
```

响应示例

```
{
  "code": 0,
  "data": {
    "camera_channel": 1,
    "camera_name": "xxx",
    "dev_ip": "127.0.0.2",
    "dev_name": "测试 3",
    "dev_port": 5679,
    "dev_tag": "1",
    "dev_type": "relay",
    "id": 1
  },
  "msg": ""
}
```

10.3 删除相机与继电器绑定关系

接口描述

接口 url	http://\${ip}:\${port}/api/json
请求方式	POST
请求参数格式	JSON
接口描述	删除相机与继电器绑定关系

请求参数

参数	类型	描述	是否必填	示例
msg_id	string	“1539”	是	
id	int	id 添加绑定关系时返回的 id	是	

响应参数

参数	类型	描述
code	int	结果码（0 即为 OK）
data	string	
msg	string	结果描述

参考示例

请求示例

```
{
  "msg_id":"1539",
  "id":1
}
```

响应示例

```
{
  "code": 0,
  "data": "",
  "msg": ""
}
```

附录 A：错误码

错误码	中文简体	中文繁體	English
-1	未知错误	未知錯誤	Unknown error
-105	数据不存在	資料不存在	Data does not exist
-108	用户不存在	用戶不存在	User does not exist
-109	相机不存在	相機不存在	Camera does not exist
-110	库中没有人脸图片	庫中沒有人臉圖片	No face image in the library
-113	用户名或密码错误	使用者名或密碼錯誤	Username or password error
-114	用户已存在	使用者已存在	User already exists
-115	该通道相机已存在	該通道相機已存在	The channel already has a camera
-116	人脸库图片已存在	人臉庫圖片已存在	The face image already exists
-117	人脸库不存在	人臉庫不存在	The face library does not exist
-118	人脸库数量已达上限	人臉庫數量已達上限	The number of face library has reached the limit
-119	人脸库已存在	人臉庫已存在	The face library already exists
-122	图片数量已达库容上限	圖片數量已達庫容上限	The number of images has reached the limit
-123	该库已和相机绑定，删	該庫已和相機綁定，刪	The library has been

	除前请解绑	除前請解綁	bound to the camera please untie it before deleting
-126	人像库中没有此人脸图 片	人像库中没有此人脸图 片	No such face image in the library
-610	抓拍模式切换失败	抓拍模式切換失敗	Snap mode switch failed
-140	用户没有权限	用戶沒有權限	User does not have permission to operate
-141	用户未登录/session 会 话已过期	用戶未登錄/session 會 話已過期	User is not logged in
-142	用户密码太弱	用戶密碼太弱	User password too weak
-143	角色 id 不存在	角色 id 不存在	Role id not found
-144	用户数量超出限制	用戶數量超出限制	User limit exceed
-145	当前用户名或密码错误	當前用戶名或密碼錯誤	Current user name or password incorrect
-146	该角色下存在用户不可 删除	該角色下存在用戶不可 刪除	Role cannot be deleted cause users exist in the role
-147	角色已存在	角色已存在	Role has already exist
-148	角色数量超出上限	角色數量超出上限	Role limit exceed
-151	算力不足	算力不足	Insufficient computing power
-1100	img_id 已存在	img_id 已存在	'img_id' exists.
-1101	未检测到人脸	未檢測到人臉	No face detected
-1102	人脸图片质量过低	人臉圖片品質過低	The face image quality is too low
-1104	图片格式不正确或图片 破损	圖片格式不正確或圖片 破損	Incorrect image format or broken image
-2100	无内容可导出	無內容可匯出	No content to export
-2754	人脸底库 img_id 太长	人臉底庫 img_id 太長	The img_id is too long
-2824	当前用户不能删除自己	當前用戶不能刪除自己	Current user cannot delete himself
-3596	参数输入错误	參數輸入錯誤	Parameter input error
-3599	msg_id 错误	msg_id 錯誤	'msg_id' error
-3603	升级失败	升級失敗	Upgrade failed
-3606	网络配置错误	網路設定錯誤	Network configuration error

-3607	当前系统已经处于该模式	當前系統已經處於該模式	The current system is already in this mode
-3608	切换模式失败	切換模式失敗	Switch mode failed
-3614	ntp 服务器访问错误	ntp 伺服器訪問錯誤	NTP server access error
-3616	https 模式下不能导入 crt 或 key 文件	https 模式下不能導入 crt 或 key 檔	The crt or key file cannot be imported in HTTPS mode
-2617	https 模式下不能删除 crt 或 key 文件	https 模式下不能刪除 crt 或 key 檔	The crt or key file cannot be deleted in HTTPS mode
-3619	切换 https/http 失败	切換 https/http 失敗	Toggle https/http failed
-3620	图片模式下不能修改抓拍策略	圖片模式下不能修改抓拍策略	The capture strategy cannot be modified in Image mode
-3524	系统内部错误	系統內部錯誤	System internal error
-3625	尚未导入 crt 或 key 文件	尚未導入 crt 或 key 檔	crt or key file not yet imported

备注：

-3606 网络配置错误的原因包括：

1. 与其他网口处于同一网段；2. DHCP 失败；3. 网关配置错误；4. 子网掩码配置错误

附录 B：人脸属性说明

人脸属性	描述	枚举值	描述
st_age	年龄分组	st_old, st_adult, st_child	
mustache_style	胡子	mustache_style_type_none, whiskers	无胡子； 髯
respirator_color	口罩	color_type_none, color_type_other	无口罩； 其他颜色口罩
glass_style	眼镜	glasses_style_type_none, transparent_glasses, sunglasses	无眼镜； 透明眼镜； 太阳镜
gender_code	性别	female, male	女； 男
st_expression	表情	st_angry, st_happy, st_sorrow,	愤怒； 开心； 伤心；

		st_calm, st_surprised, st_scared, st_disgust, st_yawn	沉着; 惊喜; 害怕; 失望; 哈欠
cap_style	帽子类型	hat_style_type_none, cap	无帽子; 有帽子
st_age_value	年龄值	年龄数值	

附录 C：人体属性说明

人体属性	描述	枚举值	描述
st_pedestrian_angle	朝向	st_front, st_side, st_back	正面; 侧面; 背面
gender_code	性别	male, female	男; 女
st_age	年龄分组	st_old, st_adult, st_child	老人; 成年人; 小孩
hair_style	发型	st_short, long, bald	短发; 长发; 秃头
hair_color	头发颜色	black, white, yellow	黑; 白; 黄
coat_style	上衣款式	long_coat, jacket, t_shirt, sports_wears, down_jacket, shirt, dress, business_suit	长款大衣; 夹克; T 恤; 运动服; 羽绒服; 短袖; 连衣裙; 西装
coat_color	上衣颜色	black, white, gray, red, yellow,	黑; 白; 灰; 红; 黄;

		blue, green, purple	蓝; 绿; 紫
coat_length	上衣长度	short_sleeve, long_sleeve	短款; 长款
st_coat_pattern	上衣花案	st_pure, st_stripe, st_design, st_joint, st_lattic	纯色; 条纹; 图案; 拼接; 格子
trousers_len	下衣类型	trousers, shorts, st_skirt	长裤; 短裤; 裙子
trousers_color	下衣颜色	black, white, gray, red, yellow, blue, green, purple	黑; 白; 灰; 红; 黄; 蓝; 绿; 紫
st_trousers_pattern	下衣花案	st_pure, st_stripe, st_design, st_joint, st_lattic	纯色; 条纹; 图案; 拼接; 格子
shoes_style	鞋子类型	leather_shoes, boots, walking_shoes, sandal	皮鞋; 靴子; 休闲鞋; 凉鞋
shoes_color	鞋子颜色	black, white, gray, red, yellow, blue, green, purple	黑; 白; 灰; 红; 黄; 蓝; 绿; 紫
st_bag	箱包	st_bag	
bag_color	箱包颜色	black,	黑;

		white, gray, red, yellow, blue, green, purple	白; 灰; 红; 黄; 蓝; 绿; 紫
bag_style	箱包类型	hand_bag, shoulder_bag, backpack, trolley, bag_style_type_none, waist_pack	手拎包; 单肩包; 双肩包; 拉杆箱; 无包; 腰包
st_umbrella	雨伞	st_umbrella	
umbrella_color	雨伞颜色	black, white, gray, red, yellow, blue, green, purple, transparent	黑; 白; 灰; 红; 黄; 蓝; 绿; 紫; 透明
st_respirator	口罩	st_respirator	
respirator_color	口罩颜色	white, gray, blue, black	白色; 灰色; 蓝色; 黑色
st_hat	帽子	st_hat	
cap_color	帽子颜色	black, white, gray, red, yellow, blue, green, purple	黑; 白; 灰; 红; 黄; 蓝; 绿; 紫
cap_style	帽子类型	hat_style_type_none, bonnet, cap,	无帽子; 无檐帽; 鸭舌帽;

		bucket_hat	渔夫帽
st_hold_object_in_front	胸前抱东西	st_hold_object_in_front	

附录 D：Http/Websocket 布控推送接收示例

1. httpserver 搭建示例

```
#!/usr/bin/env python3

import tornado.ioloop
import tornado.web
import pprint
import sys
import os
import time
import json
import logging
import socket
import datetime

file_handler = logging.FileHandler('/tmp/http_server.log', 'a', encoding='utf-8')
formatter = logging.Formatter("%(asctime)-15s %(message)s")
file_handler.setFormatter(formatter)
logger = logging.getLogger()
logger.addHandler(file_handler)
logger.setLevel(logging.INFO)

# httpserver 的 IP 地址
local_IP = socket.gethostbyname(socket.gethostname())

class MyDumpHandler(tornado.web.RequestHandler):

    def post(self)
        logger.info("=" * 100 + "\n")
        logger.info(datetime.datetime.now())
        logger.info("\n[BODY]")
        try:
            body = self.request.body.decode('utf', 'backslashreplace')
            logger.info("try1:\n%s" % body)
            # logger.info('BODY: \n' + body)
        except BaseException as e:
```

```

logger.info("BaseException1 %s" % e)
try:
    body = self.request.body.decode('utf-8')
    logger.info("try2:\n%s" % body)
    # logger.info('BODY: \n' + body)
except BaseException as e:
    logger.info("BaseException2 %s" % e)
    pass

os.makedirs('/tmp/images', exist_ok=True)
ip = self.request.remote_ip # 星云 M 的 IP 地址
file = self.request.files
rsp_json = self.request.body_arguments["json"][0].decode("utf-8")

#推送数据字段打印
js_dict = json.loads(rsp_json)
if js_dict["msg_id"] == 774:
    response_774_dict = ["camera_name", "device_id", "channel", "position", "img_id", "img_path",
"lib_name", "lib_type", "person_addr", "person_age", "person_gender",
"person_idcard", "person_name", "snap_id", "similarity", "quality", "snap_feat", "snap_path", "trigger"]
    self.assert_resp_in_data(response_774_dict, js_dict["data"])
elif js_dict["msg_id"] == 775:
    response_775_dict = ["device_id", "trigger"]
    self.assert_resp_in_data(response_775_dict, js_dict["data"])
else:
    print(js_dict["msg_id"])

#推送图片写入
t = str(time.time()).replace('.', '')
if file:
    logger.info(file['snap'][0]['filename'] + ' saved')
    fobj = open('/tmp/images/' + ip + 'snap' + t + '.jpg', "wb")
    fobj.write(file['snap'][0]['body'])
    fobj.close()

    logger.info(file['snap_frame'][0]['filename'] + ' saved')
    fobj = open('/tmp/images/' + ip + 'snap_frame' + t + '.jpg', "wb")
    fobj.write(file['snap_frame'][0]['body'])
    fobj.close()
if 'img' in file:

```

```

        logger.info(file['img'][0]['filename'] + ' saved')
        fobj = open('/tmp/images/' + ip + 'image' + t + '.jpg', "wb")
        fobj.write(file['img'][0]['body'])
        fobj.close()

    @staticmethod
    def assert_resp_in_data(resp_dict, resp_data):
        for item in resp_dict:
            assert item in resp_data, print("lost item %s" % item)

if __name__ == "__main__":
    try:
        print("\nHttp server listen on http://" + local_IP +
              ":8880 now ... (/tmp/http_server.log, /tmp/images)")
        tornado.web.Application([(r"/.*", MyDumpHandler), ]).listen(8080)
        tornado.ioloop.IOLoop.instance().start()
    except BaseException as e:
        print(e)
        pass

```

2. Websocket 接收推送结果示例

```

#!/usr/bin/env python3
import time
import websocket
import ssl
import base64
import sys
import logging

IP = '10.5.2.81' #星云 M 的 IP 地址

key = ''

file_handler = logging.FileHandler(
    '/tmp/ws.log', 'a', encoding='utf-8')
formatter = logging.Formatter("%(asctime)-15s %(message)s")
file_handler.setFormatter(formatter)
logger = logging.getLogger()
logger.addHandler(file_handler)
logger.setLevel(logging.INFO)

def on_message(ws, message):

```

```

logger.info(message)
d = json.loads(message)
print(d['msg'])
if 'data' in d:
    # print(dir(d['data']))
    # print(pretty_json(d['data']['img']))
    t = str(time.time()).replace('.', '')
    # print(json.dumps(d))

    response_777_dict = ["camera_name", "device_id", "channel", "img_id", "img_path",
"lib_name", "lib_type", "person_addr", "person_age", "person_gender", "person_idcard",
    "person_name", "position", "ranking", "similarity", "quality", "snap_id", "snap_buf", "snap_feat",
"snap_path", "snap_frame", "trigger"]
    assert_resp_in_data(response_777_dict, d['data'])
    if d['data']['img'] != -1:
        data = d['data']['img'].split(',', 1)
        f = imgdata = base64.b64decode(data[1])
        fobj = open('/tmp/wsimages/' + IP + 'snap' + t + '.jpg', "wb")
        fobj.write(f)
        fobj.close()

def assert_resp_in_data(resp_dict, resp_data):
    for item in resp_dict:
        assert item in resp_data, print(item)

def on_error(ws, error):
    print(error)

def on_close(ws):
    print("### closed ###")

#订阅推送消息
def on_open(ws):
    def run(*args):
        ws.send('{ "key": "%s", "msg_id": "776" }' % key)
        print("thread terminating...")
    thread.start_new_thread(run, ())

if __name__ == "__main__":
    websocket.enableTrace(True)

```

```
ws = websocket.WebSocketApp('ws://' + IP + '/ws/',  
    on_message=on_message,  
    on_error=on_error,  
    on_close=on_close,  
    on_open=on_open)  
ws.run_forever(sslopt={"cert_reqs": ssl.CERT_NONE})
```



www.sensetime.com

技术支持

IDEA-Support@sensetime.com

商业合作

+86-400-900-5986 (周一至周五 09:30-18:00)
business@sensetime.com

关注我们



微信订阅号



官方微博