# 人脸API开发手册（Android）

提供全套基于深度学习方法的人脸技术，
包括人脸检测、特征提取及人脸验证等功能。

SENSETIME

# 前言

我们致力于提供便捷准确的人脸视觉技术（包括但不限于人脸验证，人脸搜索，人脸检测，关键点检测，属性检测等），为移动设备交互、行业摄像头应用等不同场景提供强有力的技术支持。这份文档会为您提供API的详细介绍与基本的开发指南，帮助您更好地使用我们的技术服务。

衷心感谢您对我们技术与产品的信任和支持！


PUBLISHED BY SENSETIME
sensetime.com
*First printing, January 2016*

# 目录

# 第1章 Android接口文档

## 1.1 初始化

**功能：**客户端初始化函数（建议在程序入口出调用，或者在调用其他STAPI之前调用）

**声明：**

```
public STAPI(String apiId, String apiSecret) {
        super();
        this.apiId = apiId;
        this.apiSecret = apiSecret;
        this.webSite = WEBSITE_CN;
}
```

**参数：**

- **apiId:** Sensetime 的API ID
- **apiSecret:** Sensetime 的API SECRET

## 1.2 信息获取

**功能：**获得当前账户的使用信息，包括频率限制以及各种数量限制，建议在程序入口调用，一方面测试调用方法是否正确，一方面也可以验证自己的api_id 和api_secret

**声明：**

```
public JSONObject infoApi() throws STAPIException{
        return requestGet("info", "api");
}
```

**返回值：**当前账户的使用信息

---

**功能：**该API 用于查询异步调用时的HTTP 请求处理情况，请参考文档使用

**声明：**

```
public JSONObject infoTask(String taskId) throws STAPIException{
        STAPIParameters4Get params = new STAPIParameters4Get();
        params.setTaskId(taskId);
        return requestGet("info", "task", params);
}
```

**参数：**

- **taskId:** 必须，有效期为24 小时。异步调用时获得的任务ID

**返回值：**异步调用时的HTTP 请求处理情况

---

**功能：**获得图片的相关信息

**声明：**

```
public JSONObject infoImage(String imageId,Boolean withFaces) throws
    STAPIException{
        STAPIParameters4Get params = new STAPIParameters4Get();
        params.setImageId(imageId);
        if(withFaces!=null){
            params.setWithFaces(withFaces);
        }
        return requestGet("info", "image", params);
}
```

**参数：**

- **imageId:** 必须，图片ID
- **withFaces:** 非必须，是否返回人脸所属图片信息，为true 时返回。默认为false，不返回

**返回值：**STImage 对象的字典

**功能：**获取检测过的人脸的相关信息

**声明：**

```
public JSONObject infoFace(String faceId, Boolean withImage) throws
    STAPIException{
        STAPIParameters4Get params = new STAPIParameters4Get();
        params.setFaceId(faceId);
        if(withImage!=null){
            params.setWithImage(withImage);
        }
        return requestGet("info", "face", params);
}
```

**参数：**

- **faceId:** 必须，人脸ID
- **withImage:** 非必须，是否返回人脸所属图片信息，为true 时返回。默认为false，不返回

**返回值：**STFace对象的字典

---

**功能：**查询该账户目前所拥有的人的信息

**声明：**

```
public JSONObject infoListPersons() throws STAPIException{
        return requestGet("info", "list_persons");
}
```

**返回值：**STPerson 对象字典

---

**功能：**查询该账户所拥有的组

**声明：**

```
public JSONObject infoListGroups() throws STAPIException{
        return requestGet("info", "list_groups");
}
```

**返回值：**STGroup 对象字典

---

**功能：**查询该账户所拥有的人脸集合

**声明：**

```
public JSONObject infoListFacesets() throws STAPIException{
        return requestGet("info", "list_facesets");
}
```

**返回值：**STFaceSet对象字典

**功能：**获取人的信息及其含有的人脸ID

**声明：**

```
public JSONObject infoPerson(String personId) throws STAPIException{
        STAPIParameters4Get params = new STAPIParameters4Get();
        params.setPersonId(personId);
        return requestGet("info", "person",params);
}
```

**参数：**

- **personId:** 必须，人的ID

**返回值：**STPerson 对象的字典

---

**功能：**查询组的相关信息，包括组的名字、组的自定义数据以及组所拥有的人的信息

**声明：**

```
public JSONObject infoGroup(String groupId) throws STAPIException{
        STAPIParameters4Get params = new STAPIParameters4Get();
        params.setGroupId(groupId);
        return requestGet("info", "group",params);
}
```

**参数：**

- **groupId:** 必须，组的ID

**返回值：**STGroup对象的字典

---

**功能：**获取人脸集合的信息及其含有的人脸的ID

**声明：**

```
public JSONObject infoFaceset(String facesetId) throws STAPIException{
        STAPIParameters4Get params = new STAPIParameters4Get();
        params.setFacesetId(facesetId);
        return requestGet("info", "faceset",params);
}
```

**参数：**

- **facesetId:** 必须，人脸集合的ID

**返回值：**STFaceSet对象的字典

## 1.3 人脸检测与分析

**功能：**提供图片，进行人脸检测以及人脸分析

**声明：**

```
public JSONObject faceDetection(Bitmap imageBitmap,Boolean
    withLandmarks106,Boolean withAttributes,Boolean isAutoRotate,String
    userData) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFile(imageBitmap);
        if(withLandmarks106!=null){
            params.setLandmarks106(withLandmarks106);
        }
        if(withAttributes!=null){
            params.setAttributes(withAttributes);
        }
        if(isAutoRotate!=null){
            params.setAutoRotate(isAutoRotate);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("face", "detection", params);
    }

    public JSONObject faceDetection(STAPIParameters4Post params) throws
        STAPIException{
        return requestPost("face", "detection", params);
    }

    public JSONObject faceDetection(Bitmap imageBitmap) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFile(imageBitmap);
        return requestPost("face", "detection", params);
    }

    public JSONObject faceDetection(File imageFile) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFile(imageFile);
        return requestPost("face", "detection", params);
    }

    public JSONObject faceDetection(byte[] imageBytes) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFile(imageBytes);
        return requestPost("face", "detection", params);
    }
```

```
public JSONObject faceDetection(String url) throws STAPIException{
    STAPIParameters4Post params = new STAPIParameters4Post();
    params.setUrl(url);
    return requestPost("face", "detection", params);
}

public JSONObject faceDetection(Bitmap imageBitmap,
    STAPIParameters4Post params) throws STAPIException{
    if (params == null) params = new STAPIParameters4Post();
    params.setFile(imageBitmap);
    return requestPost("face", "detection", params);
}

public JSONObject faceDetection(File imageFile,STAPIParameters4Post
    params) throws STAPIException{
    if (params == null) params = new STAPIParameters4Post();
    params.setFile(imageFile);
    return requestPost("face", "detection", params);
}

public JSONObject faceDetection(byte[] imageBytes,
    STAPIParameters4Post params) throws STAPIException{
    if (params == null) params = new STAPIParameters4Post();
    params.setFile(imageBytes);
    return requestPost("face", "detection", params);
}

public JSONObject faceDetection(String imageUrl,STAPIParameters4Post
     params) throws STAPIException{
    if (params == null) params = new STAPIParameters4Post();
    params.setUrl(imageUrl);
    return requestPost("face", "detection", params);
}
```

**参数：**对于可选参数（如isAutoRotate，withLandmarks106和withAttributes），
如果没有需求，请不要开启，这样会减少系统计算时间

- **imageBitmap:** 必须，格式必须为JPG（JPEG），BMP，PNG，GIF，TIFF 之一，宽和高必须大于8px，小于等于4000px，文件尺寸小于等于5 MB，上传本地需上传的图片文件
- **withLandmarks106:** 非必须，值为true 时，计算106 个关键点
- **withAttributes:** 非必须，值为true 时，提取人脸属性
- **isAutoRotate:** 非必须，值为true 时，对图片进行自动旋转
- **userData:** 非必须，用户自定义信息

**返回值：**STImage 对象

**功能：**一对一人脸对比

**声明：**

```
public JSONObject faceVerificationByFace(String faceId,String face2Id)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFace2Id(face2Id);
        return requestPost("face", "verification", params);
}
```

**参数：**

- **faceId:** 人脸ID
- **face2Id:** 人脸2 的ID, 脸与脸对比返回该字段

**返回值：**对比的分数，大于0.6 时判断为同一人

---

**功能：**将一个人脸与另外一个人进行对比

**声明：**

```
public JSONObject faceVerificationByPerson(String faceId,String personId
    ) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setPersonId(personId);
        return requestPost("face", "verification", params);
    }
```

**参数：**

- **faceId:** 人脸ID
- **personId:** 人的ID, 脸与人对比返回该字段

**返回值：**对比的分数，大于0.6 时判断为同一人

---

**功能：**在众多人脸中搜索出与目标人脸最为相似的一张或者多张人脸

**声明：**

```
public JSONObject faceSearch(String faceId,String[] faceIds,int top)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFaceIds(faceIds);
        if(top>=1 && top<=10){
            params.setTop(top);
        }
        return requestPost("face", "search", params);
}
```

**参数：**

- **faceId:** 人脸ID
- **faceIds:** 人脸ID 数组
- **top:** 非必须，最多返回top 个相似人脸，值为1-10，默认为3

**返回值：**相似的人脸字典

**功能：**在众多人脸中搜索出与目标人脸最为相似的一张或者多张人脸

**声明：**

```
public JSONObject faceSearch(String faceId,String facesetId,int top)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFacesetId(facesetId);
        if(top>=1 && top<=10){
            params.setTop(top);
        }
        return requestPost("face", "search", params);
    }

    public JSONObject faceSearch(STAPIParameters4Post params) throws
        STAPIException{
        return requestPost("face", "search", params);
    }

    public JSONObject faceSearch(String faceId,String[] faceIds) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFaceIds(faceIds);
        return requestPost("face", "search", params);
    }

    public JSONObject faceSearch(String faceId,String facesetId) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFacesetId(facesetId);
        return requestPost("face", "search", params);
    }

    public JSONObject faceSearch(String faceId,String[] faceIds,
        STAPIParameters4Post params) throws STAPIException{
        if (params == null) params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFaceIds(faceIds);
        return requestPost("face", "search", params);
    }

    public JSONObject faceSearch(String faceId,String facesetId,
        STAPIParameters4Post params) throws STAPIException{
        if (params == null) params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setFacesetId(facesetId);
        return requestPost("face", "search", params);
    }
```

```
    public JSONObject faceIdentification(STAPIParameters4Post params)
        throws STAPIException{
        return requestPost("face", "identification", params);
    }

    public JSONObject faceIdentification(String faceId,String groupId)
        throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setGroupId(groupId);
        return requestPost("face", "identification", params);
    }
```

**参数：**

- **faceId:** 人脸ID
- **facesetId:** 人脸集合ID
- **top:** 非必须，最多返回top 个相似人脸，值为1-10，默认为3

**返回值：**相似的人脸字典，包括人脸ID 以及置信度

---

**功能：**将一个目标人脸与某个组中的所有人进行对比

**声明：**

```
public JSONObject faceIdentification(String faceId,String groupId,
    Integer top) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setGroupId(groupId);
        if(top>=1 && top<=5){
            params.setTop(top);
        }
        return requestPost("face", "identification", params);
    }

    public JSONObject faceIdentification(String faceId,String groupId,
        STAPIParameters4Post params) throws STAPIException{
        if (params == null) params = new STAPIParameters4Post();
        params.setFaceId(faceId);
        params.setGroupId(groupId);
        return requestPost("face", "identification", params);
    }

    public JSONObject faceTraining(STAPIParameters4Post params) throws
        STAPIException{
        return requestPost("face", "training", params);
    }
```

**参数：**

- **faceId:** 人脸ID
- **groupId:** 组的ID
- **top:** 非必须，最多返回top 个人，值为1-5，默认为2

**返回值：**当搜索出的人与目标人脸的置信度大于0.6 时candidates 中才会有结果返回

**功能：**对人脸、人、人脸集合、组进行训练

**声明：**

```
public JSONObject faceTraining(String[] faceIds,String[] personIds,
    String[] facesetIds,String[] groupIds) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        if(faceIds!=null && faceIds.length>0){
            params.setFaceId(faceIds);
        }
        if(personIds!=null && personIds.length>0){
            params.setPersonId(personIds);
        }
        if(facesetIds!=null && facesetIds.length>0){
            params.setFacesetId(facesetIds);
        }
        if(groupIds!=null && groupIds.length>0){
            params.setGroupId(groupIds);
        }
        return requestPost("face", "training", params);
    }

    public JSONObject faceTrainingByFaceIds(String[] faceIds) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFaceId(faceIds);
        return requestPost("face", "training", params);
    }

    public JSONObject faceTrainingByPersonIds(String[] personIds) throws
         STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setPersonId(personIds);
        return requestPost("face", "training", params);
    }

    public JSONObject faceTrainingByFacesetIds(String[] facesetIds)
        throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFacesetId(facesetIds);
        return requestPost("face", "training", params);
    }

    public JSONObject faceTrainingByGroupIds(String[] groupIds) throws
        STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setGroupId(groupIds);
        return requestPost("face", "training", params);
    }
```

**参数：**请求参数face_id、person_id、faceset_id、group_id 至少存在一个，可多选

- **faceIds:** 人脸ID 数组，多个人脸信息时以逗号分隔
- **personIds:** 人的ID 数组，多个人时以逗号分隔
- **facesetIds:** 人脸集合ID 数组，多个人脸集合时以逗号分隔
- **groupIds:** 组的ID 数组，多个组时以逗号分隔

**返回值：**status OK成功，否则失败

## 1.4 人的管理

**功能：**创建一个人。创建人时，可以同时为其添加人脸信息，也可以加上自定义信息

**声明：**

```
public JSONObject personCreate(String name,String[] faceIds,String
    userData) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setName(name);
        if(faceIds!=null && faceIds.length>0){
            params.setFaceId(faceIds);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("person", "create",params);
    }

    public JSONObject personCreate(String name,STAPIParameters4Post
        params) throws STAPIException{
        if (params == null) params = new STAPIParameters4Post();
        params.setName(name);
        return requestPost("person", "create",params);
    }
```

**参数：**

- **name:** 必须，人名
- **faceIds:** 非必须，人脸的ID数组，一个face_id 只能添加到一个person 中
- **userData:** 非必须，用户自定义信息

**返回值：**JSON类型的Person对象

---

**功能：**删除一个人,删除人后，该人所拥有的人脸ID 若未过期，则人脸ID 不会失效

**声明：**

```
public JSONObject personDelete(String personId) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setPersonId(personId);
        return requestPost("person", "delete",params);
    }
```

**参数：**

- **personId:** 要删除的人的ID

**返回值：**status OK成功，否则失败

**功能：**人创建成功后，调用本API，可以为该人添加人脸信息

**声明：**

```
public JSONObject personAddFace(String personId,String[] faceIds) throws
    STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setPersonId(personId);
        params.setFaceId(faceIds);
        return requestPost("person", "add_face",params);
    }
```

**参数：**

- **personId:** 要添加的人的ID
- **faceIds:** 人脸的ID 数组

**返回值：**status OK成功，否则失败

---

**功能：**移除人脸，使其不再属于这个人

**声明：**

```
public JSONObject personRemoveFace(String personId,String[] faceIds)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setPersonId(personId);
        params.setFaceId(faceIds);
        return requestPost("person", "remove_face",params);
    }
```

**参数：**

- **personId:** 要移除的人的ID
- **faceIds:** 人脸的ID 数组

**返回值：**status OK成功，否则失败

---

**功能：**用于修改person 的user_data 和name 信息

**声明：**

```
public JSONObject personChange(String personId,String name,String
    userData) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setPersonId(personId);
        if(!isNull(name)){
            params.setName(name);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("person", "change",params);
    }
```

**参数：** name 和user_data 两者中至少存在一个

- **personId:** 人的ID
- **name:** 人名
- **userData:** 用户自定义数据

**返回值：**status OK成功，否则失败

## 1.5 组的管理

**功能：**创建一个组，创建组的同时可以向该组中添加人

**声明：**

```
public JSONObject groupCreate(String name,String[] personIds,String
    userData) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setName(name);
        if(personIds!=null && personIds.length > 0){
            params.setPersonId(personIds);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("group", "create",params);
    }
```

**参数：**

- **name:** 组名
- **personIds:** 非必须，STPerson 数组，多个数据时用逗号分隔
- **userData:** 非必须，用户自定义信息

**返回值：**JSON类型的Group对象

---

**功能：**删除一个组

**声明：**

```
public JSONObject groupDelete(String groupId) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setGroupId(groupId);
        return requestPost("group", "delete",params);
    }
```

**参数：**

- **groupId:** STGroup 对象中的strGroupID

**返回值：**status OK成功，否则失败

---

**功能：**组创建成功后，调用本API，可以为组添加人(一个人可以属于多个组)

**声明：**

```
public JSONObject groupAddPerson(String groupId,String[] personIds)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setGroupId(groupId);
        params.setPersonId(personIds);
        return requestPost("group", "add_person",params);
    }
```

**参数：**

- **groupId:** STGroup 对象中的strGroupID
- **personIds:** STPerson 中strPersonID 的数组（如果其中有一个或多个person_id 无效时，那么其他的person_id 也会添加失败）

**返回值：**status OK成功，否则失败

**功能：**移除组中的人

**声明：**

```
public JSONObject groupRemovePerson(String groupId,String[] personIds)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setGroupId(groupId);
        params.setPersonId(personIds);
        return requestPost("group", "remove_person",params);
    }
```

**参数：**

- **groupId:** STGroup 对象中strGroupID
- **personIds:** STPerson 中strPersonID 的数组

**返回值：**status OK成功，否则失败

---

**功能：**修改group 的user_data 和name 信息

**声明：**

```
public JSONObject groupChange(String groupId,String name,String userData
    ) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setGroupId(groupId);
        if(!isNull(name)){
            params.setName(name);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("group", "change",params);
    }
```

**参数：**请求参数name、user_data 两者中至少存在一个

- **groupId:** STGroup 对象中strGroupID 组的ID
- **name:** 组名
- **userData:** 用户自定义数据

**返回值：**status OK成功，否则失败

## 1.6 人脸集合的管理

**功能：**创建一个人脸集合

**声明：**

```
public JSONObject facesetCreate(String name,String[] faceIds,String
    userData) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setName(name);
        if(faceIds!=null && faceIds.length > 0){
            params.setFaceId(faceIds);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("faceset", "create",params);
    }
```

**参数：**

- **name:** 人脸集合名
- **faceIds:** 非必须，人脸的ID
- **userData:** 非必须，用户自定义信息

**返回值：**JSON类型的faceset对象

---

**功能：**删除一个人脸集合

**声明：**

```
public JSONObject facesetDelete(String facesetId) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFacesetId(facesetId);
        return requestPost("faceset", "delete",params);
    }
```

**参数：**

- **facesetId:** 要删除的人脸集合的ID

**返回值：**status OK成功，否则失败

---

**功能：**向人脸集合中添加人脸

**声明：**

```
public JSONObject facesetAddFace(String facesetId,String[] faceIds)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFacesetId(facesetId);
        params.setFaceId(faceIds);
        return requestPost("faceset", "add_face",params);
    }
```

**参数：**

- **facesetId:** 人脸集合的ID
- **faceIds:** 人脸的ID 数组

**返回值：**status OK成功，否则失败

**功能：**移除人脸集合所含有的人脸，使其不再属于该人脸集合

**声明：**

```
public JSONObject facesetRemoveFace(String facesetId,String[] faceIds)
    throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFacesetId(facesetId);
        params.setFaceId(faceIds);
        return requestPost("faceset", "remove_face",params);
    }
```

**参数：**

- **facesetId:** 人脸集合的ID
- **faceIds:** 人脸的ID 数组

**返回值：**status OK成功，否则失败

---

**功能：**修改人脸集合的user_data 和name 信息

**声明：**

```
public JSONObject facesetChange(String facesetId,String name,String
    userData) throws STAPIException{
        STAPIParameters4Post params = new STAPIParameters4Post();
        params.setFacesetId(facesetId);
        if(!isNull(name)){
            params.setName(name);
        }
        if(!isNull(userData)){
            params.setUserData(userData);
        }
        return requestPost("faceset", "change",params);
    }
```

**参数：**请求参数name 与user_data 二者中至少存在一个

- **facesetId:** 人脸集合的ID(STFaceSet的strFaceSetID)
- **name:** 人脸集合名
- **userData:** 用户自定义数据

**返回值：**status OK成功，否则失败

## 1.7 返回代码

**部分HTTP错误代码：**

- **200:** 确定。客户端请求已成功
- **400:** 错误的请求(最常见的错误，通常是提供的参数（数量、类型）不符合要求，或者操作内容无效)
- **401:** 用户验证错误(未授权)
- **403:** 禁止访问(可能是本用户没有权限调用该API，或者超过了调用限额)
- **404:** 未找到(没有找到API，通常是URL 写错了)
- **413:** 请求体过大(通常是上传的文件过大导致)
- **5xx:** 内部服务器错误(其中500 错误最常见)

---

**返回状态代码（API错误类型）：**

- **OK:** 正常
- **UNAUTHORIZED:** 账号或密钥错误
- **KEY_EXPIRED:** 账号过期，具体情况见reason 字段内容
- **RATE_LIMIT_EXCEEDED:** 调用频率超出限额
- **NOT_FOUND:** 请求路径错误
- **INTERNAL_ERROR:** 服务器内部错误
- **INVALID_XXXX:** XXXX 无效

## 1.8 GitHub 仓库

**更多代码内容请前往以下链接查阅：**

https://github.com/SenseTimeApp/STAPI-Android

关于Cloud API更详尽的调用方法、返回错误等描述，请参考文档《云平台接口开发手册》