# Android 自动化

运行在 **Android设备**(如:华为,小米,Vivo,Oppo,魅族,一加等) 上的自动化小程序

通过编程,可快速实现 **免Root** App自动化测试 , 游戏自动化测试 , AI目标检测,办公自动化,数据爬虫 等小程序.

## ‘遥遥领先’的软件平台[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "遥遥领先的软件平台" \o "Direct link to ‘遥遥领先’的软件平台)

编程:

**Python编程,Pip模块拓展,Pycharm开发,WebIDE开发,VsCode开发**

功能:

**免Root,脱机运行,点击,滑动,文本输入**, **控件检索,多点找色,文字识别,全分辨率找图,目标检测**,**云数据库**,**WebUI** 等..

企业服务:

**卡密系统**,**打包独立APP服务**,**云端控制**

# 下载

**仅需安装APP,即可** 开发 **或** 运行自动化程序了

* 手机浏览器 **扫描二维码** 进行安装

| **最新版APP文件** | **备注** |
| --- | --- |
| [AScript-32位.apk](http://py.airscript.cn/getApk" \t "F:/cs01/cs01/cj2/_blank) | 部分模拟器只支持32位的APP,较旧的设备 |
| [AScript-64位.apk](https://py.airscript.cn/getApk?sys=2" \t "F:/cs01/cs01/cj2/_blank) | 绝大对数手机同时支持32,64位. **但部分模拟器不支持64位** |

## 更新历史[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "更新历史" \o "Direct link to 更新历史)

Loading...

# 准备事项

对于初次接触AScript 的开发人员

## 您需要准备[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "您需要准备" \o "Direct link to 您需要准备)

Windows或Mac 电脑一台: 用于代码编写

Android 设备 或模拟器一台: 用于实时测试程序

我们为您准备了 一些 **新手任务**.

1. [第一个小程序](F:/cs01/cs01/cj2/docs/android/guide/first) (约耗时:5分钟)
2. [学习Python基础语法](F:/cs01/cs01/cj2/docs/android/guide/python) (约耗时:4小时)
3. [学习AScript Api](F:/cs01/cs01/cj2/docs/android/guide/api)(约耗时:4小时)
4. [发布小程序](F:/cs01/cs01/cj2/docs/android/guide/appshop)(约耗时:5分钟)

当您按步骤完成这些任务的时候.

恭喜您,您已经学会了AS 研发自动化小程序.

# 1.第一个小程序

## 1. apk的安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1-apk的安装" \o "Direct link to 1. apk的安装)

* 您可以 **点击下载** [AS.apk](http://py.airscript.cn/getApk" \t "F:/cs01/cs01/cj2/_blank)文件. 也可通过 **扫描二维码** 进行安装

## 2. 访问编辑器[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2-访问编辑器" \o "Direct link to 2. 访问编辑器)

保持手机 和 电脑在同一个局域网(wifi) 下,浏览器中输入 AS >开发者>局域网IP

## 3. 输出HelloWorld[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "3-输出helloworld" \o "Direct link to 3. 输出HelloWorld)

创建一个工程,在工程中输入 以下代码

! 请注意,使用 英文双引号 ""

print("Hello World")

点击 ▶️ 运行程序

## 4.任务达成[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "4任务达成" \o "Direct link to 4.任务达成)

如果您也输出了 “HelloWorld”

恭喜您,任务达成.

# 2.Python语法

AS 是由 Python3.8驱动的.

在开发之前,您必须掌握一定的Python基础语法.

## 1.Python基础[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1python基础" \o "Direct link to 1.Python基础)

以下是您必须要掌握的Python基础语法

| **Python基础** | **备注** | **文档** | **视频教程** |
| --- | --- | --- | --- |
| 注释 | 基础 | [文档](https://www.runoob.com/python3/python3-comment.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=4" \t "F:/cs01/cs01/cj2/_blank) |
| 变量 | 基础 | [文档](https://www.runoob.com/python/python-variable-types.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=5" \t "F:/cs01/cs01/cj2/_blank) |
| 数据类型 | 基础 | [文档](https://www.runoob.com/python/python-variable-types.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=6" \t "F:/cs01/cs01/cj2/_blank) |
| 运算符 | 基础 | [文档](https://www.runoob.com/python3/python3-basic-operators.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=7" \t "F:/cs01/cs01/cj2/_blank) |
| 条件语句 | 基础 | [文档](https://www.runoob.com/python3/python3-conditional-statements.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=8" \t "F:/cs01/cs01/cj2/_blank) |
| pass语句 | 基础 | [文档](https://www.runoob.com/python/python-pass-statement.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=9" \t "F:/cs01/cs01/cj2/_blank) |
| 循环语句 | 基础 | [文档](https://www.runoob.com/python3/python3-loop.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=10" \t "F:/cs01/cs01/cj2/_blank) |
| 列表 | 基础 | [文档](https://www.runoob.com/python3/python3-list.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=11" \t "F:/cs01/cs01/cj2/_blank) |
| 列表操作符与方法 | 基础 | [文档](https://www.runoob.com/python3/python3-list.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=12" \t "F:/cs01/cs01/cj2/_blank) |
| 元组 | 基础 | [文档](https://www.runoob.com/python3/python3-tuple.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=13" \t "F:/cs01/cs01/cj2/_blank) |
| 字典 | 基础 | [文档](https://www.runoob.com/python3/python3-dictionary.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=14" \t "F:/cs01/cs01/cj2/_blank) |
| 字典的方法 | 基础 | [文档](https://www.runoob.com/python3/python3-dictionary.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=15" \t "F:/cs01/cs01/cj2/_blank) |
| 集合 | 基础 | [文档](https://www.runoob.com/python3/python3-set.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=16" \t "F:/cs01/cs01/cj2/_blank) |
| 集合的交叉集 | 基础 | [文档](https://www.runoob.com/python3/python3-set.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=17" \t "F:/cs01/cs01/cj2/_blank) |
| 序列 | 基础 | [文档](F:/cs01/cs01/cj2/1extracted_div_content.html) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=18" \t "F:/cs01/cs01/cj2/_blank) |
| for迭代器 | 基础 | [文档](https://www.runoob.com/python3/python3-iterator-generator.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=19" \t "F:/cs01/cs01/cj2/_blank) |
| 函数 | 基础 | [文档](https://www.runoob.com/python3/python3-function.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=20" \t "F:/cs01/cs01/cj2/_blank) |
| print | 基础 | [文档](https://www.runoob.com/python3/python3-inputoutput.html" \t "F:/cs01/cs01/cj2/_blank) | [视频](https://www.bilibili.com/video/BV1WX4y1s7Ab?p=21" \t "F:/cs01/cs01/cj2/_blank) |

## 2.任务达成[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2任务达成" \o "Direct link to 2.任务达成)

如果您已经 完成了Python基础语法的学习.

我们给您出道题来检验一下学习成果吧~

题目:   
1.自定义一个函数 mysort  
2.给这个函数 传入一个字典 {"自在": 31, "郭立员": 18, "随便大佬": 24 }  
3.让函数打印出 年龄在 15 - 25 之间的所有人的名字.

如果您能很快的完成 上面的题目.

恭喜您 任务达成!

# 3.自动化函数

## 1.API学习[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1api学习" \o "Direct link to 1.API学习)

我们需要您 对AS API 有一个基础的认识与学习

下列 内容需要您掌握.

| **章节** | **备注** | **文档** | **视频教程** |
| --- | --- | --- | --- |
| 动作指令 | 学会点击,滑动,触摸,手势等操作 | [文档](F:/cs01/cs01/cj2/docs/android/api/action.md) | [视频教程](https://www.bilibili.com/video/BV1HX4y1i7pf/?p=4" \t "F:/cs01/cs01/cj2/_blank) |
| 控件检索 | 通过组合指令能够找到任意控件 | [文档](F:/cs01/cs01/cj2/docs/android/api/node/selector) | [视频教程](https://www.bilibili.com/video/BV1HX4y1i7pf/?p=11" \t "F:/cs01/cs01/cj2/_blank) |
| 图色检索 | 在任意游戏界面能通过图色检索拿到想要的坐标点等信息 | [文档](F:/cs01/cs01/cj2/docs/android/api/screen/) | [视频教程](https://www.bilibili.com/video/BV1HX4y1i7pf/?p=20" \t "F:/cs01/cs01/cj2/_blank) |
| UI交互 | 能随意创建UI窗口,并与Python取得交互 | [文档](F:/cs01/cs01/cj2/docs/android/api/ui/window) | [视频教程](F:/cs01/cs01/cj2/1extracted_div_content.html) |

学会这几个章节,就可以制作出有意思的自动化小程序了~

## 2.任务达成[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2任务达成" \o "Direct link to 2.任务达成)

# 4.发布小程序

当制作完成小程序后,您就可以上架小程序了.

跟着我们的任务引导,您就可以通过任意方式控制 小程序的分发了.

## 1.上传[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1上传" \o "Direct link to 1.上传)

* 1. 在代码编辑器中导出后缀名为as的工程包
  2. 打开 [开发者后台](http://py.airscript.cn/admin/apply/list" \t "F:/cs01/cs01/cj2/_blank),进入小程序列表
  3. 点击新增,填写资料,提交发布
  4. 在Airscript App中 通过id检索

上传后,在小程序列表中 会有对应小程序ID

我们可以在Airscript首页通过这个ID搜索并运行小程序.

## 2.打包独立APP[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2打包独立app" \o "Direct link to 2.打包独立APP)

* 1. 您需要发布一款小程序才可打包
  2. 在 [开发者后台](http://py.airscript.cn/admin/apply/list" \t "F:/cs01/cs01/cj2/_blank) 中找到一款发布的小程序,点击后方打包小程序
  3. 填写打包信息,点击开始打包等待完成即可.
  + 打包需要10分钟左右,可以离开界面

# 5.打包为独立APP

当我们上架 第一个小程序后,即可将小程序打包为独立运行的APK文件进行分发了

## 打包[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "打包" \o "Direct link to 打包)

* 1. 您需要发布一款小程序才可打包
  2. 在 [开发者后台](http://py.airscript.cn/admin/apply/list" \t "F:/cs01/cs01/cj2/_blank) 中找到一款发布的小程序,点击后方打包小程序
  3. 填写打包信息,点击开始打包等待完成即可.
  + 打包需要10分钟左右,可以离开界面

## 分发[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "分发" \o "Direct link to 分发)

下载完成后,我们就可以点击下载,进行分发啦~

用户可以通过扫码安装,也可以下载APK文件进行安装.

安装好,打开APP就只显示自己的小程序啦~

# 6.使用权限

我们可以发售激活码来控制哪些用户,可以运行我们的小程序.

## 1.小程序设置为收费[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1小程序设置为收费" \o "Direct link to 1.小程序设置为收费)

* 1. 先将小程序设置为 需要激活码启动

## 2.生成激活码[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2生成激活码" \o "Direct link to 2.生成激活码)

## 3. 在设备中使用激活码[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "3-在设备中使用激活码" \o "Direct link to 3. 在设备中使用激活码)

# 哪个开发工具适合你呢?

[视频教程 · 开发工具的介绍,安装与使用](https://www.bilibili.com/video/BV1sc411R7yb" \t "F:/cs01/cs01/cj2/_blank)

## 新手入门[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "新手入门" \o "Direct link to 新手入门)

[如果您是一个新手 ,推荐您使用 原生WEB 开发](F:/cs01/cs01/cj2/docs/android/devtools/webide)

优势: 免配置环境 极简开发

劣势: 无代码提示,手动导包

## 高手开发[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "高手开发" \o "Direct link to 高手开发)

[推荐Pycharm 开发](F:/cs01/cs01/cj2/docs/android/devtools/pycharm)

大型工程必备

优势: 自动导包,代码提示,格式化代码 等.

劣势: 需安装工具,配置环境,较为繁琐.

# 入门 · WebIde

免部署模式

采用‘WEB-IDE’作为开发工具.

APP内置服务器,供开发者访问,免去开发者的环境配置.极大降低研发成本.

## 安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装" \o "Direct link to 安装)

[下载](F:/cs01/cs01/cj2/docs/android/download.md)并在您的移动设备上安装**AS App**即可.

## 连接[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "连接" \o "Direct link to 连接)

### 局域网[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "局域网" \o "Direct link to 局域网)

打开APP后,点击“开发者”,即可看到 **开发地址->** 局域网IP(192.168....\*)

* 确认手机和电脑处于同一个局域网(PS:连接同一个Wifi)
* 在您的电脑(windows,mac,linux) 等设备中,访问该ip地址即可研发

### 公网[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "公网" \o "Direct link to 公网)

打开APP后,点击“开发者”,即可看到 “开发地址->公网IP”

* 点击公网IP开关,给予VPN权限
* 在您的电脑(windows,mac,linux) 等设备中,访问公网地址([http://ide.airscript.cn/.\*)即可研发](http://ide.airscript.cn/.*)%E5%8D%B3%E5%8F%AF%E7%A0%94%E5%8F%91" \t "F:/cs01/cs01/cj2/_blank)

## 研发[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "研发" \o "Direct link to 研发)

### 访问首页[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "访问首页" \o "Direct link to 访问首页)

* 开发工具首页,点击 **"开始研发"**

### 创建工程[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建工程" \o "Direct link to 创建工程)

* 创建一个新工程

# 资深 · PyCharm

## 1.安装Python[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1安装python" \o "Direct link to 1.安装Python)

### 下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "下载" \o "Direct link to 下载)

1. 自行到[官网](https://www.python.org/downloads/" \t "F:/cs01/cs01/cj2/_blank)下载合适的版本
2. （推荐）通过[官网](https://www.python.org/ftp/python/3.8.10/python-3.8.10-amd64.exe" \t "F:/cs01/cs01/cj2/_blank)或者通过[百度网盘](https://pan.baidu.com/s/1VVIscWwUVltYCAwdLiMSQw?pwd=yvw8" \t "F:/cs01/cs01/cj2/_blank) 下载和AS设备相同版本解释器。

### 安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装" \o "Direct link to 安装)

等待安装完成后点击Close

然后打开按住键盘的win + r组合键，输入 cmd 再输入 python 验证是否安装成功，如果看到如下界面，代表安装成功。

## 2. 安装PyCharm[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2-安装pycharm" \o "Direct link to 2. 安装PyCharm)

### 下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "下载-1" \o "Direct link to 下载)

#### 下载地址[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "下载地址" \o "Direct link to 下载地址)

版本限制

目前仅支持指定版本(2024.1)Pycharm ,最新版本API不兼容

专业版：[https://download-cdn.jetbrains.com.cn/python/pycharm-professional-2024.1.exe](https://download-cdn.jetbrains.com.cn/python/pycharm-professional-2024.1.exe" \t "F:/cs01/cs01/cj2/_blank)

社区版：[https://download-cdn.jetbrains.com.cn/python/pycharm-community-2024.1.exe](https://download-cdn.jetbrains.com.cn/python/pycharm-community-2024.1.exe" \t "F:/cs01/cs01/cj2/_blank)

* pycharm是Jetbrains公司开发的一款专门用于python开发的IDE，是目前在市场上用于python开发最好的工具。使用pycharm开发python有代码智能提示、语法检测、包管理等强大的功能。
* pycharm有2个版本，分别是收费的Professional版和免费的Community版，如果不是有强烈的python开发需求，仅仅是开发AS的话，社区版足够使用了。建议下载社区版。

### 安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装-1" \o "Direct link to 安装)

等待安装完成后打开pycharm

### 测试环境[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "测试环境" \o "Direct link to 测试环境)

然后可以看到如下界面，点击图中内容创建项目

配置解释器

点击OK后，会回到创建界面，点击Create创建项目

直接点击右上角的运行，看看是否成功

### 汉化[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "汉化" \o "Direct link to 汉化)

等待安装完成后，点击重启pycharm

即可看到中文界面

## 3.AirScript插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "3airscript插件" \o "Direct link to 3.AirScript插件)

使用pycharm开发AS需要用到AS开发插件，该插件在pycharm上提供和web ide一致的开发体验。

### 下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "下载-2" \o "Direct link to 下载)

目前插件只能通过离线的方式安装，点击下面的链接下载最新插件。

[Pycharm AirScript离线版下载](https://pan.baidu.com/s/1UGM6ucM_Wn7X4RiAQm7q9g?pwd=pggv" \t "F:/cs01/cs01/cj2/_blank)

### 安装插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装插件" \o "Direct link to 安装插件)

然后选择刚刚下载插件包的位置，即可安装，安装完成后重启IDE使插件生效。

### 使用[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "使用" \o "Direct link to 使用)

打开手机上的AS找到开发地址

把设备信息配置到插件

点击连接设备

提示设备连接成功即代表成功连接，如连接失败，可到浏览器访问手机AirScript APP提供的地址，看看是否可以成功访问。例如：

浏览器可以打开，代表能成功访问，如果浏览器能打开插件还是提示连接失败，请联系作者。

#### 新建/拉取项目[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "新建拉取项目" \o "Direct link to 新建/拉取项目)

依次输入项目名，选择设备列表。

目录的最后一级为项目名，项目名对应设备上这个界面的项目名

比如我现在新建的项目为AirScriptProject，而我的设备上可以看到已经有一个同名项目为AirScriptProject，那么这次创建就是拉取项目，如果我这次新建的是设备上没有的，那么就会在设备上新建项目。

项目创建成功后重新连接设备，点击旁边的运行即可。

## 4.代码提示的安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "4代码提示的安装" \o "Direct link to 4.代码提示的安装)

### 安装提示包[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装提示包" \o "Direct link to 安装提示包)

使用pip工具可以安装ascript-tip的代码提示包

pip install ascript-tip

### 更新提示包[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "更新提示包" \o "Direct link to 更新提示包)

pip install --upgrade ascript-tip

# VsCode

需安装好 AS App

Vs开发也需要在您的手机设备上安装 Ascript App

否则无法连接调试

VsCode 插件适合开发一些大型的项目工程.

其中对Python,Html,Js 的代码提示比较友好,我们也**'比较推荐'**这种形式的开发

对于图色工具,节点工具,导包等,我们还是需要在网页端完成,VsCode 目前还没有提供这些功能入口.

## 安装AS App[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装as-app" \o "Direct link to 安装AS App)

[下载](F:/cs01/cs01/cj2/docs/android/download.md)并在您的移动设备上安装**AScript App**即可.

## 下载VsCode[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "下载vscode" \o "Direct link to 下载VsCode)

* 下载地址

[https://code.visualstudio.com](https://code.visualstudio.com" \t "F:/cs01/cs01/cj2/_blank)

## 插件的安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件的安装" \o "Direct link to 插件的安装)

* 1. 打开VsCode 中的插件拓展
  2. 搜索 ‘AirScript’
  3. 点击安装 并重启 VsCode

如下图所示:

## 插件的使用[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件的使用" \o "Direct link to 插件的使用)

* 1. 创建一个空工程文件夹,文件夹名称任意,并用VsCode 打开
  2. 在弹出框中,连接手机设备!
  3. 创建新工程,或拉取手机中的项目.
  4. 右键工程文件夹 运行或停止

如下图(2)所示:

# 编辑器拓展接口

开放接口 是 AS APP 启动后 对外开放的接口

通过对外开放的接口,可以实现工程的管理,工程的运行停止.

可以实现第三方编辑器的插件开发

## 接口规范[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "接口规范" \o "Direct link to 接口规范)

### host地址[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host地址" \o "Direct link to host地址)

host地址 为请求开放接口 的 基址.

它是一个http地址,有了它后才可以定位请求哪台机器的开放接口.

#### 如何获得host地址?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "如何获得host地址" \o "Direct link to 如何获得host地址?)

* 1. 首先需要您在手机等移动设备上安装 AS app
  2. host地址获取: AS App主页 - 开发者 - (局域网ip地址,或公网ip地址)
  3. 如上图中:

局域网ip:[http://192.168.31.58:9096](http://192.168.31.58:9096" \t "F:/cs01/cs01/cj2/_blank)

公网ip:[http://ide.airscript.cn/1/2/9096/](http://ide.airscript.cn/1/2/9096/" \t "F:/cs01/cs01/cj2/_blank)

都可作为Host 地址,来请求 AS 开放接口

### 请求方式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "请求方式" \o "Direct link to 请求方式)

Post/Get

开放接口中 Post 或 Get 形式都是支持的.

### 参数要求[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "参数要求" \o "Direct link to 参��数要求)

Get,Post 形式中如出现特殊字符,或中文等情况.

请务必进行urlencode 处理.

### 数据返回[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "数据返回" \o "Direct link to 数据返回)

所有接口的返回数据格式均为json

* 结构如下:

{  
 "code": 1,  
 "msg": "success",  
 "data": []/{}  
}

* 返回值说明

| **返回值** | **必须** | **备注** |
| --- | --- | --- |
| code | 必有 | 接口成功返回1,其他值为失败 |
| msg | 必有 | 接口的处理结果信息 |
| data | 必有 | 每个接口的回执数据,有可能是一个class,或者是一个数组 |

## 工程管理[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "工程管理" \o "Direct link to 工程管理)

工程类的开放接口 ,可以实现工程的管理.

如通过接口,创建,删除,重命名,获取已创建的工程列表等功能.

### 创建[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建" \o "Direct link to 创建)

创建一个新的工程

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/create

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| name | 必填 | 工程的名称 |

* 返回结果

{  
 "code": 1,  
 "msg": "success"  
}

### 删除[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除" \o "Direct link to 删除)

删除一个AS工程

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/remove

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| name | 必填 | 要删除的工程名 |

* 返回结果

{  
 "code": 1,  
 "msg": "success"  
}

### 重命名[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "重命名" \o "Direct link to 重命名)

工程重命名

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/rename

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| name | 必填 | 目标工程名称 |
| rename | 必填 | 要替换的名称 |

* 返回结果

{  
 "code": 1,  
 "msg": "success"  
}

### 工程列表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "工程列表" \o "Direct link to 工程列表)

获取所有已创建的工程

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/getlist

* 参数

无

* 返回结果

{  
 "code": 1,  
 "data": [{  
 "childs": [],  
 "ico": "/storage/emulated/0/airscript/model/888/res/img/logo.png",  
 "isFile": false,  
 "lastModified": 1685325500000,  
 "lastModified\_format": "2023-05-29 09:58:20",  
 "length": 3488,  
 "length\_format": "3.406KB",  
 "name": "888",  
 "path": "/storage/emulated/0/airscript/model/888"  
 }, {  
 "childs": [],  
 "ico": "/storage/emulated/0/airscript/model/测试log/res/img/logo.png",  
 "isFile": false,  
 "lastModified": 1685287188000,  
 "lastModified\_format": "2023-05-28 23:19:48",  
 "length": 3488,  
 "length\_format": "3.406KB",  
 "name": "测试log",  
 "path": "/storage/emulated/0/airscript/model/测试log"  
 }],  
 "msg": "success"  
}

* 返回值说明

| **返回值** | **必须** | **备注** |
| --- | --- | --- |
| ico | 必有 | 工程的logo |
| lastModified | 必有 | 最后一次更新的时间 |
| lastModified\_format | 必有 | 格式化后的最后一次更新时间 |
| length | 必有 | 工程的大小,单位字节 |
| length\_format | 必有 | 格式化后的工程大小 |
| name | 必有 | 工程的名称 |
| path | 必有 | 工程的文件路径 |

## 小程序[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "小程序" \o "Direct link to 小程序)

小程序接口,可以控制本地小程序的启停,打包,获取小程序源码文件列表 等操作

### 启动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "启动" \o "Direct link to 启动)

启动一个本地小程序

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/run

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| name | 必填 | 本地小程序的名称 |

### 停止[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "停止" \o "Direct link to 停止)

停止当前运行的小程序

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/stop

* 参数

无

### 小程序文件信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "小程序文件信息" \o "Direct link to 小程序文件信息)

获取小程序的详细信息,返回文件列表等信息

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/get

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| name | 必填 | 本地小程序名称 |

* 返回结果

{  
 "code": 1,  
 "data": {  
 "childs": [{  
 "childs": [{  
 "childs": [{  
 "childs": [],  
 "isFile": true,  
 "lastModified": 1682498798000,  
 "lastModified\_format": "2023-04-26 16:46:38",  
 "length": 2424,  
 "length\_format": "2.367KB",  
 "name": "xcx.png",  
 "path": "/storage/emulated/0/airscript/model/a/res/img/xcx.png"  
 }],  
 "isFile": false,  
 "lastModified": 1682498798000,  
 "lastModified\_format": "2023-04-26 16:46:38",  
 "length": 3488,  
 "length\_format": "3.406KB",  
 "name": "img",  
 "path": "/storage/emulated/0/airscript/model/a/res/img"  
 }, {  
 "childs": [],  
 "isFile": false,  
 "lastModified": 1679888239000,  
 "lastModified\_format": "2023-03-27 11:37:19",  
 "length": 3488,  
 "length\_format": "3.406KB",  
 "name": "layout",  
 "path": "/storage/emulated/0/airscript/model/a/res/layout"  
 }],  
 "isFile": false,  
 "lastModified": 1679888239000,  
 "lastModified\_format": "2023-03-27 11:37:19",  
 "length": 3488,  
 "length\_format": "3.406KB",  
 "name": "res",  
 "path": "/storage/emulated/0/airscript/model/a/res"  
 }, {  
 "childs": [],  
 "isFile": true,  
 "lastModified": 1685328504000,  
 "lastModified\_format": "2023-05-29 10:48:24",  
 "length": 3,  
 "length\_format": "3.000B",  
 "name": "\_\_init\_\_.py",  
 "path": "/storage/emulated/0/airscript/model/a/\_\_init\_\_.py"  
 }],  
 "isFile": false,  
 "lastModified": 1682500673000,  
 "lastModified\_format": "2023-04-26 17:17:53",  
 "length": 3488,  
 "length\_format": "3.406KB",  
 "name": "a",  
 "path": "/storage/emulated/0/airscript/model/a"  
 },  
 "msg": "success"  
}

* 返回值说明

| **返回值** | **必须** | **备注** |
| --- | --- | --- |
| isFile | 必有 | 是否是文件,false为文件夹,true为文件 |
| lastModified | 必有 | 最后一次更新的时间 |
| lastModified\_format | 必有 | 格式化后的最后一次更新时间 |
| length | 必有 | 工程的大小,单位字节 |
| length\_format | 必有 | 格式化后的工程大小 |
| name | 必有 | 文件的名称 |
| path | 必有 | 文件路径 |
| childs | 必有 | 如果包含子文件 则会出现数组数据,每一条数据就是一个子文件的详细描述 |

### 导出小程序[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导出小程序" \o "Direct link to 导出小程序)

导出小程序包

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/export

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| name | 必填 | 本地小程序的名称 |

* 返回

数据包文件

### 获取已安装的Python包[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取已安装的python包" \o "Direct link to 获取已安装的Python包)

查询已安装的Python包

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/model/pip

* 参数

无

* 返回

{  
 "code": 1,  
 "data": {  
 "install": ["opencv-contrib-python\u003d\u003d4.1.2.30", "requests", "pymysql", "numpy"],  
 "options": ["--timeout", "1000"]  
 },  
 "msg": "success"  
}

* 返回值说明

| **返回值** | **必须** | **备注** |
| --- | --- | --- |
| install | 必有 | 已安装python包的 数组列表 |
| options | 必有 | pip install 的额外参数配置 |

### 日志监听[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "日志监听" \o "Direct link to 日志监听)

AS 内置了

## 文件管理[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "文件管理" \o "Direct link to 文件管理)

文件管理接口,可以删除,获取,重命名,上传某个文件

### 创建[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建-1" \o "Direct link to 创建)

创建文件或文件夹

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/file/create

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| path | 必填 | 文件夹的路径 |
| name | 必填 | 文件的名称 |
| type | 必填 | 文件类型: 创建文件 type=file  创建文件夹type=dir |

### 删除[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除-1" \o "Direct link to 删除)

删除制定文件

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/file/remove

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| path | 必填 | 要删除的文件全路径 |

### 获取文件内容[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取文件内容" \o "Direct link to 获取文件内容)

获取文件内容

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/file/get

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| path | 必填 | 文件的路径 |

* 返回结果

文件内容

### 存储文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "存储文件" \o "Direct link to 存储文件)

存储内容到制定文件

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/file/save

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| path | 必填 | 文件的路径 |
| content | 必填 | 文件的内容 |

### 重命名[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "重命名-1" \o "Direct link to 重命名)

指定文件重命名

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/file/rename

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| path | 必填 | 要重命名的文件全路径 |
| name | 必填 | 重命名名称 |

### 上传文件或小程序[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "上传文件或小程序" \o "Direct link to 上传文件或小程序)

上传文件,或小程序

* 接口地址

[host](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "host%E5%9C%B0%E5%9D%80)/api/file/rename

* 参数

| **参数** | **必须** | **备注** |
| --- | --- | --- |
| body data | 必填 | 需要将文件信息传入body 中的 data字段中 |
| path | 必填 | 上传的文件路径 |
| zip | 选填 | zip= 1 传入此参数,系统默认上传的是一个as小程序包,会自动解压至小程序目录中 |

# Python语言基础语法

AS 内置Python 3.8 解释器

因此你需要掌握一些Python3.8的基础语法.

您可以通过 **以下两种方式学习**.

## 视频教程[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "视频教程" \o "Direct link to 视频教程)

[Python基础语法官方视频教程](https://www.bilibili.com/video/BV1WX4y1s7Ab" \t "F:/cs01/cs01/cj2/_blank)

## 相关语法文档[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "相关语法文档" \o "Direct link to 相关语法文档)

[菜鸟文档-Python](https://www.runoob.com/python/python-basic-syntax.html" \t "F:/cs01/cs01/cj2/_blank)

# Ascript App内置库

AScript 内置了三部分库

**这三部分库,开发时可直接使用.**

* [Python 标准函数库](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "python-%E6%A0%87%E5%87%86%E5%87%BD%E6%95%B0%E5%BA%93)
* [拓展库](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E6%8B%93%E5%B1%95%E5%BA%93)
* [安卓API](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E5%AE%89%E5%8D%93api)

## Python 标准函数库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "python-标准函数库" \o "Direct link to Python 标准函数库)

由于Android 平台的迁移,大部分Python标准库是支持的.

我们都可以通过 import 语句进行导入

### 标准库文档[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "标准库文档" \o "Direct link to 标准库文档)

[Python 标准库 API 文档](https://docs.python.org/zh-cn/3.8/library/" \t "F:/cs01/cs01/cj2/_blank)

### AScript 不支持的标准库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "ascript-不支持的标准库" \o "Direct link to AScript 不支持的标准库)

以下模块不受支持，因为它们需要Android上没有的操作系统功能：

- crypt  
- grp  
- nis  
- spwd

以下模块不受支持，因为它们需要我们目前不包括的库：  
  
- curses  
- dbm.gnu  
- dbm.ndbm  
- readline  
- tkinter  
- turtle

## 拓展库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "拓展库" \o "Direct link to 拓展库)

以下第三方常用库,在ASsript App中已内置,你可以通过 import 直接使用.

* "opencv-contrib-python==4.1.2.30",
* "requests",
* "pymysql",
* "numpy",
* "websocket-client",
* "pillow",
* "pandas",
* "openpyxl"

*#例如 opencv numpy 可导入使用.*  
import cv2  
import numpy as np

## 安卓API[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安卓api" \o "Direct link to 安卓API)

android库是由java编写

在AScript中,可通过 [import 导入java类,来使用android api](F:/cs01/cs01/cj2/docs/android/api/python/pyandroid)

## 导入 import[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导入-import" \o "Direct link to 导入 import)

import 可以导入哪些库

标准库

工程目录下的库 (工程目录下的py文件)

拓展库 (通过pip install 安装的库)

java或android库( AS 把java和android api 封装成了内置python库)

下面是一些案例

* 导入本工程下的python文件

这是一个as工程结构,我们来看不同层级,是如何导入文件的

.demo "工程名称"  
├─ \_\_init\_\_.py "小程序启动入口"  
├─ main.py  
├─ app   
│ └─ home   
│ └─ banner.py   
└─ vip   
 └─ login.py

*# 在\_\_init\_\_.py 中导入同级的 main.py*  
from . import main

*# 在\_\_init\_\_.py 中导入vip/login.py*  
from .vip import login

*# 在\_\_init\_\_.py 中导入app/home/banner.py*  
from .app.home import banner

*# 在banner.py 中导入login.py*  
from ascript.android.system import R  
\_\_import\_\_(R.name+".vip.login")

# pip安装Python拓展库

如果AS app中内置的python包无法满足你的开发需求.

开发者可通过pip配置,将需要的第三方python包编译进 AS app中.

Python包拓展需要重新打包 AS

为何不能动态拓展?

Android 7.0开始对动态引入 动态链接库做了限制,因此我们只能重新打包把需要的动态链接库内置进 AS App中

## 安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安装" \o "Direct link to 安装)

安装Python库,需要前往开发者后台操作

1. 前往[开发者后台](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "https://py.airscript.cn/admin/pip/list)
2. 点击共享资源/Pip拓展/新增
3. 加入自己想添加的库,提交等待编译
4. 下载编译好的AScript.apk文件,安装至手机
5. 代码中通过 import 导入即可使用.

拓展库注意事项!

需要指定版本时,可以直接写例如: “numpy==1.1.2”

当需要增加 pip options 参数,如需指定自己的仓库

key设置为:"--extra-index-url" value设置为:"https://example.com/private/repository"

## 开发者已编译好的拓展库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "开发者已编译好的拓展库" \o "Direct link to 开发者已编译好的拓展库)

[共享Python拓展库](F:/cs01/cs01/cj2/docs/android/share/pip)

# Python调用Java代码

AS 的强大在于可以与AndroidSDK 混编

在AS中 可以像引入Python包那样,引入Java Class

### 导入java 类[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导入java-类" \o "Direct link to 导入java 类)

* 利用jclass 导入 java类

from java import jclass  
*# 通过 java Claendar获取当前时间的毫秒值*  
Calendar = jclass("java.util.Calendar")  
c = Calendar.getInstance()  
t = c.getTimeInMillis()  
print(t)

* 例如: 在Python中使用 java 的 Map

from java.util import HashMap  
  
map = HashMap()  
map.put("1","自在")  
map.put("2",True)  
  
print(map.get("1"))

### 实现 java 接口[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "实现-java-接口" \o "Direct link to 实现 java 接口)

* 例如:用动态代理,来使用java 多线程Thread

from java.lang import Runnable, Thread  
from java import dynamic\_proxy  
class R(dynamic\_proxy(Runnable)):  
 def \_\_init\_\_(self, name):  
 super().\_\_init\_\_()  
 self.name = name  
 def run(self):  
 print("Running " + self.name)  
  
r = R("hello")  
t = Thread(r)  
t.start()

# 工程结构

## 结构[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "结构" \o "Direct link to 结构)

一个新的 AS 工程目录结构

.  
├─ \_\_init\_\_.py "小程序启动入口"  
├─ res "资源目录"  
│ ├─ img "图片资源"  
│ │ └─ logo.png "默认的logo"  
│ └─ ui "布局文件资源"  
│  
└─ other "开发者自由创建"  
 └─ ?.py

### 更改logo[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "更改logo" \o "Direct link to 更改logo)

替换 /res/img/logo.png 即可.

### 资源路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "资源路径" \o "Direct link to 资源路径)

系统中提供了访问工程的上下文 [R](F:/cs01/cs01/cj2/docs/android/api/system/" \l "%E7%8E%AF%E5%A2%83%E4%B8%8E%E8%B5%84%E6%BA%90)

# 模式

不同的模式可以共用一套代码,**系统会自动用不同模式下的方法去实现相应功能**

目前 **无障碍模式功能最全,且最稳定.**

* [无障碍模式](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E6%97%A0%E9%9A%9C%E7%A2%8D%E6%A8%A1%E5%BC%8F) , [图色模式/原HID模式](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E5%9B%BE%E8%89%B2%E6%A8%A1%E5%BC%8F) , [Root/激活模式](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "root%E6%88%96%E6%BF%80%E6%B4%BB%E6%A8%A1%E5%BC%8F)

## 3种模式的区别[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "3种模式的区别" \o "Direct link to 3种模式的区别)

| **功能** | **无障碍** | **图色模式(原HID模式)** | **Root或激活模式** |
| --- | --- | --- | --- |
| 动作 | ✅ | 虚拟HID或HID硬件 | ✅ |
| 控件检索 | ✅ | ❌ | ✅ |
| 图色检索: 找色,找图,文字识别,目标 | ✅ | ✅ | ✅ |
| UI | ✅ | ✅ 透传模式部分机型不支持 | ✅ 透传模式部分机型不支持 |
| 其他功能 | ✅ | ✅ | ✅ |
| 被检测概率 | 中 | 小 | 大 |
| 适用场景 | APP类 | 游戏类/羊毛类 | APP类/游戏类 |
| 独立运行 | ✅ | ✅ | ✅ |

## 无障碍模式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "无障碍模式" \o "Direct link to 无障碍模式)

借助Android 的无障碍访问权限,来实现手机的自动化程序.

该模式较为稳定,且拥有最稳定,最兼容的权限.

该模式下,只需在手机上安装 AirScript APP 即可编写运行程序.

安装APP后选择该 “无障碍模式运行” 即可

## 图色模式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图色模式" \o "Direct link to 图色模式)

图色模式下,不用开无障碍权限.

可通过 虚拟HID,HID物理硬件 执行点击,滑动,模拟按键等功能.

### 物理HID硬件魔鼠[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "物理hid硬件魔鼠" \o "Direct link to 物理HID硬件魔鼠)

使用物理HID硬件魔鼠,可 **不用开启,开发者权限,无障碍权限** 下执行动作.

### 虚拟HID[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "虚拟hid" \o "Direct link to 虚拟HID)

用电脑模拟虚拟HID设备,来达到执行动作的目的

* 我们可以按照以下方式,来完成HID的配置

#### 1.驱动的安装[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1驱动的安装" \o "Direct link to 1.驱动的安装)

* mac 驱动的安装

brew install libusb

* windows驱动的安装

先行下载[usb驱动](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/libusbK-inf-wizard.exe" \t "F:/cs01/cs01/cj2/_blank),并双击打开

安装步骤:

#### 2.软件安装与启动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2软件安装与启动" \o "Direct link to 2.软件安装与启动)

根据您的电脑操作系统 下载HID设备管理器软件:

| **平台** |
| --- |
| [win-x64](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/AirScript-win-x64-vpn.exe.zip" \t "F:/cs01/cs01/cj2/_blank) |
| [macos-x64](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/AirScript-mac-x64-vpn.tar.gz" \t "F:/cs01/cs01/cj2/_blank) |
| [macos-arm64](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/AirScript-mac-arm-vpn.tar.gz" \t "F:/cs01/cs01/cj2/_blank) |
| [linux-x64](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/ASHID-linux-x64" \t "F:/cs01/cs01/cj2/_blank) |

#### 3.HID的激活与初始化[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "3hid的激活与初始化" \o "Direct link to 3.HID的激活与初始化)

::: danger 激活前注意事项

* 必须关闭开发者模式,否则可能会出现error -3 的错误.

:::

#### 4.设备的绑定[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "4设备的绑定" \o "Direct link to 4.设备的绑定)

::: danger 绑定前注意事项

* 请保持设备和电脑 处于同一个局域网,否则不可用.

:::

* 启动APP,选择HID模式
* 绑定IP地址 与 设备ID

## Root或激活模式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "root或激活模式" \o "Direct link to Root或激活模式)

### Root模式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "root模式" \o "Direct link to Root模式)

如果您的设备已经具有Root权限.

则拥有最高权限,这种模式不用开启无障碍,对手机无系统版本要求.

适合于模拟器使用.

安装APP后选择该 “Root模式运行” 即可

### 激活模式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "激活模式" \o "Direct link to 激活模式)

所谓激活模式是指: 给未Root的设备临时升权.

在激活模式下,可同样执行 adb shell指令, 且不用连接usb数据线.

激活步骤

* 1. 需要安装第三方应用 [Shizuku](https://apt.izzysoft.de/fdroid/repo/moe.shizuku.privileged.api_1049.apk" \t "F:/cs01/cs01/cj2/_blank)
  2. 通过adb 或无限调试激活shizuku
  3. 打开AScript程序,选择Root或激活模式

手机重启后失效.

# 基本操作

from ascript.android import action

模拟 **手指** 操作

## 点击[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "点击" \o "Direct link to 点击)

模拟 **手指点击屏幕**

* 函数

action.click(x:int,y:int,dur:int = 20):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 要点击的屏幕X坐标 |
| y | int | 是 | 要点击的屏幕Y坐标 |
| dur | float | 否 | 点击持续时间,默认20毫秒 |

* 示例

*# 模拟手指点击屏幕坐标 (898,1916)*  
from ascript.android import action  
action.click(898,1916)

*# 模拟手指长按屏幕坐标 (898,1916) 按住两秒不放*  
from ascript.android import action  
action.click(898,1916,2000)

## 滑动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "滑动" \o "Direct link to 滑动)

模拟 **手指滑动屏幕**

* 函数

action.slide(x:int,y:int,x1:int,y1:int,dur:int=20)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 滑动起始点X坐标 |
| y | int | 是 | 滑动起始点Y坐标 |
| x1 | int | 是 | 滑动结束点X坐标 |
| y1 | int | 是 | 滑动结束点Y坐标 |
| dur | float | 否 | 点击持续时间,默认20毫秒 |

* 示例

*# 模拟手指滑动屏幕 从坐标(157,798) 滑动至坐标(946,834) ,耗时20毫秒*  
from ascript.android import action  
action.slide(157,798,946,834)

*# 模拟手指滑动屏幕 从坐标(157,798) 滑动至坐标(946,834) ,耗时3000毫秒*  
from ascript.android import action  
action.slide(157,798,946,834,3000)

## 输入[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "输入" \o "Direct link to 输入)

模拟 **输入文本**

* 函数

action.input(msg:str="",selector:Selector=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| msg | str | 是 | 输入的内容 |
| selector | Selector | 否 | 控件检索规则 当Selector 为None时,您的输入框必须已经获取了焦点 |

* 示例

*# 模拟在当前 ‘已获取焦点’的的文本框中输入‘你好 aslib!’*  
from ascript.android import action  
action.input("你好 aslib!")

*# 模拟在 控件检查器查到的 文本框中输入‘你好 aslib!’*  
from ascript.android import action  
from ascript.android.node import Selector  
  
selector = Selector().id("com.aojoy.airscript:id/search\_bar\_text")  
action.input("你好 aslib!",selector)

## 触摸[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "触摸" \o "Direct link to 触摸)

模拟 **手指 按下,移动,抬起**

*#class*  
action.Touch

### 按下[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "按下" \o "Direct link to 按下)

模拟 **手指按下**

action.Touch.down(x,y,dur:int=20)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 触摸X点坐标 |
| y | int | 是 | 触摸Y点坐标 |
| dur | float | 否 | 动作持续时间,默认20毫秒 |

### 移动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "移动" \o "Direct link to 移动)

模拟 **手指移动**

action.Touch.move(x, y, dur: int = 20)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 触摸X点坐标 |
| y | int | 是 | 触摸Y点坐标 |
| dur | float | 否 | 动作持续时间,默认20毫秒 |

### 抬起[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "抬起" \o "Direct link to 抬起)

模拟 **手指抬起**

action.Touch.up(x, y, dur: int = 20)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 触摸X点坐标 |
| y | int | 是 | 触摸Y点坐标 |
| dur | float | 否 | 动作持续时间,默认20毫秒 |

*# 一个完整的 触摸 操作*  
from ascript.android import action  
  
*# 从坐标(100,200)的位置按下*  
action.Touch.down(299,629)  
*# 按下的同时,移动到坐标(300,300)*  
action.Touch.move(780,645)  
*# 按下的同时,再次移动到坐标(400,350),移动用时 1000毫秒*  
action.Touch.move(445,1152,1000)  
*# 移动至坐标(450,400)位置 抬起*  
action.Touch.up(445,1152)

## 按键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "按键" \o "Direct link to 按键)

模拟 **模拟按键操作**

*#class*  
action.Key

### Home键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "home键" \o "Direct link to Home键)

模拟设备 **Home按键**

action.Key.home()

### 返回键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "返回键" \o "Direct link to 返回键)

模拟设备 **返回按键**

action.Key.back()

#### 弹出通知栏[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "弹出通知栏" \o "Direct link to 弹出通知栏)

模拟设备 **调出通知栏**

action.Key.notifactions()

### 任务栏键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "任务栏键" \o "Direct link to 任务栏键)

模拟设备 **任务栏键**

action.Key.recents()

### 截屏键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "截屏键" \o "Direct link to 截屏键)

模拟设备 **关闭屏幕**

action.Key.screenshot()

### 锁屏键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "锁屏键" \o "Direct link to 锁屏键)

模拟设备 **关闭屏幕**

action.Key.lockscreen()

## 手指路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "手指路径" \o "Direct link to 手指路径)

模拟手指 **移动路径**

*#class*  
from ascript.android.action import Path

### 创建路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建路径" \o "Direct link to 创建路径)

创建一根手指路径, 多根手指创建多个即可.

Path(self,start\_time:int=0, duration:int=20, will\_continue:bool=False)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| start\_time | int | 否 | 延迟执行时间 |
| duration | int | 否 | 路径执行时常 |
| will\_continue | bool | 否 | 执行完毕后,是否抬起手指 True:抬起 False:不抬起,下一个手势将连续执行 |

from ascript.android.action import Path  
  
line1 = Path(0,1000);  
*# 移动初始点*  
line1.moveTo(804,708)   
*# 画直线到点*  
line1.lineTo(827,1479)  
*# 使用二次贝塞尔曲线 从点(500,800) 到 (250,900)*  
line1.quadTo(500,800,250,900)

### 路径方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "路径方法" \o "Direct link to 路径方法)

Path 实现了 [android.graphics.Path](https://developer.android.google.cn/reference/android/graphics/Path?hl=en" \t "F:/cs01/cs01/cj2/_blank) 中的大部分方法.

具体参数参考:[android.graphics.Path](https://developer.android.google.cn/reference/android/graphics/Path?hl=en" \t "F:/cs01/cs01/cj2/_blank)

| **支持的绘制方法如下** |  |
| --- | --- |
| moveTo | rCubicTo |
| lineTo | rMoveTo |
| quadTo | reset |
| rCubicTo | rewind |
| rQuadTo | addArc |
| addCircle | addOval |
| addRect | addRoundRect |
| arcTo | cubicTo |
| setLastPoint | rLineTo |

### 多指操作[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "多指操作" \o "Direct link to 多指操作)

* 函数

action.gesture(paths:List[Path],listener=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| paths | [Path](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E6%89%8B%E6%8C%87%E8%B7%AF%E5%BE%84)[] | 是 | 要执行的一组手指路径 |
| listener | class | 否 | 手势回调对象, 函数需包含两个函数 手势执行完毕回调:onCompleted(), 手势中断回调:onCancelled() |

*#案例1:执行一组动作,分别模拟两个手指同时操作*  
from ascript.android import action  
from ascript.android.action import Path  
  
*# 模拟第一根手指*  
line1 = Path(0,1000);  
line1.moveTo(804,708)   
line1.lineTo(827,1479)  
  
*# 模拟第二根手指*  
line2 = Path(0,1200);  
line2.moveTo(500,200)   
line2.lineTo(500,500)  
  
action.gesture([line1,line2])

*#案例2,执行一组动作,分别模拟两个手指同时操作,并监听动作是否执行结束*  
from ascript.android import action  
from ascript.android.action import Path  
  
line1 = Path(0,1000)  
line1.moveTo(200,200)   
line1.lineTo(200,500)  
  
*# 创建另一个新的path路径 模拟 手指2(\*第二条路径相对于第一条,会延迟500ms启动)*  
line2 = Path(500,1200)  
line2.moveTo(500,200)   
line2.lineTo(500,500)  
  
class listener:  
 def onCompleted():  
 print("结束了")  
 def onCancelled():  
 print("被中断取消了")  
  
*# 执行 这一组动作*  
action.gesture([line1,line2],listener)

*#案例3:连续 执行两个Path路径,中途不松手指*  
from ascript.android import action  
from ascript.android.action import Path  
  
line1 = Path(0,1000,True);  
line1.moveTo(200,200)   
line1.lineTo(200,500)  
  
*# 创建另一个新的path路径 模拟 手指2*  
line2 = Path(1000,500);  
line2.moveTo(200,500)   
line2.lineTo(500,500)  
  
class listener:  
 def onCompleted():  
 gesture.perform(line2)  
 def onCancelled():  
 print("被中断取消了")  
  
action.gesture([line1],listener)

## 捕获点击[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "捕获点击" \o "Direct link to 捕获点击)

捕获用户点击屏幕的坐标

* 函数

action.catch\_click(msg: str = None, shine: bool = True):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| msg | str | 否 | 捕获屏幕坐标时,要显示的文字.默认无文字显示 |
| shine | bool | 否 | 是否闪烁屏幕提示用户. 默认闪烁,False为不闪烁 |

* 返回值

返回用户点击的屏幕坐标位置.

Point 对象. 包含 x,y 属性.分别代表横竖坐标值

* 示例

*# 捕获用户点击屏幕的坐标*  
from ascript.android import action  
p = action.catch\_click()  
print(p)

*# 捕获用户点击屏幕的坐标,给出提示文字,并取消屏幕闪烁*  
from ascript.android import action  
p = action.catch\_click("请点击屏幕任意位置",False)  
print(p)

# 虚拟外设

请先切换 [HID模式](F:/cs01/cs01/cj2/docs/android/api/mode" \l "hid%E6%A8%A1%E5%BC%8F),并安装驱动与控制器

模拟 **外设触控** , 类似于在Android设备上插一个 鼠标 ,从而进行 模拟操作 .

info

在自动化测试中. 当前应用设置,**无障碍点击无效时**. 可以使用此函数**模拟鼠标点击**.

* 优势:点击,滑动,输入等不可被屏蔽
* 劣势:需要Android设备与PC电脑 Usb线连接,设置为充电模式,并下载Hid客户端,在app中绑定Id

*#class*  
action.Hid

## 外设点击[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "外设点击" \o "Direct link to 外设点击)

模拟鼠标 点击

action.Hid.click(x:int,y:int,dur:int=20)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 点击的屏幕X坐标 |
| y | int | 是 | 点击的屏幕Y坐标 |
| dur | int | 否 | 点击时常 默认20毫秒 |

## 外设滑动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "外设滑动" \o "Direct link to 外设滑动)

模拟鼠标 拖拽滑动

action.Hid.slide(x:int,y:int,x1:int,y1:int,dur:int=20):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 滑动的起始点X坐标 |
| y | int | 是 | 滑动的起始点Y坐标 |
| x1 | int | 是 | 滑动的结束点X坐标 |
| y1 | int | 是 | 滑动的结束点Y坐标 |
| dur | int | 否 | 滑动时长 默认20毫秒 |

## 外设按键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "外设按键" \o "Direct link to 外设按键)

模拟键盘按键

action.Hid.key(\*\*keycode):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| keycode | str 可变参数 | 是 | 按键值 |

### key值[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "key值" \o "Direct link to key值)

**类型**:

字母键:

* + - a - A键
    - b - B键
    - c - C键 ...
    - z - Z键

数字键:

* + - 1 - 数字1键
    - 2 - 数字2键 ...
    - 0 - 数字0键

功能键:

* + - enter - 回车键
    - esc - ESC键
    - backspace - 退格键
    - tab - Tab键
    - space - 空格键
    - capsLock - 大写锁定键
    - f1 - F1功能键 ...
    - f12 - F12功能键
    - printScreen - 打印屏幕键
    - scrollLock - 滚动锁定键
    - pause - 暂停键

导航键:

* + - insert - 插入键
    - home - Home键
    - pageUp - 上翻页键
    - delete - 删除键
    - end - End键
    - pageDown - 下翻页键
    - rightArrow - 右箭头键
    - leftArrow - 左箭头键
    - downArrow - 下箭头键
    - upArrow - 上箭头键

符号键:

* + - minus - 减号键
    - equal - 等号键
    - leftBracket - 左方括号键
    - rightBracket - 右方括号键
    - backslash - 反斜杠键
    - semicolon - 分号键
    - quote - 单引号键
    - grave - 重音符键
    - comma - 逗号键
    - period - 句号键
    - slash - 斜杠键

小键盘:

numLock - 小键盘锁定键

keypadDivide - 小键盘除号键

keypadMultiply - 小键盘乘号键

keypadSubtract - 小键盘减号键

keypadAdd - 小键盘加号键

keypadEnter - 小键盘回车键

keypad1 - 小键盘1键 ...

keypad9 - 小键盘9键

keypad0 - 小键盘0键

keypadDecimal - 小键盘小数点键

**描述**: 要发送的按键值。

Shift:

* + - **类型**: Boolean | undefined
    - **描述**: Shift键的状态，表示是否按下。不传默认为 false。

Ctrl:

* + - **类型**: Boolean | undefined
    - **描述**: Ctrl键的状态，表示是否按下。不传默认为 false。

Alt:

* + - **类型**: Boolean | undefined
    - **描述**: Alt键的状态，表示是否按下。不传默认为 false。

Command:

* + - **类型**: Boolean | undefined
    - **描述**: Command键的状态，表示是否按下。不传默认为 false。

id:

* + - **类型**: String
    - **描述**: 要操作的设备的唯一标识符。

# AScript输入法

AS 内置 输入法模块.

主要解决问题场景有. 部分控件不支持输入文字. 部分输入框没有确认按钮,需要发送回车指令等.

使用输入法API前,请手动切换系统输入法为 **AScript输入法**

from ascript.android.action import Ime

## 输入文字[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "输入文字" \o "Direct link to 输入文字)

让输入法在焦点输入框内,输入文字

Ime.input(info: str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| info | str | 是 | 要输入的信息 |

* 示例

from ascript.android.action import Ime  
Ime.input("自在老师")

## 清空文字[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "清空文字" \o "Direct link to 清空文字)

让输入法清空在焦点内的文字

Ime.input\_clear()

* 示例

from ascript.android.action import Ime  
Ime.input\_clear()

## 发送按键指令[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发送按键指令" \o "Direct link to 发送按键指令)

发送按键指令, 如回车,换行,删除等等

Ime.keys(keys: Union[int, list], duration: int = 20)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| keys | int | list[int] | 是 |

* 示例

*# 模拟回车按键*  
from ascript.android.action import Ime  
Ime.keys([66])

### 按键码[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "按键码" \o "Direct link to 按键码)

以下码值为Android代码,**我们只需把变量等号后的数字填入**Ime.keys(..)**即可**.

public static final int FLAG\_WOKE\_HERE = 1;  
 public static final int KEYCODE\_0 = 7;  
 public static final int KEYCODE\_1 = 8;  
 public static final int KEYCODE\_11 = 227;  
 public static final int KEYCODE\_12 = 228;  
 public static final int KEYCODE\_2 = 9;  
 public static final int KEYCODE\_3 = 10;  
 public static final int KEYCODE\_3D\_MODE = 206;  
 public static final int KEYCODE\_4 = 11;  
 public static final int KEYCODE\_5 = 12;  
 public static final int KEYCODE\_6 = 13;  
 public static final int KEYCODE\_7 = 14;  
 public static final int KEYCODE\_8 = 15;  
 public static final int KEYCODE\_9 = 16;  
 public static final int KEYCODE\_A = 29;  
 public static final int KEYCODE\_ALL\_APPS = 284;  
 public static final int KEYCODE\_ALT\_LEFT = 57;  
 public static final int KEYCODE\_ALT\_RIGHT = 58;  
 public static final int KEYCODE\_APOSTROPHE = 75;  
 public static final int KEYCODE\_APP\_SWITCH = 187;  
 public static final int KEYCODE\_ASSIST = 219;  
 public static final int KEYCODE\_AT = 77;  
 public static final int KEYCODE\_AVR\_INPUT = 182;  
 public static final int KEYCODE\_AVR\_POWER = 181;  
 public static final int KEYCODE\_B = 30;  
 public static final int KEYCODE\_BACK = 4;  
 public static final int KEYCODE\_BACKSLASH = 73;  
 public static final int KEYCODE\_BOOKMARK = 174;  
 public static final int KEYCODE\_BREAK = 121;  
 public static final int KEYCODE\_BRIGHTNESS\_DOWN = 220;  
 public static final int KEYCODE\_BRIGHTNESS\_UP = 221;  
 public static final int KEYCODE\_BUTTON\_1 = 188;  
 public static final int KEYCODE\_BUTTON\_10 = 197;  
 public static final int KEYCODE\_BUTTON\_11 = 198;  
 public static final int KEYCODE\_BUTTON\_12 = 199;  
 public static final int KEYCODE\_BUTTON\_13 = 200;  
 public static final int KEYCODE\_BUTTON\_14 = 201;  
 public static final int KEYCODE\_BUTTON\_15 = 202;  
 public static final int KEYCODE\_BUTTON\_16 = 203;  
 public static final int KEYCODE\_BUTTON\_2 = 189;  
 public static final int KEYCODE\_BUTTON\_3 = 190;  
 public static final int KEYCODE\_BUTTON\_4 = 191;  
 public static final int KEYCODE\_BUTTON\_5 = 192;  
 public static final int KEYCODE\_BUTTON\_6 = 193;  
 public static final int KEYCODE\_BUTTON\_7 = 194;  
 public static final int KEYCODE\_BUTTON\_8 = 195;  
 public static final int KEYCODE\_BUTTON\_9 = 196;  
 public static final int KEYCODE\_BUTTON\_A = 96;  
 public static final int KEYCODE\_BUTTON\_B = 97;  
 public static final int KEYCODE\_BUTTON\_C = 98;  
 public static final int KEYCODE\_BUTTON\_L1 = 102;  
 public static final int KEYCODE\_BUTTON\_L2 = 104;  
 public static final int KEYCODE\_BUTTON\_MODE = 110;  
 public static final int KEYCODE\_BUTTON\_R1 = 103;  
 public static final int KEYCODE\_BUTTON\_R2 = 105;  
 public static final int KEYCODE\_BUTTON\_SELECT = 109;  
 public static final int KEYCODE\_BUTTON\_START = 108;  
 public static final int KEYCODE\_BUTTON\_THUMBL = 106;  
 public static final int KEYCODE\_BUTTON\_THUMBR = 107;  
 public static final int KEYCODE\_BUTTON\_X = 99;  
 public static final int KEYCODE\_BUTTON\_Y = 100;  
 public static final int KEYCODE\_BUTTON\_Z = 101;  
 public static final int KEYCODE\_C = 31;  
 public static final int KEYCODE\_CALCULATOR = 210;  
 public static final int KEYCODE\_CALENDAR = 208;  
 public static final int KEYCODE\_CALL = 5;  
 public static final int KEYCODE\_CAMERA = 27;  
 public static final int KEYCODE\_CAPS\_LOCK = 115;  
 public static final int KEYCODE\_CAPTIONS = 175;  
 public static final int KEYCODE\_CHANNEL\_DOWN = 167;  
 public static final int KEYCODE\_CHANNEL\_UP = 166;  
 public static final int KEYCODE\_CLEAR = 28;  
 public static final int KEYCODE\_COMMA = 55;  
 public static final int KEYCODE\_CONTACTS = 207;  
 public static final int KEYCODE\_COPY = 278;  
 public static final int KEYCODE\_CTRL\_LEFT = 113;  
 public static final int KEYCODE\_CTRL\_RIGHT = 114;  
 public static final int KEYCODE\_CUT = 277;  
 public static final int KEYCODE\_D = 32;  
 public static final int KEYCODE\_DEL = 67;  
 public static final int KEYCODE\_DPAD\_CENTER = 23;  
 public static final int KEYCODE\_DPAD\_DOWN = 20;  
 public static final int KEYCODE\_DPAD\_DOWN\_LEFT = 269;  
 public static final int KEYCODE\_DPAD\_DOWN\_RIGHT = 271;  
 public static final int KEYCODE\_DPAD\_LEFT = 21;  
 public static final int KEYCODE\_DPAD\_RIGHT = 22;  
 public static final int KEYCODE\_DPAD\_UP = 19;  
 public static final int KEYCODE\_DPAD\_UP\_LEFT = 268;  
 public static final int KEYCODE\_DPAD\_UP\_RIGHT = 270;  
 public static final int KEYCODE\_DVR = 173;  
 public static final int KEYCODE\_E = 33;  
 public static final int KEYCODE\_EISU = 212;  
 public static final int KEYCODE\_ENDCALL = 6;  
 public static final int KEYCODE\_ENTER = 66;  
 public static final int KEYCODE\_ENVELOPE = 65;  
 public static final int KEYCODE\_EQUALS = 70;  
 public static final int KEYCODE\_ESCAPE = 111;  
 public static final int KEYCODE\_EXPLORER = 64;  
 public static final int KEYCODE\_F = 34;  
 public static final int KEYCODE\_F1 = 131;  
 public static final int KEYCODE\_F10 = 140;  
 public static final int KEYCODE\_F11 = 141;  
 public static final int KEYCODE\_F12 = 142;  
 public static final int KEYCODE\_F2 = 132;  
 public static final int KEYCODE\_F3 = 133;  
 public static final int KEYCODE\_F4 = 134;  
 public static final int KEYCODE\_F5 = 135;  
 public static final int KEYCODE\_F6 = 136;  
 public static final int KEYCODE\_F7 = 137;  
 public static final int KEYCODE\_F8 = 138;  
 public static final int KEYCODE\_F9 = 139;  
 public static final int KEYCODE\_FOCUS = 80;  
 public static final int KEYCODE\_FORWARD = 125;  
 public static final int KEYCODE\_FORWARD\_DEL = 112;  
 public static final int KEYCODE\_FUNCTION = 119;  
 public static final int KEYCODE\_G = 35;  
 public static final int KEYCODE\_GRAVE = 68;  
 public static final int KEYCODE\_GUIDE = 172;  
 public static final int KEYCODE\_H = 36;  
 public static final int KEYCODE\_HEADSETHOOK = 79;  
 public static final int KEYCODE\_HELP = 259;  
 public static final int KEYCODE\_HENKAN = 214;  
 public static final int KEYCODE\_HOME = 3;  
 public static final int KEYCODE\_I = 37;  
 public static final int KEYCODE\_INFO = 165;  
 public static final int KEYCODE\_INSERT = 124;  
 public static final int KEYCODE\_J = 38;  
 public static final int KEYCODE\_K = 39;  
 public static final int KEYCODE\_KANA = 218;  
 public static final int KEYCODE\_KATAKANA\_HIRAGANA = 215;  
 public static final int KEYCODE\_L = 40;  
 public static final int KEYCODE\_LANGUAGE\_SWITCH = 204;  
 public static final int KEYCODE\_LAST\_CHANNEL = 229;  
 public static final int KEYCODE\_LEFT\_BRACKET = 71;  
 public static final int KEYCODE\_M = 41;  
 public static final int KEYCODE\_MANNER\_MODE = 205;  
 public static final int KEYCODE\_MEDIA\_AUDIO\_TRACK = 222;  
 public static final int KEYCODE\_MEDIA\_CLOSE = 128;  
 public static final int KEYCODE\_MEDIA\_EJECT = 129;  
 public static final int KEYCODE\_MEDIA\_FAST\_FORWARD = 90;  
 public static final int KEYCODE\_MEDIA\_NEXT = 87;  
 public static final int KEYCODE\_MEDIA\_PAUSE = 127;  
 public static final int KEYCODE\_MEDIA\_PLAY = 126;  
 public static final int KEYCODE\_MEDIA\_PLAY\_PAUSE = 85;  
 public static final int KEYCODE\_MEDIA\_PREVIOUS = 88;  
 public static final int KEYCODE\_MEDIA\_RECORD = 130;  
 public static final int KEYCODE\_MEDIA\_REWIND = 89;  
 public static final int KEYCODE\_MEDIA\_SKIP\_BACKWARD = 273;  
 public static final int KEYCODE\_MEDIA\_SKIP\_FORWARD = 272;  
 public static final int KEYCODE\_MEDIA\_STEP\_BACKWARD = 275;  
 public static final int KEYCODE\_MEDIA\_STEP\_FORWARD = 274;  
 public static final int KEYCODE\_MEDIA\_STOP = 86;  
 public static final int KEYCODE\_MEDIA\_TOP\_MENU = 226;  
 public static final int KEYCODE\_MENU = 82;  
 public static final int KEYCODE\_META\_LEFT = 117;  
 public static final int KEYCODE\_META\_RIGHT = 118;  
 public static final int KEYCODE\_MINUS = 69;  
 public static final int KEYCODE\_MOVE\_END = 123;  
 public static final int KEYCODE\_MOVE\_HOME = 122;  
 public static final int KEYCODE\_MUHENKAN = 213;  
 public static final int KEYCODE\_MUSIC = 209;  
 public static final int KEYCODE\_MUTE = 91;  
 public static final int KEYCODE\_N = 42;  
 public static final int KEYCODE\_NAVIGATE\_IN = 262;  
 public static final int KEYCODE\_NAVIGATE\_NEXT = 261;  
 public static final int KEYCODE\_NAVIGATE\_OUT = 263;  
 public static final int KEYCODE\_NAVIGATE\_PREVIOUS = 260;  
 public static final int KEYCODE\_NOTIFICATION = 83;  
 public static final int KEYCODE\_NUM = 78;  
 public static final int KEYCODE\_NUMPAD\_0 = 144;  
 public static final int KEYCODE\_NUMPAD\_1 = 145;  
 public static final int KEYCODE\_NUMPAD\_2 = 146;  
 public static final int KEYCODE\_NUMPAD\_3 = 147;  
 public static final int KEYCODE\_NUMPAD\_4 = 148;  
 public static final int KEYCODE\_NUMPAD\_5 = 149;  
 public static final int KEYCODE\_NUMPAD\_6 = 150;  
 public static final int KEYCODE\_NUMPAD\_7 = 151;  
 public static final int KEYCODE\_NUMPAD\_8 = 152;  
 public static final int KEYCODE\_NUMPAD\_9 = 153;  
 public static final int KEYCODE\_NUMPAD\_ADD = 157;  
 public static final int KEYCODE\_NUMPAD\_COMMA = 159;  
 public static final int KEYCODE\_NUMPAD\_DIVIDE = 154;  
 public static final int KEYCODE\_NUMPAD\_DOT = 158;  
 public static final int KEYCODE\_NUMPAD\_ENTER = 160;  
 public static final int KEYCODE\_NUMPAD\_EQUALS = 161;  
 public static final int KEYCODE\_NUMPAD\_LEFT\_PAREN = 162;  
 public static final int KEYCODE\_NUMPAD\_MULTIPLY = 155;  
 public static final int KEYCODE\_NUMPAD\_RIGHT\_PAREN = 163;  
 public static final int KEYCODE\_NUMPAD\_SUBTRACT = 156;  
 public static final int KEYCODE\_NUM\_LOCK = 143;  
 public static final int KEYCODE\_O = 43;  
 public static final int KEYCODE\_P = 44;  
 public static final int KEYCODE\_PAGE\_DOWN = 93;  
 public static final int KEYCODE\_PAGE\_UP = 92;  
 public static final int KEYCODE\_PAIRING = 225;  
 public static final int KEYCODE\_PASTE = 279;  
 public static final int KEYCODE\_PERIOD = 56;  
 public static final int KEYCODE\_PICTSYMBOLS = 94;  
 public static final int KEYCODE\_PLUS = 81;  
 public static final int KEYCODE\_POUND = 18;  
 public static final int KEYCODE\_POWER = 26;  
 public static final int KEYCODE\_PROG\_BLUE = 186;  
 public static final int KEYCODE\_PROG\_GREEN = 184;  
 public static final int KEYCODE\_PROG\_RED = 183;  
 public static final int KEYCODE\_PROG\_YELLOW = 185;  
 public static final int KEYCODE\_Q = 45;  
 public static final int KEYCODE\_R = 46;  
 public static final int KEYCODE\_REFRESH = 285;  
 public static final int KEYCODE\_RIGHT\_BRACKET = 72;  
 public static final int KEYCODE\_RO = 217;  
 public static final int KEYCODE\_S = 47;  
 public static final int KEYCODE\_SCROLL\_LOCK = 116;  
 public static final int KEYCODE\_SEARCH = 84;  
 public static final int KEYCODE\_SEMICOLON = 74;  
 public static final int KEYCODE\_SETTINGS = 176;  
 public static final int KEYCODE\_SHIFT\_LEFT = 59;  
 public static final int KEYCODE\_SHIFT\_RIGHT = 60;  
 public static final int KEYCODE\_SLASH = 76;  
 public static final int KEYCODE\_SLEEP = 223;  
 public static final int KEYCODE\_SOFT\_LEFT = 1;  
 public static final int KEYCODE\_SOFT\_RIGHT = 2;  
 public static final int KEYCODE\_SOFT\_SLEEP = 276;  
 public static final int KEYCODE\_SPACE = 62;  
 public static final int KEYCODE\_STAR = 17;  
 public static final int KEYCODE\_STB\_INPUT = 180;  
 public static final int KEYCODE\_STB\_POWER = 179;  
 public static final int KEYCODE\_STEM\_1 = 265;  
 public static final int KEYCODE\_STEM\_2 = 266;  
 public static final int KEYCODE\_STEM\_3 = 267;  
 public static final int KEYCODE\_STEM\_PRIMARY = 264;  
 public static final int KEYCODE\_SWITCH\_CHARSET = 95;  
 public static final int KEYCODE\_SYM = 63;  
 public static final int KEYCODE\_SYSRQ = 120;  
 public static final int KEYCODE\_SYSTEM\_NAVIGATION\_DOWN = 281;  
 public static final int KEYCODE\_SYSTEM\_NAVIGATION\_LEFT = 282;  
 public static final int KEYCODE\_SYSTEM\_NAVIGATION\_RIGHT = 283;  
 public static final int KEYCODE\_SYSTEM\_NAVIGATION\_UP = 280;  
 public static final int KEYCODE\_T = 48;  
 public static final int KEYCODE\_TAB = 61;  
 public static final int KEYCODE\_TV = 170;  
 public static final int KEYCODE\_TV\_ANTENNA\_CABLE = 242;  
 public static final int KEYCODE\_TV\_AUDIO\_DESCRIPTION = 252;  
 public static final int KEYCODE\_TV\_AUDIO\_DESCRIPTION\_MIX\_DOWN = 254;  
 public static final int KEYCODE\_TV\_AUDIO\_DESCRIPTION\_MIX\_UP = 253;  
 public static final int KEYCODE\_TV\_CONTENTS\_MENU = 256;  
 public static final int KEYCODE\_TV\_DATA\_SERVICE = 230;  
 public static final int KEYCODE\_TV\_INPUT = 178;  
 public static final int KEYCODE\_TV\_INPUT\_COMPONENT\_1 = 249;  
 public static final int KEYCODE\_TV\_INPUT\_COMPONENT\_2 = 250;  
 public static final int KEYCODE\_TV\_INPUT\_COMPOSITE\_1 = 247;  
 public static final int KEYCODE\_TV\_INPUT\_COMPOSITE\_2 = 248;  
 public static final int KEYCODE\_TV\_INPUT\_HDMI\_1 = 243;  
 public static final int KEYCODE\_TV\_INPUT\_HDMI\_2 = 244;  
 public static final int KEYCODE\_TV\_INPUT\_HDMI\_3 = 245;  
 public static final int KEYCODE\_TV\_INPUT\_HDMI\_4 = 246;  
 public static final int KEYCODE\_TV\_INPUT\_VGA\_1 = 251;  
 public static final int KEYCODE\_TV\_MEDIA\_CONTEXT\_MENU = 257;  
 public static final int KEYCODE\_TV\_NETWORK = 241;  
 public static final int KEYCODE\_TV\_NUMBER\_ENTRY = 234;  
 public static final int KEYCODE\_TV\_POWER = 177;  
 public static final int KEYCODE\_TV\_RADIO\_SERVICE = 232;  
 public static final int KEYCODE\_TV\_SATELLITE = 237;  
 public static final int KEYCODE\_TV\_SATELLITE\_BS = 238;  
 public static final int KEYCODE\_TV\_SATELLITE\_CS = 239;  
 public static final int KEYCODE\_TV\_SATELLITE\_SERVICE = 240;  
 public static final int KEYCODE\_TV\_TELETEXT = 233;  
 public static final int KEYCODE\_TV\_TERRESTRIAL\_ANALOG = 235;  
 public static final int KEYCODE\_TV\_TERRESTRIAL\_DIGITAL = 236;  
 public static final int KEYCODE\_TV\_TIMER\_PROGRAMMING = 258;  
 public static final int KEYCODE\_TV\_ZOOM\_MODE = 255;  
 public static final int KEYCODE\_U = 49;  
 public static final int KEYCODE\_UNKNOWN = 0;  
 public static final int KEYCODE\_V = 50;  
 public static final int KEYCODE\_VOICE\_ASSIST = 231;  
 public static final int KEYCODE\_VOLUME\_DOWN = 25;  
 public static final int KEYCODE\_VOLUME\_MUTE = 164;  
 public static final int KEYCODE\_VOLUME\_UP = 24;  
 public static final int KEYCODE\_W = 51;  
 public static final int KEYCODE\_WAKEUP = 224;  
 public static final int KEYCODE\_WINDOW = 171;  
 public static final int KEYCODE\_X = 52;  
 public static final int KEYCODE\_Y = 53;  
 public static final int KEYCODE\_YEN = 216;  
 public static final int KEYCODE\_Z = 54;  
 public static final int KEYCODE\_ZENKAKU\_HANKAKU = 211;  
 public static final int KEYCODE\_ZOOM\_IN = 168;  
 public static final int KEYCODE\_ZOOM\_OUT = 169;

# 控件查找器

from ascript.android.node import Selector

控件查找器,通过约束方法,和查询方法,**来获取**屏幕中满足条件的控件.

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 控件缓存[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "控件缓存" \o "Direct link to 控件缓存)

是否启用控件缓存机制.

info

启用后,控件将不在刷新,请及时释放缓存

启用后,查找速度将有大幅度提升.

* 函数

Selector.cache(is\_cache:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| is\_cache | bool | 是 | True:打开缓存,缓存当前页面所有控件 False:关闭缓存 |

* 示例

*# 开启缓存*  
from ascript.android.node import Selector  
  
Selector.cache(True)

*# 关闭控件缓存*  
from ascript.android.node import Selector  
  
Selector.cache(False)

### 创建查找器[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建查找器" \o "Direct link to 创建查找器)

创建一个控件查找器

* 函数

*#class*  
Selector(mode:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| mode | int | 否 | 检索模式  0(默认):普通检索模式,这种模式下只展示重要控件 1:复杂模式,这种模式下展示所有控件.层级会很深 2:简单模式下过滤掉系统控件.  3:复杂模式下过滤掉系统控件 |

* 示例

from ascript.android.node import Selector  
*# 创建一个选择器,简单模式检索*  
selector = Selector()

from ascript.android.node import Selector  
*# 创建一个选择器 ,复杂模式检索*  
Selector(1)

from ascript.android.node import Selector  
*# 创建一个选择器 ,简单模式,并且过滤掉系统控件*  
Selector(0|2)

from ascript.android.node import Selector  
*# 创建一个选择器 ,复杂模式,并且过滤掉系统控件*  
Selector(1|2)

### 查找控件一个[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "查找控件一个" \o "Direct link to 查找控件一个)

查找控件,返回一个满足条件的控件

* 函数

*#class*  
Selector(mode:int).find()

* 返回值

[Node](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E6%8E%A7%E4%BB%B6%E5%AF%B9%E8%B1%A1) 控件对象

* 示例

*#案例:查找出符合条件的第一个控件*  
from ascript.android.node import Selector  
node = Selector().find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

### 查找控件全部[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "查找控件全部" \o "Direct link to 查找控件全部)

查找控件,返回满足条件的控件

* 函数

*#class*  
Selector(mode:int).find\_all(num:int=999999)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| num | int | 否 | 控件结果最大数量 默认:满足条件的所有 |

* 返回值

[Node[]](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E6%8E%A7%E4%BB%B6%E5%AF%B9%E8%B1%A1) 控件对象列表

* 示例

*#案例:查找出所有符合条件的控件*  
from ascript.android.node import Selector  
nodes = Selector().find\_all()  
  
if nodes:  
 *#找到了一组控件*  
 print( len(nodes) )  
else:  
 print('没有找到任何控件')

### 获取控件树[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取控件树" \o "Direct link to 获取控件树)

获取完整的控件树,返回Json树状格式

* 函数

*#class*  
Selector.dump(mode:int=0)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| mode | int | 否 | 检索模式  0(默认):普通检索模式,这种模式下只展示重要控件 1:复杂模式,这种模式下展示所有控件.层级会很深 2:简单模式下过滤掉系统控件.  3:复杂模式下过滤掉系统控件 |

* 返回值

Json 数据 ,可转换为Python字典

* 示例

*#案例:获取控件树json数据,并转换为字典*  
import json  
*# 导入节点检索模块*  
from ascript.android.node import Selector  
res = Selector.dump()  
print(res) *# 这是json字符串*  
if res:  
 *# 转换为字典对象*  
 obj = json.loads(res);  
 print(obj)

### 属性约束[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "属性约束" \o "Direct link to 属性约束)

#### id[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "id" \o "Direct link to id)

通过 id 约束

* 函数

Selector().id(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 控件ID,支持正则表达式,**通常用控件查找器获取** |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 导包*  
from ascript.android.node import Selector  
  
*# 查找id值为 “com.aojoy.airscript:id/search\_bar\_text” 的控件*  
node = Selector().id("com.aojoy.airscript:id/search\_bar\_text").find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

*# 案例2: 在id约束中使用正则表达式*  
*# 导包*  
from ascript.android.node import Selector  
  
*# 查找id值为 ‘search\_bar\_text结尾’ 的控件 ,在正则中 ‘.’代表任何值,‘\*’代表出现多次,‘^’代表以什么开头,‘$’代表以什么结尾*  
node = Selector().id(".\*search\_bar\_text$").find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 文本[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "文本" \o "Direct link to 文本)

通过 text 约束

* 函数

Selector().text(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 屏幕中显示的任意文本 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例1: 在text约束中使用完整文本*  
*# 导包*  
from ascript.android.node import Selector  
  
*# 查找文本为 '开发者'*  
node = Selector().text("开发者").find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

*# 案例2: 在text约束中使用正则匹配 所有文本中包含 'AirScript' 的控件*  
*# 导包*  
from ascript.android.node import Selector  
  
*# 查找 文本 中包含 了‘IP’ 的控件*  
nodes = Selector().text(".\*IP.\*").find\_all()  
  
if nodes:  
 *#找到了控件*  
 print( len(nodes))  
else:  
 print('没有找到任何控件')

#### 类型[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "类型" \o "Direct link to 类型)

通过 type 约束

* 函数

Selector().type(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 控件的类型 如:**TextView**, **ListView** 等等 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出 所有类型为 TextView 的控件*  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 type 为“TextView”*  
nodes = Selector().type("TextView").find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 描述[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "描述" \o "Direct link to 描述)

通过 控件描述desc 约束

* 函数

Selector().desc(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 控件的描述信息 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出 描述信息为 ‘这是关闭按钮’ 的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 desc 为“这是关闭按钮”*  
node = Selector().desc("这是关闭按钮").find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 默认文本[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "默认文本" \o "Direct link to 默认文本)

通过控件 **默认文本** hintText 约束

* 函数

Selector().hintText(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 控件的默认文本 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出 默认文本 包含 “小程序” 的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 hint\_text 包含 “小程序” 的控件*  
node = Selector().hintText(".\*小程序.\*").find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 包名[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "包名" \o "Direct link to 包名)

通过控件 **包名** packageName 约束

* 函数

Selector().packageName(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 控件的所属包名 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出 包名 为 com.aojoy.airscript 的控件*  
from ascript.android.node import Selector  
  
nodes = Selector().packageName("com.aojoy.airscript").find\_all()  
  
if nodes:  
 *#找到了控件*  
 print(nodes)  
else:  
 print('没有找到任何控件')

#### 路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "路径" \o "Direct link to 路径)

通过控件 **路径** path 约束

* 函数

Selector().path(val:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | str正则 | 是 | 控件的路径 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出 路径 为 /FrameLayout/LinearLayout/FrameLayout/EditText 的控件*  
from ascript.android.node import Selector  
  
node = Selector().path("/FrameLayout/LinearLayout/FrameLayout/EditText").find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 子控件数量[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "子控件数量" \o "Direct link to 子控件数量)

通过控件 **子控件数量** childCount 约束

* 函数

Selector().childCount(val:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | int | 是 | 子控件的数量 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例1: 查找 子控件个数 为 13 的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
node = Selector().childCount(13).find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

*#案例2: 查找 子控件个数 为 6-7 之间, 9-13 之间的 所有*   
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 desc 为“这是关闭按钮”*  
nodes = Selector().childCount(6.7,9.13).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for n in nodes:  
 print(n)  
else:  
 print('没有找到任何控件')

#### 输入类型[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "输入类型" \o "Direct link to 输入类型)

通过控件 **输入类型** inputType 约束

* 函数

Selector().inputType(val:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | int | 是 | 控件的输入类型 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找输入类型为 524289 的 控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 输入类型 为1*  
node = Selector().inputType(524289).find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 绘制排序[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "绘制排序" \o "Direct link to 绘制排序)

通过控件 **绘制排序** drawingOrder 约束

绘制排序不是很可信

绘制排序,部分情况下会变化.请不要过度依赖该属性

* 函数

Selector().drawingOrder(val:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | int | 是 | 控件的绘制排名 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找出 绘制序列属性 为38的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 drawingOrder 属性为2的控件*  
node = Selector().drawingOrder(38).find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 控件深度[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "控件深度" \o "Direct link to 控件深度)

通过控件 **控件深度** depth 约束

* 函数

Selector().depth(val:int):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | int | 是 | 控件的层级深度 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找出 绘制深度属性 为4的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
node = Selector().depth(4).find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 最大文本长度[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "最大文本长度" \o "Direct link to 最大文本长度)

通过控件 **最大文本长度** maxTextLength 约束

* 函数

Selector().maxTextLength(val:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | int | 是 | 控件的最大文本内容长度 通常用控件查找器获取 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找最大文本长度 限制为 5000 的 控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 最大文本长度 为 50*  
node = Selector().maxTextLength(5000).find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否可点击 约束[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否可点击-约束" \o "Direct link to 是否可点击 约束)

通过控件 clickable 约束

* 函数

Selector().clickable(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:可以点击,False:不可点击 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找出当前页面中 所有可以被点击的控件*  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 可以点击*  
nodes = Selector().clickable(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否可以被选中[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否可以被选中" \o "Direct link to 是否可以被选中)

通过控件 checkable 约束

* 函数

Selector().checkable(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:可被选中,False:不可被选中 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出当前页面中 所有可以选中的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 可以选中*  
nodes = Selector().checkable(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否选中[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否选中" \o "Direct link to 是否选中)

通过控件 checked 约束

* 函数

Selector().checked(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:已选中,False:未选中 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找出当前页面中 所有已经选中的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 可以选中*  
nodes = Selector().checked(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否可编辑[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否可编辑" \o "Direct link to 是否可编辑)

通过控件 editable 约束

* 函数

Selector().editable(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:可编辑,False:不可编辑 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出当前页面中 所有可以编辑文本的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 可以编辑*  
nodes = Selector().editable(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否可访问[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否可访问" \o "Direct link to 是否可访问)

通过控件 enabled 约束

* 函数

Selector().enabled(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:可操作,如点击,输入,False:不可操作,点击,输入无效 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出当前页面中 所有不可以访问的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 不可以访问*  
nodes = Selector().enabled(False).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否可被关闭[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否可被关闭" \o "Direct link to 是否可被关闭)

通过控件 dismissable 约束

* 函数

Selector().dismissable(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:可关闭,False:不可关闭 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出当前页面中 所有可以被关闭的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 不可以访问*  
nodes = Selector().dismissable(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否可以获取焦点[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否可以获取焦点" \o "Direct link to 是否可以获取焦点)

通过控件 focusable 约束

* 函数

Selector().focusable(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:可获取焦点,False:不允许获取焦点 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找出当前页面中 所有所有可以被获取焦点的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 可以获取焦点的控件*  
nodes = Selector().focusable(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否有焦点[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否有焦点" \o "Direct link to 是否有焦点)

通过控件 focused 约束

* 函数

Selector().focused(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:已获取焦点,False:没有焦点 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出当前页面中 所有所有可以被获取焦点的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 已经获取焦点*  
node = Selector().focused(True).find()  
  
if node:  
 *#找到了控件*  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否显示中[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否显示中" \o "Direct link to 是否显示中)

通过控件 visible 约束

* 函数

Selector().visible(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:显示在屏幕上 False:隐藏或在屏幕之外 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例: 查找出,所有用户不可见的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().visible(False).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 是否支持长按[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "是否支持长按" \o "Direct link to 是否支持长按)

通过控件 longClickable 约束

* 函数

Selector().longClickable(val:bool)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| val | bool | 是 | True:支持 False:不支持 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*#案例: 查找出当前页面中 所有支持长按的控件*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 已经获取焦点*  
nodes = Selector().longClickable(True).find\_all()  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

### 关系约束[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "关系约束" \o "Direct link to 关系约束)

#### 获取父控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取父控件" \o "Direct link to 获取父控件)

通过控件 parent 约束

* 函数

Selector().parent(\*val:float)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| \*val | float可变参数 | 是 | 获取第val个父控件 默认获取所有父元素 (2):获取爷爷元素 (3):获取太爷爷元素 (1,3):获取第1和第3个父元素 (1.3):获取第1-3 之间的所有父元素 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例1: 获取指定 id控件的 所有父元素(父元素,爷爷元素,太爷爷,祖太爷...统统获取)*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 获取所有父元素*  
nodes = Selector().id("com.aojoy.airscript:id/search\_bar\_text").parent().find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

*# 案例2: 获取指定 id控件的 太爷爷元素*  
  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().id("com.aojoy.airscript:id/search\_bar\_text").parent(3).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

*# 案例3: 获取父元素,太太爷元素*  
  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().id("com.aojoy.airscript:id/search\_bar\_text").parent(1,3).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

*# 案例4: 获取往上数3辈 所有元素*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 获取1-3 之间的所有父元素*  
nodes = Selector().id("com.aojoy.airscript:id/search\_bar\_text").parent(1.3).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node)  
else:  
 print('没有找到任何控件')

#### 获取子控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取子控件" \o "Direct link to 获取子控件)

通过控件 child 约束

* 函数

Selector().child(\*val:float)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| \*val | float可变参数 | 是 | 获取第val个子控件 默认获取所有孩子控件 (1):获取第1个孩子控件 (-1):获取倒数第1个孩子 (1.3):获取1-3之间的所有孩子 (-1.3):获取倒数 1-3之间的所有孩子  (1,3):获取第1个和第3个子控件 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例1: 获取所有孩子*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 默认无参下,获取所有*  
nodes = Selector().id("com.aojoy.airscript:id/search\_query\_section").child().find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.type)  
else:  
 print('没有找到任何控件')

*# 案例2: 获取第一个孩子*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 获取第一个孩子*  
nodes = Selector().id("com.aojoy.airscript:id/search\_query\_section").child(1).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.type)  
else:  
 print('没有找到任何控件')

*# 案例3: 获取第1和 第2个孩子*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 可变参数,填写多个*  
nodes = Selector().id("com.aojoy.airscript:id/search\_query\_section").child(1,2).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.type)  
else:  
 print('没有找到任何控件')

*# 案例4: 获取第1-3个孩子*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 填写一个小数,获取区间孩子*  
nodes = Selector().id("com.aojoy.airscript:id/search\_query\_section").child(1.3).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.type)  
else:  
 print('没有找到任何控件')

*# 案例5: 获取第倒数第一个孩子*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 负数,倒数取值*  
nodes = Selector().id("com.aojoy.airscript:id/search\_query\_section").child(-1).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.type)  
else:  
 print('没有找到任何控件')

*# 案例6: 获取第倒数第1-3 个孩子*  
  
*# 导包*  
from ascript.android.node import Selector  
  
*# 约束 负数,倒数区间取值*  
nodes = Selector().id("com.aojoy.airscript:id/search\_query\_section").child(-1.3).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.type)  
else:  
 print('没有找到任何控件')

#### 获取兄弟控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取兄弟控件" \o "Direct link to 获取兄弟控件)

通过控件 brother 约束

* 函数

Selector().brother(\*val:float)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| \*val | float可变参数 | 是 | 获取第val个子控件 默认获取所有兄弟控件 (1):获取第1个兄弟控件  (1,2):获取第1和第2个兄弟控件 (1.4):获取1-4之间的所有兄弟控件 (0.1):获取当前控件的下一个兄弟控件  (-0.1):获取当前控件的上一个兄弟控件 (-1):获取倒数第1个兄弟控件 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 案例1: 获取所有兄弟*  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother().find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

*# 案例2: 获取第一个兄弟控件*   
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother(1).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

*# 案例3: 获取第1和第2个兄弟控件*  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother(1,2).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

*# 案例4:获取1-4之间的所有兄弟控件*  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother(1.4).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

*# 案例5:获取当前控件的下一个兄弟控件*  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother(0.1).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

*# 案例6:获取当前控件的上一个兄弟控件*  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother(-0.1).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

*# 案例7:获取倒数第1个兄弟控件*  
*# 导包*  
from ascript.android.node import Selector  
  
nodes = Selector().desc(".\*充电.\*").brother(-1).find\_all();  
  
if nodes:  
 *#找到了控件*  
 for node in nodes:  
 print(node.desc)  
else:  
 print('没有找到任何控件')

### 动作约束[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "动作约束" \o "Direct link to 动作约束)

#### 点击控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "点击控件" \o "Direct link to 点击控件)

通过控件 click 约束

* 函数

Selector().click()

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 查找 到指定控件,执行点击方法, “点击方法也是约束的一种,因此不能忘记后面的find或\_find\_all”*  
*#导包*  
from ascript.android.node import Selector  
Selector().type("ImageView").id("com.aojoy.airscript:id/iv\_statue1").click().find()

#### 长按控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "长按控件" \o "Direct link to 长按控件)

通过控件 long\_click 约束

* 函数

Selector().long\_click()

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 指定控件 长按操作*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/gv\_historyrun").child(1).long\_click().find()

#### 滑动控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "滑动控件" \o "Direct link to 滑动控件)

通过控件 slide 约束

* 函数

Selector().slide(ore:int=1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| ore | int | 是 | 滑动方向 1:向前滑动-(默认) -1:向后滑动 |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 查找 到指定可控件 ,执行向后滑动*  
*#导包*  
from ascript.android.node import Selector  
  
Selector().type("RecyclerView").slide(1).find()

#### 向控件输入信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "向控件输入信息" \o "Direct link to 向控件输入信息)

通过控件 input 约束

* 函数

Selector().input(msg:str="")

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| msg | str | 否 | 输入的信息 "":清空输入框中的值 (默认) |

* 返回值

[Selector](F:/cs01/cs01/cj2/1extracted_div_content.html) 返回控件查找器,用于链式编程

* 示例

*# 查找 到指定可控件 , 输入‘自在老师’*  
*#导包*  
from ascript.android.node import Selector  
  
Selector().id("com.aojoy.airscript:id/search\_bar\_text").input("自在老师").find()

# 控件对象

from ascript.android.node import Node

通过控件选择器 **Selector()** [.find()](F:/cs01/cs01/cj2/docs/android/api/node/selector/" \l "%E6%9F%A5%E6%89%BE%E6%8E%A7%E4%BB%B6%E4%B8%80%E4%B8%AA) 或 [.find\_all()](F:/cs01/cs01/cj2/docs/android/api/node/selector/" \l "%E6%9F%A5%E6%89%BE%E6%8E%A7%E4%BB%B6%E5%85%A8%E9%83%A8) 而获得此对象

## 属性[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "属性" \o "Direct link to 属性)

| **属性** | **备注** |
| --- | --- |
| id | 控件ID 部分APP中ID属性,随手机安装可能动态变化,谨慎使用 |
| text | 控件的文本 |
| type | 控件的类型 |
| desc | 控件的描述 |
| hintText | 控件的默认展示文本 |
| packageName | 控件所属包名 |
| rect | 控件在屏幕中的位置 **rect 属性** rect.left:x坐标 rect.top:y坐标 **rect 方法** rect.width():控件的宽度 rect.height():控件的高度 rect.centerX():控件的中心坐标X rect.centerY():控件的中心坐标Y |
| childCount | 子控件数量 |
| inputType | 输入类型 |
| maxTextLength | 控件最大文本长度 |
| drawingOrder | 是否可点击 |
| checkable | 是否可选中 |
| checked | 是否已选中 |
| editable | 是否支持编辑 |
| enabled | 是否可访问 |
| visible | 是否针对用户展示 |
| dismissable | 是否可取消 |
| focusable | 是否可以获取焦点 |
| focused | 是否已获取了焦点 |
| longClickable | 是否可以长按 |
| center\_x | 控件中心坐标x |
| center\_y | 控件中心坐标y |

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 内部查找[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "内部查找" \o "Direct link to 内部查找)

在此控件的子控件中继续查找

#### 查找一个[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "查找一个" \o "Direct link to 查找一个)

找到满足条件的第一个控件

* 函数

控件对象.find(selector:Selector)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| selector | [Selector](F:/cs01/cs01/cj2/docs/android/api/node/selector/) | 是 | 控件选择器 |

* 返回值

[Node](F:/cs01/cs01/cj2/1extracted_div_content.html) 控件对象

* 示例

*#案例:查找到一个控件,再此基础上再次进行二次查找*  
  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_query\_section").type("FrameLayout").find()  
  
if node:  
 *# 在node 控件中 查找 它的倒数第一个子控件*  
 nchild = node.find( Selector().child(-1) )  
 if nchild:  
 print(nchild)

#### 查找全部[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "查找全部" \o "Direct link to 查找全部)

找到满足条件的所有控件

* 函数

.find\_all(selector:Selector)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| selector | [Selector](F:/cs01/cs01/cj2/docs/android/api/node/selector/) | 是 | 控件选择器 |

* 返回值

[Node[]](F:/cs01/cs01/cj2/1extracted_div_content.html) 控件对象列表

* 示例

*#案例:查找到一个控件,再此基础上再次进行二次查找*  
  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_query\_section").type("FrameLayout").find()  
  
if node:  
 *# 在node 控件中 查找 它的所有孩子控件*  
 nchilds = node.find\_all( Selector().child() )  
 if nchilds:  
 for nchild in nchilds:  
 print(nchild)

### 获取父控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取父控件" \o "Direct link to 获取父控件)

找到满足条件的父控件

* 函数

.parent(\*val: float)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| \*val | float可变参数 | 是 | 获取第val个父控件 默认获取所有父元素 (2):获取爷爷元素 (3):获取太爷爷元素 (1,3):获取第1和第3个父元素 (1.3):获取第1-3 之间的所有父元素 |

* 返回值

[Node](F:/cs01/cs01/cj2/1extracted_div_content.html) 或 [Node[]](F:/cs01/cs01/cj2/1extracted_div_content.html) 结果为一个的时候:控件对象 返回结果多个的时候:列表

* 示例

*#案例1:查找到一个控件,再此基础上再次进行二次查找*  
  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_query\_section").type("FrameLayout").find()  
  
if node:  
 *# 获取node 控件的父元素*  
 nP = node.parent(1)  
 if nP:  
 print(nP)

### 获取子控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取子控件" \o "Direct link to 获取子控件)

找到满足条件的子控件

* 函数

.child(\*val: float)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| \*val | float可变参数 | 是 | 获取第val个子控件 默认获取所有孩子控件 (1):获取第1个孩子控件 (-1):获取倒数第1个孩子 (1.3):获取1-3之间的所有孩子 (-1.3):获取倒数 1-3之间的所有孩子  (1,3):获取第1个和第3个子控件 |

* 返回值

[Node](F:/cs01/cs01/cj2/1extracted_div_content.html) 或 [Node[]](F:/cs01/cs01/cj2/1extracted_div_content.html) 结果为一个的时候:控件对象 返回结果多个的时候:列表

* 示例

*# 二次检索 获取node 控件的第一个孩子*  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_query\_section").type("FrameLayout").find()  
  
if node:  
 node\_child = node.child(1)  
 if node\_child:  
 print(node\_child)

*# 二次检索 获取node 控件的所有孩子*  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_query\_section").type("FrameLayout").find()  
  
if node:  
 node\_childs = node.child()  
 if node\_childs:  
 print(len(node\_childs))

### 获取兄弟控件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取兄弟控件" \o "Direct link to 获取兄弟控件)

找到满足条件的兄弟控件

* 函数

.brother(\*val: float)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| \*val | float可变参数 | 是 | 获取第val个子控件 默认获取所有兄弟控件 (1):获取第1个兄弟控件  (1,2):获取第1和第2个兄弟控件 (1.4):获取1-4之间的所有兄弟控件 (0.1):获取当前控件的下一个兄弟控件  (-0.1):获取当前控件的上一个兄弟控件 (-1):获取倒数第1个兄弟控件 |

* 返回值

[Node](F:/cs01/cs01/cj2/1extracted_div_content.html) 或 [Node[]](F:/cs01/cs01/cj2/1extracted_div_content.html) 结果为一个的时候:控件对象 返回结果多个的时候:列表

* 示例

*# 二次检索 # 获取node 控件的所有兄弟控件第1个*  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_query\_section").type("FrameLayout").find()  
  
if node:  
 node\_brother = node.brother(1)  
 if node\_brother:  
 print(node\_brother)

### 控件点击[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "控件点击" \o "Direct link to 控件点击)

立即指定控件点击, 控件无论是否在界面上显示与否,都可点击

* 函数

.click()

* 示例

*# 点击控件自身*  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/search\_bar\_text").find()  
if node:  
 node.click()

### 控件长按[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "控件长按" \o "Direct link to 控件长按)

立即指定控件长按, 控件无论是否在界面上显示与否,都可长按

* 函数

.long\_click()

* 示例

*# 点击控件自身*  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().id("com.aojoy.airscript:id/gv\_historyrun").child(1).find()  
if node:  
 node.long\_click()

### 控件滑动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "控件滑动" \o "Direct link to 控件滑动)

立即指定控件滑动, 控件无论是否在界面上显示与否,都可滑动

* 函数

.s li de(ore:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| ore | int | 是 | 滑动方向 1:向前滑动-(默认) -1:向后滑动 |

* 示例

*# 控件向前滑动*  
*#导包*  
from ascript.android.node import Selector  
  
node = Selector().type("RecyclerView").find()  
if node:  
 node.slide(-1)

### 控件输入[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "控件输入" \o "Direct link to 控件输入)

立即指定控件输入信息, 控件无论是否在界面上显示与否,都可输入

* 函数

.input(msg:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| msg | str | 否 | 输入的信息 "":清空输入框中的值 (默认) |

* 示例

*# 当前控件 输入文本*  
from ascript.android.node import Selector  
node = Selector().id("com.aojoy.airscript:id/search\_bar\_text").type("EditText").find()  
if node:  
 node.input("自在老师讲的不错")

# 屏幕图像

from ascript.android import screen

从屏幕中 获取图像数据, 图像转换 等操作

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 屏幕缓存[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "屏幕缓存" \o "Direct link to 屏幕缓存)

屏幕图像数据 **缓存至某一时刻** ,后续所有的图色操作都基于缓存运行.

* 函数

screen.cache(is\_cache:bool=False,bitmap=None,file=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| is\_cache | bool | 否 | 是否缓存 True:缓存 False:取消缓存 |
| bitmap | Bitmap图片对象 | 否 | 是否将一张android bitmap 作为图色缓存 默认为None,缓存当前屏幕 如填入Bitmap参数,所有图色方法都从这个Bitmap图像中检测 |
| file | 图片文件地址 | 否 | 是否将一个图片文件作为图色缓存 默认为None,缓存当前屏幕如填入对应的图片文件地址,所有图色方法都从这个图片文件中检测 |

* 示例

*# 打开屏幕缓存*  
from ascript.android import screen  
  
screen.cache(True)

*# 关闭屏幕缓存*  
from ascript.android import screen  
  
screen.cache(False)

*# 将一个图片文件缓存为屏幕图像,所有图像方法都从这个图片文件中检测*  
from ascript.android import screen  
from ascript.android.system import R  
  
*# 将sd卡中的a.png图片作为屏幕图像缓存起来.*  
screen.cache(False,file=R.sd("a.png"))

### 截屏[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "截屏" \o "Direct link to 截屏)

获取一张当前屏幕的内存截图,截图格式为 **Android Bitmap**

如果需要 将图像存储为文件,清参考 [图像转文件](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E5%9B%BE%E5%83%8F%E8%BD%AC%E6%96%87%E4%BB%B6)

* 函数

screen.capture(x:int=None,y:int=None,x1:int=None,y1:int=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x,y,x1,y1 | int | 否 | 截图的矩形范围坐标 默认:全屏截图 |

* 返回值

[Bitmap](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "###%E5%9B%BE%E5%83%8FBitmap) Android图片对象

* 示例

*# 截取屏幕*  
from ascript.android import screen  
bitmap = screen.capture()  
  
print(bitmap)

*# 截取 指定范围屏幕*  
from ascript.android import screen  
bitmap = screen.capture( 122,358,808,979)  
print(bitmap)

### 图像某坐标色值[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图像某坐标色值" \o "Direct link to 图像某坐标色值)

获取图像指定坐标的颜色值

* 函数

screen.get\_color(x:int,y:int,bitmap = screen.capture())

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| x | int | 是 | 坐标x |
| y | int | 是 | 坐标y |
| bitmap | int | 否 | 指定图像,默认:全屏截图 |

* 返回值

[Color](F:/cs01/cs01/cj2/1extracted_div_content.html) 颜色值对象,点击查看其内部属性值

* 示例

from ascript.android import screen  
color = screen.get\_color(200,200)  
  
print(color.rgb) *#16进制RGB颜色值*  
print(color.argb) *#16进制ARGB颜色值 PS:A位透明值通道*  
print(color.r,color.g,color.b,color.a) *# 获取R通道,G通道,B通道,A通道 10进制颜色值*

### 图像所有色值[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图像所有色值" \o "Direct link to 图像所有色值)

获取整张图像所有色值

* 函数

screen.get\_colors(bitmap = screen.capture())

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap | int | 否 | 指定图像,默认:全屏截图 |

* 返回值

[int[]](F:/cs01/cs01/cj2/1extracted_div_content.html) 10进制 颜色数组列表.可通过Color构造,获取RGB颜色值

* 示例

from ascript.android import screen  
from ascript.android.screen import Color  
colors = screen.get\_colors()  
  
print(f"共有:{len(colors)}个颜色点")  
  
for i in range(len(colors)):  
 *# 打印出屏幕第一个点的颜色值*  
 if i == 0:  
 c = Color(colors[i])  
 print(c.rgb)

### 区域颜色点数量[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "区域颜色点数量" \o "Direct link to 区域颜色点数量)

在一个区域内找 指定颜色点的数量

* 函数

screen.get\_color\_num(colors:str,rect:list=None,sim:float=0.9)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| colors | int | 否 | 颜色值: 格式1: "#FFFFFF" 检索单一颜色值数量 格式2:"#FFFFFF|#000000" 多种颜色满足其一则算数 格式3:"#000000-#111111" 给定颜色值 在 两个颜色值之间则算数 |
| rect | list | 否 | 屏幕范围 |
| sim | float | 否 | 相似度 0~1之间,默认0.9:90%相似度 |

* 返回值

[Bitmap](F:/cs01/cs01/cj2/1extracted_div_content.html) Android图片对象

* 示例

*# 获取屏幕中 #6989C5 和 #FF784B 颜色值数量*  
from ascript.android import screen  
num = screen.get\_color\_num("#6989C5|#FF784B")  
  
print(num)

*# 获取屏幕指定范围内 #6989C5 和 #FF784B 颜色值数量*  
from ascript.android import screen  
num = screen.get\_color\_num("#6989C5|#FF784B",rect =[413,358,709,578])  
  
print(num)

*# 获取屏幕指定范围内 #6989C5 和 #FF784B 相似度80% 的颜色点数量*  
from ascript.android import screen  
num = screen.get\_color\_num("#6989C5|#FF784B",rect =[413,358,709,578],sim=0.8)  
  
print(num)

### 图像转换[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图像转换" \o "Direct link to 图像转换)

#### 图像转文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图像转文件" \o "Direct link to 图像转文件)

将 **Android Bitmap** 转存为文件

* 函数

screen.bitmap\_to\_file(path:str,bitmap=screen.capture(),quality=100)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| path | str | 是 | 文件路径 |
| bitmap | Bitmap | 否 | 图片 默认:全屏截图 |
| quality | int | 否 | 图片清晰度,1-100之间 ,100为100%无损图片 默认:100 |

* 示例

*# 将屏幕截图存储在SD卡下的1.png中*  
from ascript.android import screen  
from ascript.android.system import R  
screen.bitmap\_to\_file(R.sd("1.png"))

*# 将屏幕截图存储在SD卡下的1.png中,清晰度设为80%*  
from ascript.android import screen  
from ascript.android.system import R  
screen.bitmap\_to\_file(R.sd("1.png"),quality=80)

#### 文件转图片[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "文件转图片" \o "Direct link to 文件转图片)

将 **文件** 转存为Android Bitmap

* 函数

screen.file\_to\_bitmap(path:str,sampleSize:int=1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| path | str | 是 | 文件路径 |
| sampleSize | int | 否 | 采样率,如参数为2:每隔2行,2列采集1行,结果为原图的1/4大小 默认1:无损 |

* 示例

*# 将屏幕截图存储在SD卡下的1.png中*  
from ascript.android import screen  
from ascript.android.system import R  
bitmap = screen.file\_to\_bitmap(R.sd("2.png"))  
print(bitmap)

#### 图片转Base64数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图片转base64数据" \o "Direct link to 图片转Base64数据)

将 **图片** 转为 **base64数据**

* 函数

screen.bitmap\_base64(bitmap=screen.capture())

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap | Bitmap | 否 | 图片 默认:将当前屏幕转为 base64数据 |

* 示例

*# 将屏幕图像转换为base64*  
from ascript.android import screen  
b64\_str = screen.bitmap\_base64()  
print(b64\_str)

*# 将文件转换为base64*  
from ascript.android import screen  
from ascript.android.system import R  
  
bitmap = screen.file\_to\_bitmap(R.sd("22222.png"))  
b64\_str = screen.bitmap\_base64(bitmap)  
print(b64\_str)

#### 图片最大边缩放[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图片最大边缩放" \o "Direct link to 图片最大边缩放)

将 **图片** 缩放, 最大边不超过设定值,短边等比缩放.

* 函数

screen.bitmap\_maxside(bitmap=screen.capture(),max\_side\_len=9999)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap | Bitmap | 否 | 要缩放的图片 默认:当前屏幕截图 |
| max\_side\_len | int | 是 | 最大边长度 ,默认最大边长 9999 |

* 示例

*# 将当前屏幕截图,最大边缩放到800*  
bitmap= screen.bitmap\_maxside(screen.capture(),max\_side\_len=800)  
print(bitmap.getWidth(),bitmap.getHeight())

#### Bitmap 转 Python Image[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "bitmap-转-python-image" \o "Direct link to Bitmap 转 Python Image)

将 **Android Bitmap** 转换为 **Python Image(PIL)**

* 函数

screen.bitmap\_to\_pilimage(bitmap=capture())

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap | Bitmap | 否 | 要转换的Bitmap图像 默认:当前屏幕截图 |

* 返回值

[Image](F:/cs01/cs01/cj2/1extracted_div_content.html) Python PIL 图像

* 示例

*# 将当前屏幕Bitmap 转换为 Python Image*  
from ascript.android import screen  
img= screen.bitmap\_to\_pilimage()  
print(img.width,img.height)

#### Bitmap 转 Opencv Image[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "bitmap-转-opencv-image" \o "Direct link to Bitmap 转 Opencv Image)

将 **Android Bitmap** 转换为 **Opencv Image**

由于android图像是 ARGB通道 ,因此在转换后会丢失A通道,并且图像数据 为BGR三通道格式

* 函数

screen.bitmap\_to\_cvimage(bitmap=capture())

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap | Bitmap | 否 | 要转换的Bitmap图像 默认:当前屏幕截图 |

* 返回值

[Numpy array](F:/cs01/cs01/cj2/1extracted_div_content.html) Python Opencv Image

* 示例

*# 将当前屏幕Bitmap 转换为 Openv Image = cv2.imread*  
import cv2  
from ascript.android.ui import ImageWindow  
from ascript.android import screen  
  
image = screen.bitmap\_to\_cvimage()  
  
*# 展示图片*  
ImageWindow.show(image)

#### Opencv Image 转 Bitmap[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "opencv-image-转-bitmap" \o "Direct link to Opencv Image 转 Bitmap)

将 **Opencv Image** 转换为 **Android Bitmap**

将BGR通道的CV图像 转换为 Bitmap(RGB)

* 函数

screen.cvimage\_to\_bitmap(image)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| image | opencv image | 否 | 要转换的Opencv Image |

* 返回值

[Bitmap](F:/cs01/cs01/cj2/1extracted_div_content.html) Python PIL 图像

* 示例

*# 将当前屏幕Bitmap 转换为 Openv Image = cv2.imread*  
from ascript.android import screen  
img= screen.bitmap\_to\_cvimage()  
print(img)

### 图像Bitmap[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图像bitmap" \o "Direct link to 图像Bitmap)

from android.graphics import Bitmap

在Android 中的内存图像对象

#### 宽度与高度[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "宽度与高度" \o "Direct link to 宽度与高度)

获取图像宽度

* 示例

from ascript.android import screen  
bitmap = screen.capture()  
print(bitmap.getWidth(),bitmap.getHeight())

### 颜色[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "颜色" \o "Direct link to 颜色)

from ascript.android.screen import Color

#### 创建一个颜色[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建一个颜色" \o "Direct link to 创建一个颜色)

创建一个颜色值,用于颜色值转换,和值的获取

* 函数

Color(color\_argb:int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| color\_argb | int | 是 | [图像所有色值](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E5%9B%BE%E5%83%8F%E6%89%80%E6%9C%89%E8%89%B2%E5%80%BC) 该方法返回结果中的Int值 (实际为ARGB的色值) |

* 示例

*# \_\_init\_\_.py 为初始化加载文件*  
  
from ascript.android import screen  
from ascript.android.screen import Color  
colors = screen.get\_colors()  
  
*# 获取第一个int颜色值*  
color\_int = colors[0]  
*# 转换为 Color对象,方便对颜色值进行转换*  
color = Color(color\_int)  
  
print(color.rgb)

#### 属性[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "属性" \o "Direct link to 属性)

| **属性** | **返回类型** | **说明** |
| --- | --- | --- |
| r | int | R通道 10进制 颜色值 |
| g | int | G通道 10进制 颜色值 |
| b | int | B通道 10进制 颜色值 |
| a | int | A通道(透明值) 10进制 颜色值 |
| rgb | str | 16进制 RGB颜色值 |
| argb | str | 16进制 ARGB颜色值 |

* 示例

from ascript.android import screen  
from ascript.android.screen import Color  
colors = screen.get\_colors()  
  
*# 获取第一个int颜色值*  
color\_int = colors[0]  
*# 转换为 Color对象,方便对颜色值进行转换*  
color = Color(color\_int)  
  
print(color.rgb) *#16进制RGB颜色值*  
print(color.argb) *#16进制ARGB颜色值 PS:A位透明值通道*  
print(color.r,color.g,color.b,color.a) *# 获取R通道,G通道,B通道,A通道 10进制颜色值*

# 多点找色

from ascript.android.screen import FindColors

从屏幕中找到 **符合颜色特征点**的 点位

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 找一个点[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找一个点" \o "Direct link to 找一个点)

找到 一个符合特征点的位置 **立刻返回**

* 函数

FindColors.find(colors:str,rect:list=None,space:int=5,ori:int=2,diff:float=0.9)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| colors | str | 是 | 颜色特征点, **通常用图色助手获取** |
| rect | list | 否 | 圈定屏幕范围 |
| space | int | 是 | 找色结果间距 默认:5像素 ,如果返回的结果,多个点位的像素值 在5像素内重合.则只保留一个 |
| ori | int | 否 | 找色方向 1-8 个方向 ,2(默认):左上角到右下角，横向开始找色 [详情见...](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E5%85%B3%E4%BA%8E%E6%89%BE%E8%89%B2%E6%96%B9%E5%90%91) |
| diff | float | 否 | 相似度 0-1 之间, 1为100%匹配 默认:0.9 |

* 返回值

[Point](F:/cs01/cs01/cj2/1extracted_div_content.html) 点位对象. 可通过 .x .y 获取坐标

*# 查找一个满足颜色特征的点*  
*# 导包*  
from ascript.android.screen import FindColors  
  
point = FindColors.find('634,2126,#05D395|675,2142,#05D395|889,2146,#05D395|939,2117,#05D395|868,2085,#05D395|652,2124,#05D294',rect=[82,2010,1005,2215])  
if point:  
 *# 打印 x坐标 和 y坐标*  
 print(point.x,point.y)

### 找到全部点[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找到全部点" \o "Direct link to 找到全部点)

找到 一个符合特征点的位置 **立刻返回**

* 函数

FindColors.find\_all(colors:str,rect:list=None,space:int=5,ori:int=2,diff:float=0.9)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| colors | str | 是 | 颜色特征点, **通常用图色助手获取** |
| rect | list | 否 | 圈定屏幕范围 |
| space | int | 是 | 找色结果间距 默认:5像素 ,如果返回的结果,多个点位的像素值 在5像素内重合.则只保留一个 |
| ori | int | 否 | 找色方向 1-8 个方向 ,2(默认):左上角到右下角，横向开始找色 [详情见...](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "%E5%85%B3%E4%BA%8E%E6%89%BE%E8%89%B2%E6%96%B9%E5%90%91) |
| diff | float | 否 | 相似度 0-1 之间, 1为100%匹配 默认:0.9 ,当图色工具中颜色带有偏色,则不会使用该值 |

* 返回值

[Point](F:/cs01/cs01/cj2/1extracted_div_content.html) 点位对象. 可通过 .x .y 获取坐标

* 示例

*# 查找所有满足颜色特征的点*  
*# 导包*  
from ascript.android.screen import FindColors  
  
points = FindColors.find\_all('634,2126,#05D395|675,2142,#05D395|889,2146,#05D395|939,2117,#05D395|868,2085,#05D395|652,2124,#05D294',rect=[82,2010,1005,2215])  
if points:  
 *# 打印 x坐标 和 y坐标*  
 for p in points:  
 print(p.x,p.y)

## 关于找色方向[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "关于找色方向" \o "Direct link to 关于找色方向)

找色方向是指:**找到颜色特征点的先后顺序**

如图:

* 1: 左上角到右下角，纵向开始找色，先找到 a 顶点。
* 2: 左上角到右下角，横向开始找色，先找到 b 顶点。
* 3: 右上角到左下角，横向开始找色，先找到 c 顶点。
* 依此类推

# 多点比色

from ascript.android.screen import CompareColors

对比当前屏幕图像 **固定点位颜色特征** 是否相同

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 比色[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "比色" \o "Direct link to 比色)

对比颜色

* 函数

CompareColors.compare(colors:str,diff:float=0.9)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| colors | str | 是 | 颜色特征点, **通常用图色助手获取** |
| diff | float | 否 | 颜色相似度 0-1 之间 默认0.9 |

* 返回值

[bool](F:/cs01/cs01/cj2/1extracted_div_content.html) True:比色成功,False:比色失败

* 示例

*# 比对颜色点,是否与屏幕中的颜色相同*  
*# 导包*  
from ascript.android.screen import CompareColors  
  
res = CompareColors.compare('339,342,#38D8A8|445,424,#FFFFFF|504,586,#FFFFFF')  
  
if res:  
 print('比色成功')  
else:  
 print('比色失败')

### 比色 直到成功[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "比色-直到成功" \o "Direct link to 比色 直到成功)

比对颜色,直到成功后返回结果 **会阻塞线程**

* 函数

CompareColors.compare\_until(colors:str,diff:float=0.9)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| colors | str | 是 | 颜色特征点, **通常用图色助手获取** |
| diff | float | 否 | 颜色相似度 0-1 之间 默认0.9 |

* 返回值

[bool](F:/cs01/cs01/cj2/1extracted_div_content.html) True:比色成功,False:比色失败

*# 比对颜色点,是否与屏幕中的颜色相同 ,如果不相同,直到相同才返回结果.期间会阻塞程序*  
*# 导包*  
from ascript.android.screen import CompareColors  
  
res = CompareColors.compare\_until('339,342,#38D8A8|445,424,#FFFFFF|504,586,#FFFFFF')  
  
if res:  
 print('比色成功')  
else:  
 print('比色失败')

# 找图

from ascript.android.screen import FindImages

从屏幕中找到 **局部图片** 的 位置 . **支持全分辨率**

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 找一个 混合算法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找一个-混合算法" \o "Direct link to 找一个 混合算法)

先使用 模版匹配**find\_template** ,如果找不到再使用全分辨率算法 **find\_sift** 查找

* 函数

FindImages.find(part\_img:str,rect:list=None,confidence:int=0.1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| part\_img | list,str | 是 | 局部图片名称或路径或图片路径列表 **当只填写图片名称时,将在res/img下找到该名称的图片** |
| rect | list | 否 | 圈定找图屏幕范围 |
| confidence | int | 否 | 图片结果的可信度0-1之间, 1为100%匹配,低于该可信度的结果将被过滤掉 默认:0.1 |

* 返回值

Python dict 字典

{  
 'rect': [915, 1761, 1076, 1899], *# 子图在屏幕中的范围*  
 'center\_x': 995, *# 在屏幕中的中心点坐标x*  
 'center\_y': 1830, *# 在屏幕中的中心点坐标y*  
 'confidence': 0.94925636649131775 *# 图片的相似度*  
}

* 示例

*# 在屏幕中找到局部图片的位置 find方式,先 template 后全分辨率*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/a.png") *# 这里替换为你的图片地址*  
res = FindImages.find(path)  
if res:  
 print("中心坐标:",res["center\_x"],res["center\_y"])  
 print("相似度:",res["confidence"])  
 print("在屏幕中的范围:",res["rect"])

*# 在屏幕指定范围内找到局部图片的位置 find方式,先 template 后全分辨率*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/a.png") *# 这里替换为你的图片地址*  
res = FindImages.find(path,rect =[ 59,932,374,1207])  
if res:  
 print("中心坐标:",res["center\_x"],res["center\_y"])  
 print("相似度:",res["confidence"])  
 print("在屏幕中的范围:",res["rect"])

from ascript.android.screen import FindImages  
*# 在全屏中找局部图片,过滤掉可信度低于0.8的*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/a.png") *# 这里替换为你的图片地址*  
res = FindImages.find(path,confidence=0.8)  
if res:  
 print("中心坐标:",res["center\_x"],res["center\_y"])  
 print("相似度:",res["confidence"])  
 print("在屏幕中的范围:",res["rect"])

### 找一个 模版匹配[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找一个-模版匹配" \o "Direct link to 找一个 模版匹配)

opencv中的模版匹配算法, **找色速度快**, **不支持全分辨率**

* 函数

FindImages.find\_template(part\_img:str,rect:list=None,confidence:int=0.1,rgb:bool=False)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| part\_img | list,str | 是 | 局部图片名称或路径或图片路径列表 **当只填写图片名称时,将在res/img下找到该名称的图片** |
| rect | list | 否 | 圈定找图屏幕范围 |
| confidence | int | 否 | 图片结果的可信度0-1之间, 1为100%匹配,低于该可信度的结果将被过滤掉 默认:0.1 |
| rgb | bool | 否 | True:使用原色图匹配 False(默认):使用灰度图匹配 **灰度图模式下,可以查找出相同图形,不同颜色的图案** |

* 返回值

Python dict 字典

{  
 'rect': [915, 1761, 1076, 1899], *# 子图在屏幕中的范围*  
 'center\_x': 995, *# 在屏幕中的中心点坐标x*  
 'center\_y': 1830, *# 在屏幕中的中心点坐标y*  
 'confidence': 0.94925636649131775 *# 图片的相似度*  
}

*# 在指定屏幕范围中找局部图片,过滤掉可信度低于0.8的 (不支持全分辨率)*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/a.png") *# 这里替换为你的图片地址*  
res = FindImages.find\_template(path,rect=[ 59,932,374,1207],confidence=0.8)  
if res:  
 print("中心坐标:",res["center\_x"],res["center\_y"])  
 print("相似度:",res["confidence"])  
 print("在屏幕中的范围:",res["rect"])

### 找一个 全分辨率查找[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找一个-全分辨率查找" \o "Direct link to 找一个 全分辨率查找)

全分辨率算法, **找色速度慢**, **可以在任意设备上找到图像特征匹配的图**

* 函数

FindImages.find\_sift(part\_img:str,rect:list=None,confidence:int=0.1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| part\_img | list,str | 是 | 局部图片名称或路径或图片路径列表 **当只填写图片名称时,将在res/img下找到该名称的图片** |
| rect | list | 否 | 圈定找图屏幕范围 |
| confidence | int | 否 | 图片结果的可信度0-1之间, 1为100%匹配,低于该可信度的结果将被过滤掉 默认:0.1 |

* 返回值

Python dict 字典

{  
 'rect': [915, 1761, 1076, 1899], *# 子图在屏幕中的范围*  
 'center\_x': 995, *# 在屏幕中的中心点坐标x*  
 'center\_y': 1830, *# 在屏幕中的中心点坐标y*  
 'confidence': 0.94925636649131775 *# 图片的相似度*  
}

* 示例

*# 全分辨率找 单图*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/a.png") *# 这里替换为你的图片地址*  
res = FindImages.find\_sift(path,confidence=0.8)  
if res:  
 print("中心坐标:",res["center\_x"],res["center\_y"])  
 print("相似度:",res["confidence"])  
 print("在屏幕中的范围:",res["rect"])

### 找所有 混合算法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找所有-混合算法" \o "Direct link to 找所有 混合算法)

先使用 模版匹配**find\_template** ,如果找不到再使用全分辨率算法 **find\_全分辨率** 查找

* 函数

FindImages.find\_all(part\_img:str,rect:list=None,confidence:int=0.1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| part\_img | list,str | 是 | 局部图片名称或路径或图片路径列表 **当只填写图片名称时,将在res/img下找到该名称的图片** |
| rect | list | 否 | 圈定找图屏幕范围 |
| confidence | int | 否 | 图片结果的可信度0-1之间, 1为100%匹配,低于该可信度的结果将被过滤掉 默认:0.1 |

* 返回值

Python 字典数组

[{  
 'rect': [479, 415, 597, 529], *# 子图在屏幕中的范围*  
 'center\_x': 538, *# 在屏幕中的中心点坐标x*  
 'center\_y': 472, *# 在屏幕中的中心点坐标y*  
 'confidence': 0.9975726008415222 *#图片相似度*  
}, {  
 'rect': [479, 983, 597, 1097],  
 'center\_x': 538,  
 'center\_y': 1040,  
 'confidence': 0.9972777962684631  
},...]

* 示例

*# 全分辨率找 所有图*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/2.png")  
res = FindImages.find\_all(path,confidence=0.8)  
if res:  
 *# 遍历所有 找到的结果*  
 for i in res:  
 print("中心坐标:",i["center\_x"],i["center\_y"])  
 print("相似度:",i["confidence"])  
 print("在屏幕中的范围:",i["rect"])

### 找所有 模版匹配[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找所有-模版匹配" \o "Direct link to 找所有 模版匹配)

opencv中的模版匹配算法, **找色速度快**, **不支持全分辨率**

* 函数

FindImages.find\_all\_template(part\_img:str,rect:list=None,confidence:int=0.1,rgb:bool=False)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| part\_img | list,str | 是 | 局部图片名称或路径或图片路径列表 **当只填写图片名称时,将在res/img下找到该名称的图片** |
| rect | list | 否 | 圈定找图屏幕范围 |
| confidence | int | 否 | 图片结果的可信度0-1之间, 1为100%匹配,低于该可信度的结果将被过滤掉 默认:0.1 |
| rgb | bool | 否 | True:使用原色图匹配 False(默认):使用灰度图匹配 **灰度图模式下,可以查找出相同图形,不同颜色的图案** |

* 返回值

Python 字典数组

[{  
 'rect': [479, 415, 597, 529], *# 子图在屏幕中的范围*  
 'center\_x': 538, *# 在屏幕中的中心点坐标x*  
 'center\_y': 472, *# 在屏幕中的中心点坐标y*  
 'confidence': 0.9975726008415222 *#图片相似度*  
}, {  
 'rect': [479, 983, 597, 1097],  
 'center\_x': 538,  
 'center\_y': 1040,  
 'confidence': 0.9972777962684631  
},...]

*# 找所有图 模版匹配 (不支持全分辨率)*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/2.png") *#这里替换成你的图片路径*  
res = FindImages.find\_all\_template(path,confidence=0.8)  
if res:  
 *# 遍历所有 找到的结果*  
 for i in res:  
 print("中心坐标:",i["center\_x"],i["center\_y"])  
 print("相似度:",i["confidence"])  
 print("在屏幕中的范围:",i["rect"])

### 找所有 全分辨率算法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "找所有-全分辨率算法" \o "Direct link to 找所有 全分辨率算法)

全分辨率算法, **找色速度慢**, **可在所有设备上找到图像特征匹配的图像**

* 函数

FindImages.find\_all\_sift(part\_img:str,rect:list=None,confidence:int=0.1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| part\_img | list,str | 是 | 局部图片名称或路径或图片路径列表 **当只填写图片名称时,将在res/img下找到该名称的图片** |
| rect | list | 否 | 圈定找图屏幕范围 |
| confidence | int | 否 | 图片结果的可信度0-1之间, 1为100%匹配,低于该可信度的结果将被过滤掉 默认:0.1 |

* 返回值

Python 字典数组

[{  
 'rect': [479, 415, 597, 529], *# 子图在屏幕中的范围*  
 'center\_x': 538, *# 在屏幕中的中心点坐标x*  
 'center\_y': 472, *# 在屏幕中的中心点坐标y*  
 'confidence': 0.9975726008415222 *#图片相似度*  
}, {  
 'rect': [479, 983, 597, 1097],  
 'center\_x': 538,  
 'center\_y': 1040,  
 'confidence': 0.9972777962684631  
},...]

* 示例

*# 找所有图 全分辨率算法 (全分辨率)*  
*# 导包*  
from ascript.android.screen import FindImages  
*# 导入上下文环境包,方便导入图片地址*  
from ascript.android.system import R  
path = R.res("/img/2.png")  
res = FindImages.find\_all\_sift(path,confidence=0.8)  
if res:  
 *# 遍历所有 找到的结果*  
 for i in res:  
 print("中心坐标:",i["center\_x"],i["center\_y"])  
 print("相似度:",i["confidence"])  
 print("在屏幕中的范围:",i["rect"])

# 文字识别

from ascript.android.screen import Ocr

从屏幕中 识别文字

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 谷歌文字识别[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "谷歌文字识别" \o "Direct link to 谷歌文字识别)

谷歌 MLKIT 引擎v2, 速度快,且准确度高. (源码版本:4.0)

* 函数

Ocr.mlkitocr\_v2(rect: list = None, pattern: str = None bitmap=None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| rect | list | 否 | 识别范围 |
| pattern | regx | 否 | 正则匹配 |
| bitmap | str | 否 | 要识别的图片,默认为当前屏幕,如需传入指定图片,请传入Bitmap格式 |

* 返回值

返回值为 OcrText 数组对象

谷歌OCR并不支持 confidence属性!

谷歌OCR,并没有可信度.只要识别到的都是准确的文本.

[  
 OcrText{  
 text='应用或小程序D...', *#识别到文字*  
 center\_x=345, *# 文字中心点 在屏幕中的X坐标*  
 center\_y=202, *# 文字中心点 在屏幕中的Y坐标*  
 rect=[345,202,644,242], *# 文字段落 在屏幕中的范围数组*  
 text\_box\_position=[] *# 每一个字的在屏幕中的范围*  
 },  
 OcrText{text='应用或小程序D...', center\_x=345, center\_y=202, rect=[345,202,644,242], text\_box\_position=[]}  
]

* 示例

*# 谷歌OCR 识别中文,并使用自动分割*  
from ascript.android.screen import Ocr  
res = Ocr.mlkitocr\_v2()  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect)

### 百度飞桨v2[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "百度飞桨v2" \o "Direct link to 百度飞桨v2)

百度飞桨v2 文字识别引擎 ,速度快,精度中等

* 函数

Ocr.paddleocr\_v2(  
 rect:list=None,  
 pattern:str=None,  
 confidence:int=0.1  
 max\_side\_len:int=1200,  
 precision:int=16,  
 bitmap=None,  
 file:str=None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| rect | list | 否 | 圈定屏幕中识别的范围 |
| pattern | str | 否 | 匹配的正则表达式 如保留字符串中包含IT的识别结果:.\*IT.\* |
| confidence | float | 否 | 识别可信度:默认0.1 |
| max\_side\_len | str | 否 | 如果指定最大边时,如1200,那么所有输入资源都会同比缩放最大边至1200px,再传入引擎识别 |
| bitmap | str | 否 | 从指定的图片中识别 **该Bitmap 可以从Screen.capture()获取** |
| file | str | 否 | 从指定的文件中识别 |

* 返回值

OcrText 对象,可通过 . 获取内部属性.

[  
 OcrText{  
 text='应用或小程序D...', *#识别到文字*  
 confidence = 0.8111246228218079, *# 可信度*  
 center\_x=345, *# 文字中心点 在屏幕中的X坐标*  
 center\_y=202, *# 文字中心点 在屏幕中的Y坐标*  
 rect=[345,202,644,242], *# 文字段落 在屏幕中的范围数组*  
 text\_box\_position=[] *# 每一个字的在屏幕中的范围*  
 },  
 OcrText{text='应用或小程序D...', center\_x=345, center\_y=202, rect=[345,202,644,242], text\_box\_position=[]}  
]

* 示例

*# 找到屏幕中的所有文字,保留结果中,包含程序的. 在识别的时候,图片最大边缩放至1600(增加识别速度)*  
*# 导包*  
from ascript.android.screen import Ocr  
res = Ocr().paddleocr\_v2(rect=[40,122,714,659],pattern=".\*程序.\*",max\_side\_len=1600)  
print(res)  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("可信度",r.confidence)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect) *# 这是一个int数组,可以这样取值 r.rect[0],r.rect[1],r.rect[2],r.rect[3]*

### 百度飞桨v3[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "百度飞桨v3" \o "Direct link to 百度飞桨v3)

部分设备不支持

百度飞桨v3 文字识别引擎 ,速度慢,精度高

* 函数

Ocr.paddleocr\_v3(  
 rect:list=None,  
 pattern:str=None,  
 confidence:int=0.1  
 max\_side\_len:int=1200,  
 precision:int=16,  
 bitmap=None,  
 file:str=None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| rect | list | 否 | 圈定屏幕中识别的范围 |
| pattern | str | 否 | 匹配的正则表达式 如保留字符串中包含IT的识别结果:.\*IT.\* |
| confidence | float | 否 | 识别可信度:默认0.1 |
| max\_side\_len | str | 否 | 如果指定最大边时,如1200,那么所有输入资源都会同比缩放最大边至1200px,再传入引擎识别 |
| bitmap | str | 否 | 从指定的图片中识别 **该Bitmap 可以从Screen.capture()获取** |
| file | str | 否 | 从指定的文件中识别 |

* 返回值

OcrText 对象,可通过 . 获取内部属性.

[  
 OcrText{  
 text='应用或小程序D...', *#识别到文字*  
 confidence = 0.8111246228218079, *# 可信度*  
 center\_x=345, *# 文字中心点 在屏幕中的X坐标*  
 center\_y=202, *# 文字中心点 在屏幕中的Y坐标*  
 rect=[345,202,644,242], *# 文字段落 在屏幕中的范围数组*  
 text\_box\_position=[] *# 每一个字的在屏幕中的范围*  
 },  
 OcrText{text='应用或小程序D...', center\_x=345, center\_y=202, rect=[345,202,644,242], text\_box\_position=[]}  
]

* 示例

*# 找到屏幕中的所有文字*  
*# 导包*  
from ascript.android.screen import Ocr  
res = Ocr().paddleocr\_v3()  
print(ots)  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("可信度",r.confidence)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect) *# 这是一个int数组,可以这样取值 r.rect[0],r.rect[1],r.rect[2],r.rect[3]*

### TesserAct 文字识别[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "tesseract-文字识别" \o "Direct link to TesserAct 文字识别)

TesserAct 是谷歌维护的文字识别库,针对数字,英文较友好,可自行训练字库. (源码版本:4.0)

* 函数

Ocr.tess(data\_file= Ocr.Tess\_CHI,   
 rect: list = None,  
 pattern:str=None,  
 split\_level= RIL\_AUTO,  
 white\_list:str=None,  
 black\_list:str=None ):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| data\_file | str | 否 | 字库,可传入字库文件路径,或  Ocr.Tess\_CHI:已训练的中文字库 Ocr.Tess\_EN:英文字库  [已训练模型库](https://github.com/tesseract-ocr/tessdata/tree/4.0.0" \t "F:/cs01/cs01/cj2/_blank) |
| rect | list | 否 | 识别范围 |
| pattern | regx | 否 | 正则匹配 |
| split\_level | int | 否 | 识别结果的,分割等级:  RIL\_AUTO:自动分割(默认)  RIL\_BLOCK:块分割  RIL\_PARA:页分割 RIL\_TEXTLINE:行分割  RIL\_WORD:单词分割  RIL\_SYMBOL:符号分割 |
| white\_list | str | 否 | 识别白名单 |
| black\_list | str | 否 | 识别黑名单 |

* 返回值

OcrText 对象,可通过 . 获取内部属性.

TesserAct并不支持 confidence属性,

因为4.0的TesserAct 并不是深度学习模型,而是一种识别算法. 因此并没有可信度属性.

[  
 OcrText{  
 text='应用或小程序D...', *#识别到文字*  
 center\_x=345, *# 文字中心点 在屏幕中的X坐标*  
 center\_y=202, *# 文字中心点 在屏幕中的Y坐标*  
 rect=[345,202,644,242], *# 文字段落 在屏幕中的范围数组*  
 text\_box\_position=[] *# 每一个字的在屏幕中的范围*  
 },  
 OcrText{text='应用或小程序D...', center\_x=345, center\_y=202, rect=[345,202,644,242], text\_box\_position=[]}  
]

* 示例

*# TesserAct 识别中文,并使用自动分割*  
from ascript.android.screen import Ocr  
res = Ocr.tess()  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("可信度",r.confidence)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect) *# 这是一个int数组,可以这样取值 r.rect[0],r.rect[1],r.rect[2],r.rect[3]*

*# TesserAct 识别数字*  
from ascript.android.screen import Ocr  
res = Ocr.tess(data\_file=Ocr.Tess\_ENG,white\_list='1234567890')  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("可信度",r.confidence)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect) *# 这是一个int数组,可以这样取值 r.rect[0],r.rect[1],r.rect[2],r.rect[3]*

*# TesserAct 识别特定范围内的数字*  
from ascript.android.screen import Ocr  
res = Ocr.tess(data\_file=Ocr.Tess\_ENG,rect =[170,482,593,549], white\_list='1234567890')  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("可信度",r.confidence)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect) *# 这是一个int数组,可以这样取值 r.rect[0],r.rect[1],r.rect[2],r.rect[3]*

*# 指定字库文件,进行识别.(通常指定,自己训练的字库文件地址)*  
from ascript.android.screen import Ocr  
from ascript.android.system import R  
  
font\_file = R.sd("airscrpt/Ocr/tessdata/eng.traineddata")  
res = Ocr.tess(data\_file=font\_file)  
if res:  
 *#循环打印识别到的每一个段落*  
 for r in res:  
 print("文字:",r.text)  
 print("可信度",r.confidence)  
 print("文字中心坐标:",r.center\_x,r.center\_y)  
 print("文字范围:",r.rect) *# 这是一个int数组,可以这样取值 r.rect[0],r.rect[1],r.rect[2],r.rect[3]*

### 自定义图片字库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "自定义图片字库" \o "Direct link to 自定义图片字库)

自定义图片字库,**用小图构建字库** 请见 [CvFontLib](F:/cs01/cs01/cj2/docs/android/api/screen/cvfont)

### 点阵字库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "点阵字库" \o "Direct link to 点阵字库)

对特殊图像,做点阵字库. 从而进行识别,该方法不支持全分辨率

**点阵字库适合 范围检索,不适合全屏幕检索.**

#### 查找文字[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "查找文字" \o "Direct link to 查找文字)

给定字库,从字库中识别文字

* 函数

Ocr.matrix(font\_lib: str, rect: list = None, region=0.9):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| font\_lib | str | 是 | 字库文件 |
| rect | list | 否 | 识别范围 |
| region | float | 否 | 识别精度 0-1之间, 默认0.9=90%精度 |

* 返回值

返回值为 识别到的文字, 类型为str

from ascript.android.system import R  
from ascript.android.screen import OcrX  
res = OcrX.find\_all(R.res("font.t"),rect=[90,529,343,612])  
print(res[0])

#### 如何制作字库?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "如何制作字库" \o "Direct link to 如何制作字库?)

制作字库用到了**图色工具中的点阵识字**

1. 打开图色助手并点击点阵字库
2. 扩选要做的字
3. 按住键盘1-9取色要做的字,直到右侧出现字的二值图像
4. 选择要存放字库的工程和文件名称(文件名称默认font.t,可以更改) 此文件存放在 “工程/res/”目录下.会自动创建该文件. **制作多个文字会自动追加至该文件末尾**
5. 填写文字代号
6. 点击保存字体到字库,就完成了一个文字字库的制作

* 具体操作步骤如下图

# 检测

from ascript.android.screen import YoLov5

一种目标检测深度学习算法 ,通过对 图片中目标的标注,训练.得到模型,下载运行模型从而检测目标

应用场景

对于不同手机,不同分辨率情况下的目标检测. 能达到很好的效果.

例如 “微信跳一跳”

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### YoLov5[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "yolov5" \o "Direct link to YoLov5)

from ascript.android.screen import YoLov5

深度学习算法 ,通过对图片中目标的标注,训练.得到模型

#### 创建一个YoLov5[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建一个yolov5" \o "Direct link to 创建一个YoLov5)

* 函数

YoLov5(model\_name:str=None,path:str=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| model\_name | bool | 否 | [线上模型库](https://card.nspirit.cn/mlist.html" \t "F:/cs01/cs01/cj2/_blank)中的模型名称,格式: ("模型名称:模型版本号") 填写后,将自动下载线上模型,如缓存中存在,则从本地加载. |
| path | bool | 否 | 本地模型路径 |

* 返回值

[Yolov5](F:/cs01/cs01/cj2/1extracted_div_content.html) Yolov5 对象 可调用其find\_all 方法识别

* 示例

*# 创建Yolov5 并加载模型文件*  
from ascript.android.screen import YoLov5  
*# 从模型库中加载*  
yolo = YoLov5("微信跳一跳:1.5")

#### 检测所有目标[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "检测所有目标" \o "Direct link to 检测所有目标)

* 函数

YoLov5(model\_name:str=None,path:str=None).find\_all(rect=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| rect | [] | 否 | 屏幕范围:全屏:默认  rect = [100,100,400,400] 给定一个范围 |

* 返回值

识别目标的结果对象列表,包含的属性有:

.x:坐标x .y:坐标y .w:目标宽度 .h:目标高度 .lable:目标标签 .prob:识别可信度

* 示例

*# 创建Yolov5 并加载模型文件*  
from ascript.android.screen import YoLov5  
*# 从模型库中加载*  
yolo = YoLov5("微信跳一跳:1.5")  
  
while True:  
 res = yolo.find\_all()  
 for r in res:  
 print(r.label) *# 目标名称*  
 print(r.prob) *# 可信度*  
 print(r.x) *# 坐标x*  
 print(r.y) *# 坐标y*  
 print(r.w) *# 目标宽度*  
 print(r.h) *# 目标高度*

* 示例

*# 区域识别*  
from ascript.android.screen import YoLov5  
*# 从模型库中加载*  
yolo = YoLov5("微信跳一跳:1.5")  
  
while True:  
 ts = yolo.find\_all(rect=[7,960,1080,1868])  
 print(ts)

# 模型标注,训练,检测

yolov5 是一个优秀的开源目标检测模型.

我们在aslib中集成了它, 版本为 yolov5s-3.0

模型训练分为两个步骤:

* 1. 数据集标注
  2. 训练

接下来,我们通过文档的形式,详细来了解如何完成这两个步骤.

### 数据集格式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "数据集格式" \o "Direct link to 数据集格式)

如果您使用AS提供的标注工具,标注后将自动生成如下格式的数据包.

数据集格式必须如下

* 如果您想通过AS服务器 或者 AS提供的镜像 训练模型

那么数据集必须包含以下3个部分(images,labels,tag.txt)

* images 用于存放图片

> images下图片格式没有要求, jpg,jpeg,png 都可以  
 > 文件名称:无要求

* labels 用于存放 图片对应的标签,格式如下:

0 0.30974264705882354 0.52625 0.07169117647058823 0.09  
1 0.7256433823529411 0.453125 0.4163602941176471 0.11291666666666667

* tags 用于存放标签名称列表

nc: 5  
names: ["小人","方块","圆形","唱片机","杯子"]

如图:

### 标注[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "标注" \o "Direct link to 标注)

打开WEB编辑器 - 工具栏 - yolo目标检测 -标注训练

#### 创建标注集[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建标注集" \o "Direct link to 创建标注集)

#### 截屏/上传存入标注库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "截屏上传存入标注库" \o "Direct link to 截屏/上传存入标注库)

#### 打标签[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "打标签" \o "Direct link to 打标签)

理论来讲, 在数据集中图片越多,训练的结果识别效果越好.

因此我们尽可能的要在不同的图片中多打标签.

同一个目标,用相同的标签名称即可.

#### 疯狂打标签[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "疯狂打标签" \o "Direct link to 疯狂打标签)

图片越多越好,搞多多滴

#### 导出标注集[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导出标注集" \o "Direct link to 导出标注集)

标注工具 - 工具栏 - 导出并训练

按照这个步骤操作,你会得到 一个 cnp文件(数据集的压缩包)

### 训练[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "训练" \o "Direct link to 训练)

训练分为两种模式

AirScript 提供的云服务训练

自建服务器训练

#### AS服务器训练[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "as服务器训练" \o "Direct link to AS服务器训练)

前往 [开发者后台](http://py.airscript.cn/admin/model/list/create" \t "F:/cs01/cs01/cj2/_blank) 新增模型训练

填写信息,上传cnp数据集,点击提交.等待训练结果.

等待训练成功,即可加载使用了(50图左右,需训练30分钟左右)

#### 训练完成后加载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "训练完成后加载" \o "Direct link to 训练完成后加载)

from ascript.android.screen import YoLov5  
*# 注意,这里的 ‘微信跳一跳’就是训练的模型名称,1.5是模型的版本(需替换成你自己的模型与版本)*  
*# 创建Yolov5 并加载模型文件*  
yolo = YoLov5("微信跳一跳:1.5")  
  
while True:  
 res = yolo.find\_all()  
 for r in res:  
 print(r.label) *# 目标名称*  
 print(r.prob) *# 可信度*  
 print(r.x) *# 坐标x*  
 print(r.y) *# 坐标y*  
 print(r.w) *# 目标宽度*  
 print(r.h) *# 目标高度*

### 大数据集训练(自建服务器)[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "大数据集训练自建服务器" \o "Direct link to 大数据集训练(自建服务器))

大数据集,一般只1000张图以上的数据集.

这时候我AS的服务器训练会比较慢,各位开发者可以自行搭建服务器.

AS 提供 阿里云训练镜像给各位开发者.

1.获取镜像文件

"您需要先获得阿里云的账号ID,然后联系我们的客服企业微信,进行分享."

客服企业微信

1. 根据服务器镜像,创建 实例.
2. 模型训练服务器使用前准备

*#从宿主机进入到模型训练容器*  
[root@test ~]*# docker exec -ite8d4e63b8c6d /bin/bash*  
  
*#启动cron服务*  
[root@e8d4e63b8c6d:/usr/src/app*# /etc/init.d/cron start*  
\* Starting periodic command scheduler cron  
  
*#返回宿主机*  
[root@e8d4e63b8c6d:/usr/src/app*# exit*  
 exit  
  
*#从宿主机进入到模型转换容器*  
[root@test ~]*# docker exec -it 618381dfd1a9 /bin/bash*  
  
*#启动cron服务*  
[root@618381dfd1a9:/usr/src/app*# /etc/init.d/cron start*  
 \* Starting periodic command scheduler cron  
  
*#返回宿主机*  
[root@618381dfd1a9:/usr/src/app*# exit*  
exit  
[root@test ~]*#*

1. 进入管理后台,开始上传训练

* [http://你服务域名](http://%E4%BD%A0%E6%9C%8D%E5%8A%A1%E5%9F%9F%E5%90%8D" \t "F:/cs01/cs01/cj2/_blank)
* 账号： test
* 密码： test

# 检测

yolo v8集成在插件中,因此我们要**加载插件来完成检测**

一种目标检测深度学习算法 ,通过对 图片中目标的标注,训练.得到模型,下载运行模型从而检测目标

应用场景

对于不同手机,不同分辨率情况下的目标检测. 能达到很好的效果.

例如 微信跳一跳检测 , 多分辨率兼容的图像检测 , 物品检测, 带口罩检测 等等...

*# 加载yolov8插件*  
from ascript.android import plug  
plug.load("Yolov8Ncnn:1.3") *# 这里请更换最新版本*  
import Yolov8Ncnn

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 初始化模型[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "初始化模型" \o "Direct link to 初始化模型)

加载模型,并初始化

* 函数

Yolov8Ncnn.load(param\_path:str,bin\_path:str,nc:int,use\_gpu:bool=False)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| param\_path | bool | 是 | ncnn模型的param文件的绝对路径 |
| bin\_path | bool | 是 | ncnn模型的bin文件绝对路径 |
| nc | bool | 是 | 模型的标签数量,可在标签集data.yarm中看到 |
| use\_gpu | bool | 否 | 是否启动gpu，默认用cpu，速度相差不多~~~ |

* 返回值

[bool](F:/cs01/cs01/cj2/1extracted_div_content.html) 是否初始化成功

* 示例

from ascript.android import plug  
from ascript.android.system import R  
plug.load("Yolov8Ncnn:1.3")  
import Yolov8Ncnn as yolo  
  
*# 这里的路径换成你自己的， nc为你自己模型的标签数量*  
is\_init = yolo.load(R.res("custom.param"),R.res("custom.bin"),5)  
if is\_init:  
 print("模型初始化成功")

### 检测目标[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "检测目标" \o "Direct link to 检测目标)

* 函数

Yolov8Ncnn.detect(bitmap=None,target\_size:int=640,threshold=0.4,nms\_threshold=0.5)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap | androidBitmap | 否 | bitmap：可以不传，不传的话就用当前屏幕图像检测 |
| target\_size | int | 否 | 图像进入后缩放的检测大小,模型训练一般都用640,因此这里的640一般不要改 |
| threshold | float | 否 | 置信度,默认0.4 小于这个值的会被过滤掉 |
| nms\_threshold | float | 否 | 重叠阈值，一般是0.5不需要更改 |

* 返回值

数组[]:

[  
 {'class\_id': 1, 'confidence': 0.9673174023628235, 'rect': [551, 948, 1008, 1214]},  
 {'class\_id': 0, 'confidence': 0.9178692102432251, 'rect': [296, 1151, 370, 1364]}  
]

class\_id: 标签序号,可通过标签数组拿到对应的标签名称

confidence: 置信度

rect: 位置信息 [左,上,右,下] 4个点的坐标

示例

from ascript.android import plug  
from ascript.android.system import R  
plug.load("Yolov8Ncnn:1.3")  
import Yolov8Ncnn as yolo  
  
*# 在屏幕上绘制目标*  
yolo.draw(True,300)  
  
*# 这里的路径换成你自己的， nc为你自己模型的标签数量*  
is\_init = yolo.load(R.res("tyt.param"),R.res("tyt.bin"),5)  
  
if is\_init:  
 print("模型初始化成功")  
 res = yolo.detect()  
 for r in res:  
 print("标签ID:",r["class\_id"])  
 print("置信度:",r["confidence"])  
 print("目标坐标范围:",r["rect"])

### 绘制[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "绘制" \o "Direct link to 绘制)

检测到结果后,在屏幕上绘制目标框

* 函数

Yolov8Ncnn.draw(is\_draw:bool=True,draw\_time:int=1000)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| is\_draw | bool | 否 | 是否绘制目标,默认True |
| draw\_time | int | 否 | 绘制目标后,多久消失,默认1000毫秒 |

* 示例

from ascript.android import plug  
from ascript.android.system import R  
plug.load("Yolov8Ncnn:1.3")  
import Yolov8Ncnn as yolo  
  
yolo.draw(True,300)

### 释放模型[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "释放模型" \o "Direct link to 释放模型)

* 函数

Yolov8Ncnn.free()

# 模型标注,训练,检测

yolov8 目前支持本地训练,整个流程并不难完成.

模型训练分为两个步骤:

* 1. 数据集标注
  2. 训练

接下来,我们通过文档的形式,详细来了解如何完成这两个步骤.

### 标注[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "标注" \o "Direct link to 标注)

打开WEB编辑器 - 工具栏 - yolo目标检测 -标注训练

#### 创建标注集[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建标注集" \o "Direct link to 创建标注集)

#### 截屏/上传存入标注库[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "截屏上传存入标注库" \o "Direct link to 截屏/上传存入标注库)

#### 打标签[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "打标签" \o "Direct link to 打标签)

理论来讲, 在数据集中图片越多,训练的结果识别效果越好.

因此我们尽可能的要在不同的图片中多打标签.

同一个目标,用相同的标签名称即可.

#### 疯狂打标签[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "疯狂打标签" \o "Direct link to 疯狂打标签)

图片越多越好,搞多多滴

#### 导出标注集[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导出标注集" \o "Direct link to 导出标注集)

标注工具 - 工具栏 - 导出并训练 - 导出本地标签集

按照这个步骤操作,你会得到 一个 zip文件(数据集的压缩包)

### 本地训练[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "本地训练" \o "Direct link to 本地训练)

本地训练需要Python环境.

1.创建一个新的python环境,一般使用conda 官网地址: [https://www.anaconda.com](https://www.anaconda.com" \t "F:/cs01/cs01/cj2/_blank)

2.安装pytorch 官网地址：[https://pytorch.org/](https://pytorch.org/" \t "F:/cs01/cs01/cj2/_blank) 可在下方生成pip 进行安装

3.安装yolo库 :pip install ultralytics==8.3.31

4.安装pycharm 并用pycharm打开 已解压的数据集文件夹 ,官方地址:[https://www.jetbrains.com/pycharm/download/](https://www.jetbrains.com/pycharm/download/" \t "F:/cs01/cs01/cj2/_blank)

5.配置Pycharm的python环境为刚才我们创建的conda环境

6.打开train.py并点击运行开始训练

7.训练完成后，根据输出路径,把 model.ncnn.bin 和 model.ncnn.param 放置AScript工程下，加载即可 nc参数可查看data.yarm

[跳一跳训练模型案例](https://airscript.oss-cn-hangzhou.aliyuncs.com/res/%E8%B5%84%E6%BA%90/%E5%BE%AE%E4%BF%A1%E8%B7%B3%E4%B8%80%E8%B7%B3%E6%95%B0%E6%8D%AE%E9%9B%86%E5%92%8C%E6%A8%A1%E5%9E%8B.zip" \t "F:/cs01/cs01/cj2/_blank)

#### 视频教程[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "视频教程" \o "Direct link to 视频教程)

# 条码识别

from ascript.android.screen import CodeScanner

可识别 条形码, 二维码 等.

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 识别[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "识别" \o "Direct link to 识别)

找到 屏幕或文件中的二维码,并返回识别数据

* 函数

CodeScanner.scan(rect: list = None, bitmap=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| rect | list | 否 | 圈定要是识别的屏幕,文件范围 |
| bitmap | Bitmap | 否 | 默认从屏幕识别. 如需传入图片文件,需要使用 screen.file\_to\_bitmap(path:str,sampleSize:int=1) 转换为bitmap传入 |

* 返回值

python 数组, 数组中包含屏幕中所有码信息

[{'format': 256, 'type': 8, 'value': 'https://u.wechat.com/MDSoEI5PqQHbPfp8NEbXtdc?s=2', 'rect': [194, 717, 888, 1411]}]

* 示例

*# 识别屏幕中的二维码*  
from ascript.android.screen import CodeScanner  
res = CodeScanner.scan()  
  
if res:  
 *#循环打印所有码信息*  
 for code in res:  
 print("码类型",code["type"])  
 print("码值",code["value"])  
 print("码范围",code["rect"])

# 边缘检测

from ascript.android.screen.gp\_tasks import Canny

通过梯度检测图片中的边缘,通过x,y,宽,高限制得到特征区域

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

Canny(low\_threshold: int = 50, high\_threshold: int = 150, mode: int = RETR\_EXTERNAL,  
 draw: int = DRAW\_ALL, desc: bool = False,  
 left\_range: list = None, top\_range: list = None,  
 width\_range: list = None, height\_range: list = None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| low\_threshold | int | 否 | 最小梯度差 |
| high\_threshold | int | 否 | 最大梯度差 |
| mode | int | 否 | 检测模式,默认所有轮廓 |
| draw | int | 否 | 绘制模式  Canny.DRAW\_NONE:在图片上不进行绘制  Canny.DRAW\_ALL:在图片上绘制所有轮廓 Canny.DRAW\_RECT:在图片上绘制轮廓矩形范围  Canny.DRAW\_RECT\_FILL:在图片上绘制轮廓矩形,并将内部填白 |
| desc | bool | 否 | 是否在图片上绘制轮廓信息,如轮廓坐标,长度 |
| left\_range | list | 否 | 坐标x过滤,[minx,maxx] 如:[100,200] 保留所有轮廓坐标x 在100-200 之间轮廓 |
| top\_range | list | 否 | 坐标y过滤,[minx,maxx] 如:[100,200] 保留所有轮廓坐标y 在100-200 之间轮廓 |
| width\_range | list | 否 | 宽度过滤,[minx,maxx] 如:[100,200] 保留所有轮廓宽度 在100-200 之间轮廓 |
| height\_range | list | 否 | 高度过滤,[minx,maxx] 如:[100,200] 保留所有轮廓高度 在100-200 之间轮廓 |

* 示例

*# 检测所有轮廓,并绘制矩形范围*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import GaussianBlur  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Canny(draw=Canny.DRAW\_RECT))   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)  
  
*#打印所有轮廓信息*  
print(res.data)

*# 检测所有轮廓,绘制矩形范围,并保留轮廓宽度在100-500之间的所有*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import GaussianBlur  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Canny(draw=Canny.DRAW\_RECT,width\_range=[100,500]))   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)  
  
*#打印所有轮廓信息*  
print(res.data)

# 图像预处理

对图像/屏幕图像 进行预处理

预处理: 在识别图片,或检测图片之前 先对图片进行一些处理,让图像特征更明确.

## 裁剪[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "裁剪" \o "Direct link to 裁剪)

获取区域范围截图

from ascript.android.screen.gp\_tasks import Crop  
Crop(left: int, top: int, right: int, bottom: int)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| left | int | 是 | 范围左上角顶点x |
| top | int | 是 | 范围左上角顶点y |
| right | int | 是 | 范围右下角顶点x |
| bottom | int | 是 | 范围右下角顶点y |

* 示例

*#截取 当前屏幕的[100,100,500,500]的范围*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Crop  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Crop(100,100,500,500))   
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 灰度图[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "灰度图" \o "Direct link to 灰度图)

RGB通道转换为单通道灰白图像

* 灰度图: 只有灰白颜色的图片.

from ascript.android.screen.gp\_tasks import GrayImage  
  
GrayImage()

参数 无

示例

*# 将当前屏幕转换为灰度图*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Crop  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(GrayImage())   
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 二值化[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "二值化" \o "Direct link to 二值化)

又称阈值处理,先灰度图,将超过阈值图像区域填黑,其余填白(可反转黑白)

from ascript.android.screen.gp\_tasks import Threshold  
  
Threshold(threshold: int, inv: bool = False)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| threshold | int | 是 | 阈值,超过阈值的至为黑色,小于阈值的至为白色, 黑白可反转 |
| inv | bool | 否 | 是否黑白反转,默认不反转. |

* 示例

*# 将当前屏幕二值化,阈值为150*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Threshold  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Threshold(150))   
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

*# 将当前屏幕二值化,并黑白反转,阈值为150*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Threshold  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Threshold(150,True))   
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 腐蚀[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "腐蚀" \o "Direct link to 腐蚀)

消除图像中的边界或小细节，从而使物体在图像中显得更加清晰

from ascript.android.screen.gp\_tasks import Erode  
  
Erode(kernel\_size: int = 5, iterations: int = 1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| kernel\_size | int | 否 | 腐蚀内核大小,默认为5\*5 |
| iterations | int | 否 | 腐蚀次数,默认1 |

* 示例

*# 将当前图像进行腐蚀*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Erode  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Erode())   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

*# 将当前图像进行腐蚀,内核大小设置为9\*9,次数为2*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Erode  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Erode(9,2))   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 膨胀[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "膨胀" \o "Direct link to 膨胀)

消除黑色噪点,填补小的孔洞或断裂的区域,强调或扩展图像中的高亮区域

from ascript.android.screen.gp\_tasks import Dilate  
  
Dilate(kernel\_size: int = 5, iterations: int = 1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| kernel\_size | int | 否 | 内核大小,默认为5\*5 |
| iterations | int | 否 | 膨胀次数,默认1 |

* 示例

*# 将当前图像进行膨胀*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Dilate  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Dilate())   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

*# 将当前图像进行膨胀,内核大小设置为11,次数设置为2*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import Dilate  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(Dilate(11,2))   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 开运算[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "开运算" \o "Direct link to 开运算)

腐蚀+膨胀,去除图像中的小细节、噪声或边缘

from ascript.android.screen.gp\_tasks import MorphOpen  
  
MorphOpen(kernel\_size: int = 5, iterations: int = 1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| kernel\_size | int | 否 | 内核大小,默认为5\*5 |
| iterations | int | 否 | 执行次数,默认1 |

* 示例

*# 将当前图像进行开运算*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import MorphOpen  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(MorphOpen())   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

*# 将当前图像进行开运算,内核大小设置为11,次数设置为2*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import MorphOpen  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(MorphOpen(11,2))   
*# gp.add(....) 还可以增加更多的处理*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 闭运算[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "闭运算" \o "Direct link to 闭运算)

膨胀+腐蚀,填充物体内部的空洞，连接断开的物体，以及平滑物体的边缘

from ascript.android.screen.gp\_tasks import MorphClose  
  
MorphClose(kernel\_size: int = 5, iterations: int = 1)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| kernel\_size | int | 否 | 内核大小,默认为5\*5 |
| iterations | int | 否 | 执行次数,默认1 |

* 示例

*# 将当前图像进行闭运算*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import MorphClose  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(MorphClose())   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

*# 将当前图像进行闭运算,内核大小设置为11,次数设置为2*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import MorphClose  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(MorphClose(11,2))   
*# gp.add(....) 还可以增加更多的处理*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

## 高斯模糊[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "高斯模糊" \o "Direct link to 高斯模糊)

去除图像噪声,降低细节层次,平滑处理,边缘增强,图像融合,纹理削弱 等功能

from ascript.android.screen.gp\_tasks import GaussianBlur  
  
GaussianBlur(ksize: int = 5, sigmaX: int = 0)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| ksize | int | 否 | 内核大小,默认为5\*5 |
| sigmaX | int | 否 | 高斯模糊差 |

* 示例

*# 将当前图像进行高斯模糊*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import GaussianBlur  
from ascript.android.ui import ImageWindow  
gp = GPStack()  
gp.add(GaussianBlur())   
*# gp.add(....) 还可以增加更多的插件*  
res = gp.run()  
  
*# 展示处理后的图像*  
ImageWindow.show(res.image)

# 图色插件

列表中的插件,均为开发者上传.

其中部分插件可能包含收费服务,使用前务必了解详细.

## 图形插件列表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图形插件列表" \o "Direct link to 图形插件列表)

Loading...

# 图形识别栈

from ascript.android.screen.gp import GPStack

一个图形识别栈代指一次完整的图形处理,直到得到结果

**图形识别栈中,可以加入** 图片预处理**,**图色识别**,**图色插件,

开发者可根据需求添加不同的图形识别插件.

图形识别栈,就是用来调度图形插件运行而存在的.

## 创建图形识别栈[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建图形识别栈" \o "Direct link to 创建图形识别栈)

创建一个图形识别栈

* 函数

GPStack(cv\_img=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| cv\_img | numpy.array | 否 | 要处理的opencv img,默认为当前屏幕截图. 开发者也可通过cv2.imread 方法将文件读取为cv图片进行处理. |

* 返回值

**GpStack** 图形栈对象

## 添加图形识别插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "添加图形识别插件" \o "Direct link to 添加图形识别插件)

向图形识别栈中,加一个图形识别插件

* 函数

GPStack().add(gp\_task)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| gp\_task | Obj | 是 | 图形识别插件对象,该对象所属类必须继承于 GPTask |

* 示例

*#创建一个图形识别栈,并向其中加入一个灰度图处理插件.*  
from ascript.android.screen.gp import GPStack  
from ascript.android.screen.gp\_tasks import GrayImage  
gp = GPStack()  
gp.add(GrayImage())   
res = gp.run()  
print(res.data)

## 执行图像处理栈[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "执行图像处理栈" \o "Direct link to 执行图像处理栈)

运行图形识别栈,并得到返回图形或处理数据

* 函数

GPStack().run()

* 返回值

*#返回值是一个 Rssult对象*  
from ascript.android.screen.gp import Result

* Result 中包含的字段

| **返回值字段** | **类型** | **说明** |
| --- | --- | --- |
| image | numpy.array | 处理后的 opencv image图像,如果你需要展示图片,或对图像进一步处理,可以操作此对象 |
| offset\_x ,offset\_y | int | 图像插件处理后,相对于源图的偏移量 x, 比如 Crop插件裁剪后,则返回裁剪后的图片距源图的偏移量x,y |
| data | obj | 插件处理图像后,得到的处理结果 |

# OpenCv

import cv2

opencv 是一个成熟的图像处理与算法库

AS中内置了版本 4.1.2.30

## OpenCv 文档与教程[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "opencv-文档与教程" \o "Direct link to OpenCv 文档与教程)

* [30分钟入门opencv图像处理](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "https://www.bilibili.com/video/BV1BT4y1Z7WS/)
* [OpenCv 中文文档](http://www.woshicver.com" \t "F:/cs01/cs01/cj2/_blank)

## 截屏为OpenCv Image[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "截屏为opencv-image" \o "Direct link to 截屏为OpenCv Image)

在AS中可以直接将屏幕截图 转换为CV图像

这里的CV图像 实际上是一个 numpy.ndarray 类型.

在python opencv 中,图像都用这样一个类型去传递处理.

我们使用方法 [screen.bitmap\_to\_cvimage()](F:/cs01/cs01/cj2/docs/android/api/screen/" \l "bitmap-%E8%BD%AC-opencv-image) 可以将bitmap 转换为 numpy.ndarray

* 示例

*# 将内存截图转换为 OpencvImage*  
import cv2  
from ascript.android.ui import ImageWindow  
from ascript.android import screen  
  
*# 将内存截图Bitmap转换为cv img*  
img = screen.bitmap\_to\_cvimage()  
  
*# 展示图片*  
ImageWindow.show(img)

# 网页窗口

from ascript.android.ui import WebWindow

制作用户交互界面

此网页窗口, 使用 **WebView** 展示 Html页面 ,需要开发者掌握一定 html知识.

开启窗口后,使用 JavaScript 可以和Python语言交互与通信.

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 创建Web窗口[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建web窗口" \o "Direct link to 创建Web窗口)

创建一个WebUI交互窗口,窗口依赖Html

* 函数

WebWindow(self,html:str,tunnel=None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| html | str | 是 | html布局路径,请使用 ‘R.res’ 引用资源,也可以传入一个http地址 |
| tunnel | fun | 否 | html 向 python 传递参数的通道,该函数需存在两个形式参数 key,value,html会传递键值对参数过来 |

* 返回值

[WebWindow](F:/cs01/cs01/cj2/1extracted_div_content.html) 窗口对象

* 示例

*#展示一个简单的Html页面*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
def tunnel(k,v):  
 print(k)  
 print(v)  
  
*# 构建一个WebWindow 显示‘/res/ui/a.html’ 通信通道为tunnel 函数*  
w = WebWindow(R.ui('a.html'),tunnel)  
w.show()

### 窗口模式[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "窗口模式" \o "Direct link to 窗口模式)

可以改变窗口的模式

* 函数

WebWindow(...).mode(mode:int=0)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| mode | str | 是 | 窗口的模式 m=0 默认,窗口内和窗口外都可以获取点击触摸事件,但窗口外不可以调出键盘 m=1 只有窗口内的点击触摸事件有效  m=2 窗口内的所有点击触摸事件无效,并透传到窗口下 m=3窗口内外都可以点击触摸,但窗口内不可以调起键盘 m = -1 窗口载入进activity中.窗口内长按事件触发为系统原生方式 |

* 返回值

[WebWindow](F:/cs01/cs01/cj2/1extracted_div_content.html) 窗口对象

* 示例

*# 显示一个 不可触摸的悬浮窗*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.mode(2)  
w.show()

### 消息通道[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "消息通道" \o "Direct link to 消息通道)

设置消息通道,可以接收到js中传递的消息

* 函数

WebWindow(...).tunner(pyfun)

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| pyfun | function | 必填 | 必须包含两个形式参数 k,v |

* 示例

*# 设置和html交互的消息通道*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
def tunner(k,v):  
 print(k,v)  
  
w = WebWindow(R.ui("loger.html"))  
w.tunner(tunner) *# 在这里设置消息通道*  
w.show()

### 尺寸[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "尺寸" \o "Direct link to 尺寸)

设置窗口 宽 或 高

* 函数

WebWindow(...).size(self,width=None,height=None)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| width | number 或 string | 必填 | 当值为number类型时 width=300 设置窗口宽度为300px  当值为string类型时  .width="80vw" 设置宽度为屏幕宽度的 80%  width ="80vh" 设置宽度为屏幕高度的 80% |
| height | number 或 string | 必填 | 当值为number类型时 height =300 设置窗口高度为300px  当值为string类型时  height="80vw" 设置高度为屏幕宽度的 80%  width ="80vh" 设置高度为屏幕高度的 80% |

* 示例

*# 根据像素值 设置窗口宽高*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.size(500,300)  
w.show()

*# 根据百分比 设置窗口宽高*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.size('50vw','50vh')  
w.show()

### 背景[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "背景" \o "Direct link to 背景)

设置UI背景色,默认为白色

* 函数

WebWindow(...).background(color:str)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| color | string | 必填 | argb 颜色值  透明:#00000000  白色:#FFFFFF  黑色:#000000 |

* 示例

*# 设置背景色透明*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.background("#00000000")  
w.show()

### 拖拽[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "拖拽" \o "Direct link to 拖拽)

UI窗口是否可被用户拖拽,默认为False

* 函数

WebWindow(...).drag(is\_drag:bool=False)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| is\_drag | boolean | 必填 | 系统默认为False 设置为True 后,在UI顶部出现拖动条,和改变尺寸的按钮 |

* 示例

*# 让窗口可以拖动*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.drag(True)  
w.show()

### 遮罩[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "遮罩" \o "Direct link to 遮罩)

设置窗口遮罩层透明度,默认为0.5

* 函数

WebWindow(...).dim\_amount(dim:float=0.5)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| fol | float | 必填 | 数值在 0-1之间  0 :完全没有遮罩,1:完全黑的遮罩,0.5:半透明遮罩,默认为0.5 |

* 示例

*# 取消窗口遮罩*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.dim\_amount(0)  
w.show()

*# 窗口遮罩 为80%黑度*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.dim\_amount(0.8)  
w.show()

### 引力与位置[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "引力与位置" \o "Direct link to 引力与位置)

可更改 窗口的引力方向,随着引力方向的改变. 窗口的位置 属性也会随着引力点改变.

* 函数

WebWindow(...).gravity(gravity:int=None,offset\_x:int=None,offset\_y:int=None)

* 引力方向: 窗口靠左,窗口靠右,还是窗口居中 等.

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| gravity | number | 否 | 此值需传入 android.view.Gravity 中的常量值 |
| offset\_x | number | 否 | 距离 引力点 x轴 偏移位置 |
| offset\_y | number | 否 | 距离 引力点 y轴 偏移位置 |

* 示例

*# 窗口布局至左上角*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.gravity(3|48)  
w.show()

*# 窗口布局至左上角,偏移左上角 x:100px,y:200px*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.gravity(3|48,100,200)  
w.show()

#### 引力方向[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "引力方向" \o "Direct link to 引力方向)

| **android.view.Gravity 的常用常量** | **数值** | **备注** |
| --- | --- | --- |
| LEFT | 3 | 窗口居左 |
| TOP | 48 | 窗口居上 |
| RIGHT | 5 | 窗口居右 |
| BOTTOM | 80 | 窗口居底 |
| CENTER\_HORIZONTAL | 1 | 水平居中 |
| CENTER\_VERTICAL | 16 | 垂直居中 |
| CENTER | 17 | 屏幕居中 |

 Android 类的常量使用

可以 通过python 引入使用 from android.view import Gravity

也可以 直接填入常量数值使用

* 窗口的组合引力 应该使用 ‘|’ 连接多个引力值

*# 引力的组合写法*  
*# 右上角*  
WebWindow(...).gravity(48|5)  
  
*# 右下角*  
WebWindow(...).gravity(80|5)  
  
*#左上角*  
WebWindow(...).gravity(48|3)  
  
*#左下角*  
WebWindow(...).gravity(80|3)  
  
*#左垂直居中*  
WebWindow(...).gravity(3|16)  
  
*#右垂直居中*  
WebWindow(...).gravity(5|16)  
  
*#上横向居中*  
WebWindow(...).gravity(48|1)  
  
*#下横向居中*  
WebWindow(...).gravity(80|1)

* 示例

### 显示[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "显示" \o "Direct link to 显示)

展现UI,启动一个窗口展示html

* 函数

WebWindow(...).show()

* 示例

*# 显示一个窗口*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.show()

### 关闭[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "关闭" \o "Direct link to 关闭)

关闭当前WebWindow

* 函数

WebWindow(...).close()

* 示例

*# 显示一个窗口,2秒后关闭*  
from ascript.android.ui import WebWindow  
from ascript.android.system import R  
import time  
  
*# 构建一个WebWindow 显示‘/ui/a.html’*  
w = WebWindow(R.ui('a.html'))  
w.show()  
  
time.sleep(2)  
w.close()

### Python与Html交互[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "python与html交互" \o "Direct link to Python与Html交互)

#### Python 调用 JavaScript 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "python-调用-javascript-方法" \o "Direct link to Python 调用 JavaScript 方法)

请确定,在html 加载 javascript 代码中 存在将被调用的函数

*# 导包*  
from airscript.ui import WebWindow   
*#构建与展示UI*  
w = WebWindow(R(\_\_file\_\_).res('/ui/a.html'))  
w.show()  
  
*# 调用 javascript 中的 函数 fun1 ,并传入参数 "自在老师",2*  
w.call('fun1("自在老师",2)')

#### JavaScript 向 Python 传递数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "javascript-向-python-传递数据" \o "Direct link to JavaScript 向 Python 传递数据)

在html中,使用javascript 方法调用内置jairscript对象

<script>  
 *//这里的 key,value 用户填入自定义的值 ,在 python 的WebWindow 构造参数 tunnel 中会回调*  
 window.airscript.call('key',"value")  
</script>

#### JavaScript 中的快捷方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "javascript-中的快捷方法" \o "Direct link to JavaScript 中的快捷方法)

| **方法** | **备注** |
| --- | --- |
| airscript.close() | 关闭当前窗口 |
| airscript.get("key") | 获取 Python 中Kv 存储的数据,仅支持string数据 |
| airscript.save("key","value") | 快速存储数据到本地,和Python Kv函数数据互通,仅支持string数据 |

<script>  
 *//在JS 中直接关闭当前窗口*  
 window.airscript.close()  
</script>  
  
<script>  
 *//在JS 中获取 Python代码中 Kv 存储的值*  
 var value = window.airscript.get("key")  
</script>  
  
<script>  
 *//在JS 调用 save 将数据存储在本地 只支持存储string 类型的值.*  
 window.airscript.save("key","value")  
</script>

# 对话框

from ascript.android.ui import Dialog

### 吐司[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "吐司" \o "Direct link to 吐司)

展示一个吐司提示,用户无需操作,可自动消失

Dialog.toast(msg,dur=3000,gravity=1|16,x=0,y=200,bg\_color: str = None, color: str = None)

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| msg | string | 是 | 消息提示信息 |
| dur | number | 否 | 时间长度,单位毫秒 |
| gravity | number | 否 | [引力值](F:/cs01/cs01/cj2/docs/android/api/ui/window" \l "%E5%BC%95%E5%8A%9B%E6%96%B9%E5%90%91) |
| x | number | 否 | 在屏幕上的x坐标 -1 为屏幕中间 |
| bg\_color | string | 否 | 背景颜色值,rgb格式 如:"#000000" |
| color | string | 否 | 字体颜色值,rgb格式 如:"#FFFFFF" |

* 示例

*# 导包*  
from ascript.android.ui import Dialog  
  
*# 展示一个简单吐司提示,3秒后消失,位置偏屏幕中心点 y坐标+20%屏幕高度*  
Dialog.toast('我认识自在老师')  
  
*# 展示一个简单吐司提示,5秒后消失,位置偏屏幕中心点 y坐标+20%屏幕高度*  
Dialog.toast('我认识自在老师',5000)  
  
*# 展示一个简单吐司提示,3秒后消失,位置偏屏幕中心点 x+100px,y+200px*  
Dialog.toast('我认识自在老师',3000,100,200)  
  
*# 展示一个简单吐司提示,3秒后消失,位置偏屏幕中心点 y坐标-20%屏幕高度*  
Dialog.toast('我认识自在老师',3000,0,-0.2)  
  
*# 展示一个简单吐司提示,3秒后消失,位置偏屏幕中心点 y坐标+20%屏幕高度*  
Dialog.toast('我认识自在老师',3000,0,0.2)  
  
*# 展示一个简单吐司提示,3秒后消失,位置偏屏幕中心点 x坐标+20%屏幕宽度*  
Dialog.toast('我认识自在老师',3000,0.2,0)  
  
*# 展示一个简单吐司提示,3秒后消失,位置为屏幕左上角位置*  
Dialog.toast('我认识自在老师',3000,3|48,0,0)  
  
*# 展示一个简单吐司提示,3秒后消失,位置为屏幕左上角位置,且背景色为紫色,字体颜色黄色*  
Dialog.toast('我认识自在老师',3000,3|48,0,0,"#AD93FF","#F8C03E")

### 消息提示[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "消息提示" \o "Direct link to 消息提示)

弹出一个确认框,无取消按钮

* 函数

Dialog.alert(msg:str,submit:str="确认")

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| msg | string | 必填 | 消息提示信息 |
| submit | string | 必填 | 消息提示确认按钮文本,默认为‘确认’ |

* 示例

*# 导包*  
from ascript.android.ui import Dialog  
  
*# 案例1: 弹出一个信息提示框*  
Dialog.alert('您好,我是你们的API导师,自在老师.')

*# 导包*  
from ascript.android.ui import Dialog  
  
*# 案例2: 弹出一个信息提示框,并更改确认按钮为知道了*  
Dialog.alert('您好,我是你们的API导师,自在老师.','知道了')

### 消息确认[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "消息确认" \o "Direct link to 消息确认)

创建一个消息确认对话框

* 函数

*# 导包*  
Dialog.confirm(msg:str, title:str =None, submit:str="确认", cancel:str= "取消")

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| msg | str | 是 | 消息提示信息 |
| title | str | 否 | 设置标题 |
| submit | str | 否 | 设置确认按钮值 默认:确认 |
| cancel | str | 否 | 设置取消按钮值 默认:取消 |

* 示例

*# 弹出一个用户确认框*  
from ascript.android.ui import Dialog  
res = Dialog.confirm('您确定要学习aslib吗?')  
if res:  
 print("确定")  
else:  
 print("取消")

### 文本输入[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "文本输入" \o "Direct link to 文本输入)

创建文本输入框

Dialog.prompt(msg:str="请输入信息",title:str=None,value:str=None,hint:str=None,submit:str="确认",cancel:str="取消")

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| msg | string | 否 | 消息提示信息 默认:请输入信息 |
| title | string | 否 | 设置标题 |
| value | string | 否 | 输入框中的默认值 |
| hint | string | 否 | 输入框中的提示值 |
| submit | str | 否 | 设置确认按钮值 默认:确认 |
| cancel | str | 否 | 设置取消按钮值 默认:取消 |

* 属性方法

| **方法与参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| .title(msg) | string | 必填 | 设置标题 |
| .value(msg) | string | 必填 | 设置默认文本 |
| .hint(msg) | string | 必填 | 设置输入框中的默认提示 |
| .submit(msg) | string | 必填 | 设置确认按钮值 |
| .cancle(msg) | string | 必填 | 设置取消按钮值 |
| .close() |  |  | 关闭弹窗 |
| .show(pyfunction) | python 函数,需要两个形参 | 必填 | UI数据回调函数, 格式如: 'promat - 回调函数案例' 点击确认按钮后 k = "\_\_promat",v = 用户输入的内容 点击取消按钮后 k = "\_\_promat",v = "cancle" |

*# 弹出一个熟消息输入框*  
from ascript.android.ui import Dialog  
res = Dialog.prompt('您确定要学习aslib吗?')  
if res:  
 print(res)  
else:  
 print("取消")

# 图片展示

from ascript.android.ui import ImageWindow

## 展示图片[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "展示图片" \o "Direct link to 展示图片)

展示一张 Bitmap(内存图像) 或 File 文件图片

* 函数

ImageWindow.show(bitmap\_or\_file)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| bitmap\_or\_file | bitmap | str | 是 | Bitmap图像 或 OpenCv Image 或 图片文件路径 |

* 示例

*# 展示一张 图片文件*  
from ascript.android.ui import ImageWindow  
from ascript.android.system import R  
  
ImageWindow.show(R.img("a.png"))

*# 展示一张 Android 内存图像*  
from ascript.android.ui import ImageWindow  
from ascript.android import screen  
  
ImageWindow.show(screen.capture())

# 日志窗口

from ascript.android.ui import Loger

Loger 继承自 Window ,因此拥有Window中的所有方法,并且此Loger 是单例模式

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 创建日志窗口[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建日志窗口" \o "Direct link to 创建日志窗口)

创建 日志窗口

Loger()

### 展示日志窗口[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "展示日志窗口" \o "Direct link to 展示日志窗口)

显示日志窗口

Loger().show()

### 隐藏日志窗口[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "隐藏日志窗口" \o "Direct link to 隐藏日志窗口)

Loger().close()

#### 自定义日志窗口[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "自定义日志窗口" \o "Direct link to 自定义日志窗口)

自定义日志窗口需要下载[UI模版](http://nsnn.oss-cn-qingdao.aliyuncs.com/file/data/202308/51f9187aea12dc45bdd2f4c09b0ff59b.asui" \t "F:/cs01/cs01/cj2/_blank)

from airscript.ui.dialog import loger  
  
def tunner(k,v):  
 print(k,v)  
  
*# loger 继承 Window ,因此 Window 中的方法,loger都可以使用*  
lw = Loger(R.ui("loger.html"))  
lw.tunner(tunner) *# 设置消息通道*  
lw.show() *# 展示*

# 悬浮窗

from ascript.android.ui import FloatWindow

控件App小悬浮窗 **隐藏** 或**展示**,还可以定义展示样式

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 显示悬浮窗[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "显示悬浮窗" \o "Direct link to 显示悬浮窗)

使用 close方法来隐藏

* 函数

FloatWindow.show(x:float=None,y:float=None,dim:float=None)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| x | float | 否 | 悬浮窗显示的 坐标x 如果值为正数,则为固定坐标,如果为小数 0.2:20%的宽度 |
| x | float | 否 | 悬浮窗显示的 坐标y 如果值为正数,则为固定坐标,如果为小数 0.2:20%的高度 |
| dim | string | 否 | 悬浮窗透明度 0-1 之间,默认为1 |

* 示例

*# 显示悬浮窗*  
from ascript.android.ui import FloatWindow  
FloatWindow.show()

*# 更改悬浮窗位置*  
from ascript.android.ui import FloatWindow  
*# x为20%屏幕宽度, y 为20%屏幕高度*  
FloatWindow.show(0.2,0.4)

### 隐藏悬浮窗[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "隐藏悬浮窗" \o "Direct link to 隐藏悬浮窗)

使用 hide 方法来隐藏

* 函数

FloatWindow.hide()

* 示例

*# 隐藏悬浮窗*  
from ascript.android.ui import FloatWindow  
FloatWindow.hide()

### 注入按钮[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "注入按钮" \o "Direct link to 注入按钮)

向悬浮窗的展开菜单中,新增图标按钮

* 函数

FloatWindow.add\_menu(menu\_id:str,menu\_ico:str,menu\_click\_listener,remove\_system:bool=False)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| menu\_id | str | 是 | 按钮唯一id,随意填写 |
| menu\_ico | str | 是 | 按钮图标文件地址 |
| menu\_click\_listener | fun | 是 | 按钮点击后的回调函数 |
| remove\_system | bool | 否 | 默认False不移除系统按钮, True:移除系统按钮 |

* 示例

*# 向悬浮菜单中新增按钮*  
from ascript.android.ui import FloatWindow  
from ascript.android.system import R  
  
def a():  
 print("123")  
  
FloatWindow.add\_menu("123",R.img("a.png"),a)

*# 向悬浮菜单中新增按钮,并移除系统按钮*  
from ascript.android.ui import FloatWindow  
from ascript.android.system import R  
  
def a():  
 print("123")  
  
FloatWindow.add\_menu("123",R.img("a.png"),a,True)

# Http

[requests](https://requests.readthedocs.io/en/latest/" \t "F:/cs01/cs01/cj2/_blank) 是 Python 的Http 请求库,AirScript中内置了这个库

*# 导包*  
import requrest

### 模版[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "模版" \o "Direct link to 模版)

import requests  
  
*# 请求模版*  
r = requests.get("http://www.baidu.com")   
*# 打印状态Code*  
print(r.status\_code)   
*# 打印header 中的 content-type*  
print(r.headers['content-type'])   
*#打印字符编码*  
print(r.encoding)   
*#打印返回文本内容*  
print(r.text)   
  
*# 如果服务器返回json,可使用以下语句进行json转换*  
*# r.json()*

### get 案例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "get-案例" \o "Direct link to get 案例)

import requests  
*# 访问天气预报接口*  
r = requests.get("http://www.weather.com.cn/data/sk/101010100.html")  
*# 进行转码*  
r.encoding = r.apparent\_encoding  
*# 打印获取的所有文本信息*  
print(r.text)  
*# 转换为json对象*  
obj = r.json()  
  
print(obj['weatherinfo']['city']) *# 获取当前城市名称*

### post 案例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "post-案例" \o "Direct link to post 案例)

* Form 形式发送

import requests  
*# 使用post请求获取 ip所属信息*  
r = requests.post("https://api.oioweb.cn/api/ip/ipaddress",data ={"ip":"114.114.114.114"})  
  
*# 打印一下返回的文本*  
print(r.text)  
  
*# 看文本格式,服务器返回的是json数据,转换为json对象,方便拿属性*  
obj = r.json()  
  
*#获取 ip所在地址*  
print(obj["result"]["disp"])

* Json 形式发送

import requests  
import json  
*# 访问地址*  
url = 'http://httpbin.org/post'  
  
*# json 格式化*  
s = json.dumps({'key1': 'value1', 'key2': 'value2'})  
  
*# 发送 post请求*  
r = requests.post(url, data=s)  
  
*# 打印文本*  
print (r.text)

### 下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "下载" \o "Direct link to 下载)

* 小文件下载

*# 导入http模块*  
import requests  
*# 导入环境R,方便获取文件路径*  
from airscript.system import R  
  
*# 指定下载文件的地址*  
url = 'https://www.baidu.com/img/flexible/logo/pc/result@2.png' *# 目标下载链接*  
  
*# 通过get获取数据*  
r = requests.get(url)  
  
*# 保存文件至sd卡下的1.png*  
with open (R.sd("/1.png"), 'wb') as f:  
 f.write(r.content)

* 大文件下载

*# 导入http库*  
import requests  
*# 导入环境R,方便得到存储文件地址*  
from airscript.system import R  
*# 下载文件地址*  
url = 'https://www.baidu.com/img/flexible/logo/pc/result@2.png' *# 目标下载链接*  
  
*# get获取数据,传入stream 分批流形式获取数据*  
r = requests.get(url, stream=True)  
  
*# 打开sd卡文件,准备写入数据*  
f = open(R.sd("/2.png"), "wb")  
  
*# 循环批次写入数据*  
for chunk in r.iter\_content(chunk\_size=512):  
 if chunk:  
 f.write(chunk)  
  
*#关闭文件*  
f.close()

# WebSocket

长连接通信通道

### websocket-client[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "websocket-client" \o "Direct link to websocket-client)

AirScript 内置了 websocket-client Python包

我们可以使用该方法快速连接某一WebSocketServer

下面是使用案例:

from websocket import WebSocketApp  
  
def on\_message(self, message):  
 print("####### on\_message #######")  
 print("message：%s" % message)  
  
def on\_error(self, error):  
 print("####### on\_error #######")  
 print("error：%s" % error)  
  
def on\_close(self):  
 print("####### on\_close #######")  
  
def on\_open(self):  
 print("####### on\_open #######")  
  
  
url = "ws://192.168.31.108:10102/log/"  
  
ws = WebSocketApp(url,  
 on\_open=on\_open,  
 on\_message=on\_message,  
 on\_error=on\_error,  
 on\_close=on\_close)  
ws.run\_forever()

# MySql数据库

该模块需要开发者掌握一定的“SQL”语法

如果你还不会SQL语法,请前往[https://www.runoob.com/sql/sql-tutorial.html](https://www.runoob.com/sql/sql-tutorial.html" \t "F:/cs01/cs01/cj2/_blank) 学习相关语法.

小程序开发中,我们需要将大量数据存储在数据库,或从数据库获取数据.

### 云数据库账号注册[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "云数据库账号注册" \o "Direct link to 云数据库账号注册)

AirScript 为各位开发者 提供了 云数据库,免去开发者服务器部署环境.

开发者只需[注册,登录AirScript](http://py.airscript.cn/admin/db/list" \t "F:/cs01/cs01/cj2/_blank) 开发者账号,即可拥有自己的云端Mysql端数据库.

* 1. 登录开发者后台,并点击 "数据库"
  2. 点击 “新增” 增加数据库账号,与密码
  3. 这样你就生成了一个云端Mysql数据库

创建成功后得到: 主机地址,端口,用户名,数据库密码,数据库名 .  
  
pymsql连接时需使用.

* 1. 拿到账号密码,也可以登录[数据库云端管理后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 了,通过这个平台也可以管理数据库哦~

### 导包[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导包" \o "Direct link to 导包)

import pymysql

### 连接[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "连接" \o "Direct link to 连接)

Airscript 内置 pymysql 包方便对mysql数据库的访问

* 连接数据库

import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
  
  
*# 执行完毕后记得关闭db,不然会并发连接失败哦*  
db.close()

### 创建表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建表" \o "Direct link to 创建表)

我们可以配合,[数据库后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 来查看执行结果

*# 导包*  
import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
   
cursor = db.cursor()  
res = cursor.execute("create table stu(id int ,name varchar(20),gender int,age varchar(10))")  
print(res)  
  
*# 执行完之后要记得关闭游标和数据库连接*  
cursor.close()  
db.close()

我们发现 表已经创建成功了

### 插入数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插入数据" \o "Direct link to 插入数据)

我们可以配合,[数据库后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 来查看执行结果

*# 导包*  
import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
   
cursor = db.cursor()  
res = cursor.execute("INSERT INTO stu ( id, name,gender,age ) VALUES ( 10,'张三',1,'12岁')")  
db.commit(); *# 不要忘了提交,不然数据上不去哦*  
print(res)  
  
*# 执行完之后要记得关闭游标和数据库连接*  
cursor.close()  
db.close()

我们通过数据库后台,可以看到数据已经上去了.

### 查询数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "查询数据" \o "Direct link to 查询数据)

我们可以配合,[数据库后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 来查看执行结果

*# 导包*  
import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
   
cursor = db.cursor()  
cursor.execute("SELECT \* FROM stu WHERE name = '张三'" )  
results = cursor.fetchall()  
*# 循环遍历所有数据*  
for row in results:  
 *# 我们的表数据,总共4列,因此逐个获取每列数据*  
 print('id',row[0])  
 print('name',row[1])  
 print('gender',row[2])  
 print('age',row[3])  
  
*# 执行完之后要记得关闭游标和数据库连接*  
cursor.close()  
db.close()

### 更改数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "更改数据" \o "Direct link to 更改数据)

我们可以配合,[数据库后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 来查看执行结果

*# 导包*  
import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
   
cursor = db.cursor()  
res = cursor.execute("update stu set age = '99岁' WHERE name = '张三'")  
db.commit(); *# 不要忘了提交,不然数据上不去哦*  
print(res)  
  
*# 执行完之后要记得关闭游标和数据库连接*  
cursor.close()  
db.close()

### 删除数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除数据" \o "Direct link to 删除数据)

我们可以配合,[数据库后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 来查看执行结果

*# 导包*  
import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
   
cursor = db.cursor()  
*# res = cursor.execute("delete from stu") # 删除所有数据*  
res = cursor.execute("delete from stu where name='张三'")  
db.commit(); *# 不要忘了提交,不然数据上不去哦*  
print(res)  
  
*# 执行完之后要记得关闭游标和数据库连接*  
cursor.close()  
db.close()

### 删除表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除表" \o "Direct link to 删除表)

我们可以配合,[数据库后台](http://mysql.nspirit.cn" \t "F:/cs01/cs01/cj2/_blank) 来查看执行结果

*# 导包*  
import pymysql  
  
db = pymysql.connect(  
 host="8.140.162.237", *#开发者后台,创建的数据库 “主机地址”*  
 port=3307, *#开发者后台,创建的数据库 “端口”*  
 user='换成自己的用户名', *#开发者后台,创建的数据库 “用户名”*  
 password='换成自己的密码', *#开发者后台,创建的数据库 “初始密码”*  
 database ='换成自己的', *#开发者后台 ,创建的 "数据库"*  
 charset='utf8mb4' ""  
) *#连接数据库*  
   
cursor = db.cursor()  
res = cursor.execute("drop table stu") *# 这样这个表就被彻底删除了*  
db.commit(); *# 不要忘了提交,不然数据上不去哦*  
print(res)  
  
*# 执行完之后要记得关闭游标和数据库连接*  
cursor.close()  
db.close()  
  
  
  
*## Json*  
  
*### 转换*  
  
```python  
*#导包 标准json库*  
import json

在http数据传递,或者对象格式化时候.都需要 Json 和Object 的互相转换.

我们可以通过Python 的标准json库来完成操作

该库包含很多的 参数与方法,如需查看请点击 [json文档](https://docs.python.org/zh-cn/3/library/json.html" \t "F:/cs01/cs01/cj2/_blank)

#### class 转 json[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "class-转-json" \o "Direct link to class 转 json)

*# 导包json*  
import json  
  
*# 一个 object 对象*  
obj = [ { 'a' : 1, 'b' : 2, 'c' : 3, 'd' : 4, 'e' : 5 } ]  
  
*# 转换为 json String*  
jsonstr = json.dumps(obj)  
  
print(jsonstr)

#### json 转 class[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "json-转-class" \o "Direct link to json 转 class)

*# 导包json*  
import json  
  
*# 一段json 字符串*  
jsonStr = '[{"a": 1, "b": 2, "c": 3, "d": 4, "e": 5}]'  
  
*# 转换为对象,方便属性访问*  
obj = json.loads(jsonStr);  
  
*# 访问对象中的属性*  
print(obj[0]['a'])  
print(obj[0]['b'])

# 文件读写

文件读写中,我们使用内置函数[open](https://docs.python.org/zh-cn/3.8/library/functions.html" \l "open" \t "F:/cs01/cs01/cj2/_blank)操作文件

*# open 语法如下*  
open(file, mode='r', buffering=-1, encoding=None, errors=None, newline=None, closefd=True, opener=None)

## 路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "路径" \o "Direct link to 路径)

Android 是基于 Linux,因此文件系统和Linux 雷同.

在Android 中如果您的手机没有Root,那我们仅有权限自如的操作 "/sd" 卡路径下的文件.

### 友好的路径写法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "友好的路径写法" \o "Direct link to 友好的路径写法)

获取SD卡中文件路径 我们可以借助from airscript.system import R 环境模块.

以下是几个路径写法案例:

from airscript.system import R  
*# 获取SD卡根目录下的1.png 的完整路径*  
path = R.sd("1.png")   
print(path)

from airscript.system import R  
*# 获取SD卡根目录下的airscript文件夹下的 a.txt文件夹*  
path = R.sd("/airscript/a.txt")   
print(path)

## 创建文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建文件" \o "Direct link to 创建文件)

from airscript.system import R  
file = open(R.sd("a.txt"),mode='w')  
*# 上面这句代码意思:打开当前文件路径下的这个文件并采用写入的文件模式，当这个文件不存在的时候就会去创建*  
*# w、w+、a+、a模式都能创建新文件*  
file.close()

## 创建文件夹[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建文件夹" \o "Direct link to 创建文件夹)

创建文件夹用到了 os模块

import os  
from airscript.system import R  
path = R.sd("aaa") *# 在sd卡创建一个 aaa文件夹*  
os.mkdir(path)

## 删除文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除文件" \o "Direct link to 删除文件)

在Python 中删除文件的方法比较多

### 删除文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除文件-1" \o "Direct link to 删除文件)

os.remove(path)

import os  
from airscript.system import R  
*# 删除sd卡下的 1.txt文件*  
os.remove(R.sd("1.txt"))

### 删除文件夹[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除文件夹" \o "Direct link to 删除文件夹)

os.rmdir(path)

import os  
from airscript.system import R  
*# 删除sd卡下的 b文件夹*  
os.rmdir(R.sd("/b"))

## 读文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "读文件" \o "Direct link to 读文件)

### 读取所有内容[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "读取所有内容" \o "Direct link to 读取所有内容)

file.read()

from airscript.system import R  
file = open(R.sd("/1.txt"),"r+")  
str= file.read()  
print(str)  
file.close()

### 读取一行[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "读取一行" \o "Direct link to 读取一行)

file.readline([size])

读取整行，包括 "\n" 字符。

读取所有行并返回列表，若给定sizeint>0，返回总和大约为sizeint字节的行, 实际读取值可能比 sizeint 较大, 因为需要填充缓冲区。

from airscript.system import R  
file = open(R.sd("/1.txt"),"r+")  
content = file.readline()  
print(content)  
file.close()

### 读取所有行[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "读取所有行" \o "Direct link to 读取所有行)

file.readlines([sizeint])

读取所有行并返回列表，若给定sizeint>0，返回总和大约为sizeint字节的行, 实际读取值可能比 sizeint 较大, 因为需要填充缓冲区。

from airscript.system import R  
file = open(R.sd("/1.txt"),"r+")  
for line in file.readlines():  
 print(line)  
  
file.close()

## 写文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "写文件" \o "Direct link to 写文件)

### 写入字符串[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "写入字符串" \o "Direct link to 写入字符串)

file.write(str)

将字符串写入文件，返回的是写入的字符长度。

*# w+模式打开此文件后,每次写都会清空源文件内容,重新写入*  
from airscript.system import R  
file = open(R.sd("/1.txt"),"w+")  
file.write("自在老师")  
file.close()

### 写入字符串序列[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "写入字符串序列" \o "Direct link to 写入字符串序列)

file.writelines(sequence)

向文件写入一个序列字符串列表，如果需要换行则要自己加入每行的换行符。

from airscript.system import R  
file = open(R.sd("/1.txt"),"w+")  
strs = ["自在老师\n","你好"]  
file.writelines(strs)  
file.close()

### 在文件末尾追加[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "在文件末尾追加" \o "Direct link to 在文件末尾追加)

在末尾读写 使用到了 seek函数

from airscript.system import R  
file = open(R.sd("/1.txt"),"r+")  
file.seek(0, 2)  
file.write("我会追加在末尾")  
file.close()

# 多媒体

*# 包*  
from ascript.android import media

音视频播放,录制,邮件发送 等操作

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 音量调节[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "音量调节" \o "Direct link to 音量调节)

调节音量大小(1-100)

* 函数

media.volume(percent:int,type:int =3)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| percent | int | 是 | 音量大小 1-100之间 |
| type | int | 否 | 音量类型,默认为3,见下表 |

#### 什么是音量类型?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "什么是音量类型" \o "Direct link to 什么是音量类型?)

在Android 系统中,包含了很多音量类型,如 音乐,系统通知,闹钟,电话等等.不同的类型,都有自己的音量

该参数来自 android.media.AudioManager

| **音量类型** | **备注** |
| --- | --- |
| 音乐回放即媒体音量 | 3 |
| 窗口顶部状态栏Notification | 5 |
| 警告 | 4 |
| 铃声 | 2 |
| 系统 | 1 |
| 通话 | 0 |

* 示例

*# 设置音乐媒体音量 为80%*  
*# 导包*  
from ascript.android import media  
media.volume(80)

*# 设置通话音量为 60%*  
*# 导包*  
from ascript.android import media  
media.volume(60,0)

### 音量获取[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "音量获取" \o "Direct link to 音量获取)

获取音量大小

* 函数

media.get\_volume(type:int =3)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| type | int | 否 | 音量类型,默认为3,见下表 |

#### 什么是音量类型?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "什么是音量类型-1" \o "Direct link to 什么是音量类型?)

在Android 系统中,包含了很多音量类型,如 音乐,系统通知,闹钟,电话等等.不同的类型,都有自己的音量

该参数来自 android.media.AudioManager

| **音量类型** | **备注** |
| --- | --- |
| 音乐回放即媒体音量 | 3 |
| 窗口顶部状态栏Notification | 5 |
| 警告 | 4 |
| 铃声 | 2 |
| 系统 | 1 |
| 通话 | 0 |

* 示例

*# 获取媒体音量 音量大小*  
*# 导包*  
from ascript.android import media  
num = media.get\_volume(3)  
print(num)

### 语音朗读[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "语音朗读" \o "Direct link to 语音朗读)

将文本用语音朗读出来

* 函数

media.talk(msg:str)

部分设备不支持

部分设备缺少语音库,因此不支持,如(雷电模拟器)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| text | string | 是 | 要朗读的文本 |

* 示例

*# 文本转语音朗读出来*  
*#导包*  
from ascript.android import media  
media.talk('自在老师你好')

### 播放音频[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "播放音频" \o "Direct link to 播放音频)

播放音频文件

* 函数

media.play(path:str,callback=None)

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| path | string | 必填 | 要播放的音频文件 |
| callback | class | 可选 | 一个回调对象 回调对象格式见案例 |

* 示例

*# 播放一个音频文件*  
*# 导包*  
from ascript.android import media  
from ascript.android.system import R   
media.play(R.res("/media/aodi.wav"))

*# 播放音频文件 回调案例*  
from ascript.android import media  
from ascript.android.system import R   
  
class listener:  
 def prepare(self,player):  
 *# 获取音频时长*  
 dur = player.getDuration()  
 print(dur)  
 print('准备完毕')  
 *# 带监听器,我们需要调用start方法*  
 player.start()  
  
 def completion(self):  
 print('播放完毕')  
  
*# 播放一个音频*  
media.play(R.res("/media/aodi.wav"),listener())

#### 停止音频[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "停止音频" \o "Direct link to 停止音频)

当开始播放 时候,我们可以通过 callback拿到 player对象,从而停止音频

player.stop()

* 示例

*# 播放音频文件开始后,停止音频*  
from ascript.android import media  
from ascript.android.system import R   
import time  
  
class listener:  
 def prepare(self,player):  
 *# 获取音频时长*  
 dur = player.getDuration()  
 print(dur)  
 print('准备完毕')  
 *# 带监听器,我们需要调用start方法*  
 player.start()  
 time.sleep(1)  
 *# 等待1秒后,立刻停止音频播放*  
 player.stop()  
  
 def completion(self):  
 print('播放完毕')  
  
*# 播放一个音频*  
media.play(R.res("/media/bili.WAV"),listener())  
  
print("123")

#### 倍速播放[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "倍速播放" \o "Direct link to 倍速播放)

倍速播放,需要通过player(Android MediaPlayer) 对象,获取param,并设置速度

* 示例

*# 倍速播放音频文件*  
from ascript.android import media  
from ascript.android.system import R   
import time  
  
class listener:  
 def prepare(self,player):  
 *# 获取音频时长*  
 dur = player.getDuration()  
 print(dur)  
 *# 更改播放速度*  
 playparam = player.getPlaybackParams()  
 playparam.setSpeed(1.0) *#2倍速*  
 player.setPlaybackParams(playparam)  
 print('准备完毕')  
 *# 带监听器,我们需要调用start方法*  
 player.start()  
  
 def completion(self):  
 print('播放完毕')  
  
*# 播放一个音频*  
media.play(R.res("/media/bili.WAV"),listener())  
  
print("123")

### 获取录音对象[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取录音对象" \o "Direct link to 获取录音对象)

获取录制音频对象

* 函数

media.recode(path:str,time=None):

* 参数

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| path | string | 必填 | 录制音频的文件放置路径,如果文件不存在,会自动创建 |
| time | class | 可选 | 录制音频的时长,如果不填写.则直到stop调用后才停止 |

* 返回值

该方法会返回一个 MediaRecoder java对象.

#### 录音对象(MediaRecoder)[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "录音对象mediarecoder" \o "Direct link to 录音对象(MediaRecoder))

我们可以调用该对象的方法,控制开始录音,结束录音,设置录音时长,设置录音文件最大长度 等..

下面是该对象的常用方法:

| **常用方法** | **备注** |
| --- | --- |
| prepare() | 在设置好所有参数后,调用该方法. |
| start() | 开始录制音频 |
| stop() | 结束录制音频 |

更多方法请参阅 android.media.MediaRecorder 使用相关文档.

*# 案例: 录制音频,并在3秒后调用stop停止*  
from ascript.android import media  
import time  
m = media.recode("/sdcard/1.acc")  
  
*# 开始录音*  
m.start();   
  
*# 睡眠3秒后 停止录音*  
time.sleep(3)  
m.stop()

### 设备震动[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备震动" \o "Direct link to 设备震动)

设备震动一段时间,单位毫秒

* 函数

media.vibrate(time:int=200):

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| time | string | 必填 | 震动的时长,单位毫秒 默认:200毫秒 |

*# 手机震动2秒*  
*# 导包*  
from ascript.android import media  
media.vibrate(2000)

### 图片文件存储相册并更新[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图片文件存储相册并更新" \o "Direct link to 图片文件存储相册并更新)

将图片文件存储至相册中,并通知系统更新相册.

用于一些上传图片文件的场景.

* 函数

media.image\_to\_gallery(file\_path:str):

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| file\_path | string | 必填 | 图片文件的路径地址 |

*# 将logo图片,存储至系统相册,并通知系统更新.*  
*# 导包*  
from ascript.android import media  
media.image\_to\_gallery(R.img("logo.png"))

### 发送邮件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发送邮件" \o "Direct link to 发送邮件)

*# 导入 smtp邮件包*  
import smtplib

在这里我们使用 Python库 smtplib 实现邮件的发送

### 阿里云语音合成[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "阿里云语音合成" \o "Direct link to 阿里云语音合成)

[阿里云官网](https://ram.console.aliyun.com" \t "F:/cs01/cs01/cj2/_blank)

#### TTS合成[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "tts合成" \o "Direct link to TTS合成)

##### 初始化[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "初始化" \o "Direct link to 初始化)

初始化阿里云语音合成对象

* 函数

media.AliTts(app\_key:str,  
 ak\_id:str,  
 ak\_secret:str,  
 font\_name:str="siqi",  
 speed\_level:str="1",  
 pitch\_level:str="0",  
 volume:int=1  
 )

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| app\_key | dict | 必填 | 阿里云后台创建的appkey |
| ak\_id | dict | 必填 | 阿里云后台创建的AccessKey ID |
| ak\_secret | dict | 必填 | 阿里云后台创建的AccessKey Secret |
| font\_name | str | 否 | 发音人 ,[音色列表](https://help.aliyun.com/document_detail/84435.html" \t "F:/cs01/cs01/cj2/_blank),默认:siqi |
| speed\_level | str | 否 | 语速，取值范围：0～2，默认值：1.0，值越大语速越快。 |
| pitch\_level | string | 否 | 语调，取值范围：-500～500，默认值：0，值越大声音越尖锐。 |
| volume | int | 否 | 音量默认1, 取值:1-100之间 |

*# 初始化TTS 语音合成器*  
from ascript.android import media  
  
app\_key = "\*\*\*\*"  
ak\_id = "\*\*\*\*"  
ak\_secret = "\*\*\*\*\*\*"  
*# 上面3个值,需要传入自己的阿里云后台数据.*  
  
tts = media.AliTts(app\_key,ak\_id,ak\_secret)

*# 初始化TTS 语音合成器, 语速2倍,音调300*  
from ascript.android import media  
  
app\_key = "\*\*\*\*"  
ak\_id = "\*\*\*\*"  
ak\_secret = "\*\*\*\*\*\*"  
*# 上面3个值,需要传入自己的阿里云后台数据.*  
  
tts = media.AliTts(app\_key,ak\_id,ak\_secret,speed\_level="2.0",pitch\_level="300")

##### 合成语音[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "合成语音" \o "Direct link to 合成语音)

将文字转换为音频

media.AliTts(...).start(tts\_msg:str,auto\_play:bool=True,save\_file=None)

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| tts\_msg | dict | 是 | 转音频的文字 |
| ak\_id | dict | 否 | 是否自动播放,默认自动播放 |
| ak\_secret | dict | 否 | 要存储语音的路径 |

* 示例

*# 文字转语音*  
from ascript.android import media  
  
app\_key = "\*\*\*\*"  
ak\_id = "\*\*\*\*"  
ak\_secret = "\*\*\*\*"  
  
tts = media.AliTts(app\_key,ak\_id,ak\_secret)  
  
tts.start("遥遥领先AScript!")

#### CosyVoice合成[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "cosyvoice合成" \o "Direct link to CosyVoice合成)

##### 初始化[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "初始化-1" \o "Direct link to 初始化)

初始化阿里云语音合成对象

* 函数

media.AliTtsCosyVoice(self, app\_key: str,  
 ak\_id: str,  
 ak\_secret: str,  
 voice: str = "longze",  
 enable\_subtitle: bool = True,  
 format: str = "pcm",  
 sample\_rate: int = 16000,  
 volume: int = 50,  
 speech\_rate: int = 0,  
 pitch\_rate: int = 0  
 ):

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| app\_key | dict | 必填 | 阿里云后台创建的appkey |
| ak\_id | dict | 必填 | 阿里云后台创建的AccessKey ID |
| ak\_secret | dict | 必填 | 阿里云后台创建的AccessKey Secret |
| voice | str | 否 | 发音人 ,[音色列表](https://help.aliyun.com/zh/isi/developer-reference/natual-tts-product-introduction?spm=a2c4g.11186623.help-menu-30413.d_3_1_1_0.e7f957577q8RfI" \t "F:/cs01/cs01/cj2/_blank),默认longze |
| sample\_rate | str | 否 | 音频采样率,默认16000 |
| volume | string | 否 | 朗读音量，范围是0~100，默认50。 |
| speech\_rate | int | 否 | 语速，范围是-500~500，默认是0。 |
| pitch\_rate | int | 否 | 语调，取值范围：-500～500，默认值：0，值越大声音越尖锐。 |

*# 初始化TTS 语音合成器*  
from ascript.android import media  
  
app\_key = "\*\*\*\*"  
ak\_id = "\*\*\*\*"  
ak\_secret = "\*\*\*\*\*\*"  
*# 上面3个值,需要传入自己的阿里云后台数据.*  
  
tts = media.AliTtsCosyVoice(app\_key,ak\_id,ak\_secret)

*# 初始化TTS 语音合成器, 语速200,音调300*  
from ascript.android import media  
  
app\_key = "\*\*\*\*"  
ak\_id = "\*\*\*\*"  
ak\_secret = "\*\*\*\*\*\*"  
*# 上面3个值,需要传入自己的阿里云后台数据.*  
  
tts = media.AliTts(app\_key,ak\_id,ak\_secret,speech\_rate=100,pitch\_rate="300")

##### 合成并播放[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "合成并播放" \o "Direct link to 合成并播放)

将文字转换为音频

media.AliTtsCosyVoice(...).start(tts\_msg:str,listener=None)

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| tts\_msg | dict | 是 | 转音频的文字 |
| listener | dict | 否 | 回调函数,需要有一个形参, 回调0 为开始播放. 回调1播放结束 |

* 示例

*# 文字转语音*  
from ascript.android import media  
app\_key = "\*\*\*"  
ak\_id = "\*\*\*"  
ak\_secret = "\*\*\*\*\*\*"  
  
res = media.AliTtsCosyVoice(app\_key,ak\_id,ak\_secret,voice="longlaotie")  
  
def listener(code):  
 if code==1:  
 print("播放结束")  
  
res.start("哦~,宝贝儿",listener)

# 意图

意图是Android 四大组件(Activity,Service,content provider,broadcast receiver)的核心

他可以启动应用,启动服务 等操作.

## 安卓 Intent[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "安卓-intent" \o "Direct link to 安卓 Intent)

*# 导包*  
from airscript.intent import Intent

此Intent 继承自 android.content.Intent

接下来我们将用混编完成意图代码编写

### android.content.Intent[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "androidcontentintent" \o "Direct link to android.content.Intent)

详细属性方法请查阅[Android Intent](https://developer.android.google.cn/guide/components/intents-common?hl=zh_cn" \l "java" \t "F:/cs01/cs01/cj2/_blank)

也可以百度Android Intent 相关用法.

### 案例:发短信[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "案例发短信" \o "Direct link to 案例:发短信)

*#导包*  
from android.content import Intent   
from ascript.android.system import R  
  
it = Intent(Intent.ACTION\_VIEW)   
it.putExtra('sms\_body', '你好这是短信内容');  
it.setType('vnd.android-dir/mms-sms');  
  
*# 部分Activity 启动需要一个新的TASK ,通常都固定写这一行代码*  
it.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);  
R.context.startActivity(it);

### 案例:抖音直播间跳转[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "案例抖音直播间跳转" \o "Direct link to 案例:抖音直播间跳转)

*# 通过 房间ID 跳转抖音直播间*  
*# 导包*  
from android.content import Intent   
from ascript.android.system import R  
from android.net import Uri  
  
roomid = '6901450157649300232'  
uri = Uri.parse("snssdk1128://live?room\_id="+roomid);  
it = Intent(Intent.ACTION\_VIEW,uri)   
it.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);  
R.context.startActivity(it);

### 案例:跳转应用详情[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "案例跳转应用详情" \o "Direct link to 案例:跳转应用详情)

from airscript.intent import Intent   
from ascript.android.system import R  
from android.net import Uri  
from android.provider import Settings  
  
*# 根据需求改变包名,即可跳转,跳转后,可点击停止程序等等.*  
intent = Intent(Settings.ACTION\_APPLICATION\_DETAILS\_SETTINGS).setData(Uri.fromParts("package", "com.aojoy.airscript", None))  
intent.addFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);  
R.context.startActivity(intent);

### 案例:跳转设置界面[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "案例跳转设置界面" \o "Direct link to 案例:跳转设置界面)

from airscript.intent import Intent   
from android.content import ComponentName  
from ascript.android.system import R  
  
intent = Intent()  
package = "com.android.settings"  
activity = "com.android.settings.Settings"  
componentName = ComponentName(package, activity)  
intent.setComponent(componentName)  
intent.setAction("android.intent.action.VIEW");  
  
*# 这是固定的启动方法*  
intent.setFlags(Intent.FLAG\_ACTIVITY\_NEW\_TASK);  
R.context.startActivity(intent);

# 云控

该部分API 可以让设备主动连接至 [官方云控](http://control.ascript.cn:8088" \t "F:/cs01/cs01/cj2/_blank) 或 **开发者自搭建的云控平台**.

*# 导包*  
from ascript.android import cloud\_control

## 连接云控[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "连接云控" \o "Direct link to 连接云控)

让设备连接至云控平台, **设备只能连接官方云控或自定义云控其中之一**

### 连接至官方云控[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "连接至官方云控" \o "Direct link to 连接至官方云控)

让设备连接至[官方云控:http://control.ascript.cn:8088](http://control.ascript.cn:8088" \t "F:/cs01/cs01/cj2/_blank)

连接后可在云端控制设备,远程操控设备,定时下发执行任务等.

* 方法

cloud\_control.connect\_to\_as(dev\_user:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| dev\_user | str | 是 | 开发者账号,目前需要时vip用户才可使用 |

* 示例

from ascript.android import cloud\_control  
cloud\_control.connect\_to\_as("你自己的开发者账号")

### 连接至自定义云控[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "连接至自定义云控" \o "Direct link to 连接至自定义云控)

* 方法

cloud\_control.connect\_to\_ws(ws\_path:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| ws\_path | str | 是 | websocket server 地址.您需要自己搭建ws服务. |

* 示例

from ascript.android import cloud\_control  
cloud\_control.connect\_to\_as("你自己的开发者账号")

## 发送数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发送数据" \o "Direct link to 发送数据)

发送数据,可以在云控查数值

* 方法

cloud\_control.send(data:dict)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| data | dict | 是 | 字典形式的数据. 在云控服务器中可通过key值,显示value值 |

* 示例

*# 发送数据给云控,云控中可实时显示*  
from ascript.android import cloud\_control  
data = {"gold":180,"water":200}  
cloud\_control.send(data)

## 关闭云控[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "关闭云控" \o "Direct link to 关闭云控)

关闭和云端控制的连接

* 方法

cloud\_control.close()

* 示例

*# 关闭云控*  
from ascript.android import cloud\_control  
cloud\_control.close()

# 程序

from ascript.android import system

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 关闭[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "关闭" \o "Direct link to 关闭)

关闭当前程序

* 函数

system.exit()

* 示例

*#退出小程序 3秒后*  
from ascript.android import system  
import time  
  
time.sleep(3)  
system.exit()

### 重启[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "重启" \o "Direct link to 重启)

重启当前程序

* 函数

system.reboot(delay\_time:int=0)

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| delay\_time | int | 否 | 延迟重启的时间,单位毫秒 默认:0,立即重启 |

* 示例

*#重启小程序 立刻*  
from ascript.android import system  
import time  
  
print("我启动了")  
  
system.reboot()

*#重启小程序 3秒后*  
from ascript.android import system  
import time  
  
print("我启动了")  
  
system.reboot(3000)

### 启动应用[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "启动应用" \o "Direct link to 启动应用)

打开已安装的应用

部分手机无法启动应用 **解决方案**!

如“华为手机”在系统设置中有 **启动关联开关选项**,此开关需要手动操作 允许关联应用启动.

开启后,使用system.open() 可启动任意应用

* 函数

system.open(name\_or\_package:str)

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| name\_or\_package | str | 是 | 启动的应用名称或包名 注意:"包名启动快于应用名称启动" **包名称可以通过[APP-开发者-查包工具]获得** |

* 示例

from ascript.android import system  
  
*# 根据应用名称启动. PS:启动略慢于包名启动*  
system.open("微信")

from ascript.android import system  
  
*# 根据包名启动,推荐使用*  
system.open("com.autonavi.minimap")

### 打开网页[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "打开网页" \o "Direct link to 打开网页)

用浏览器打开网页

* 函数

system.browser(url:str)

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| url | str | 是 | 启动的网址 如:([http://www.airscript.com](http://www.airscript.com" \t "F:/cs01/cs01/cj2/_blank)) |

* 示例

*# 案例:打开AirSctipt 官网*  
*#导包*  
from ascript.android import system  
  
*# 输入网址,用浏览器打开*  
system.browser("http://www.airscript.cn")

### 获取当前运行APP[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取当前运行app" \o "Direct link to 获取当前运行APP)

获取当前正在运行的APP,默认获取1分钟内打开的操作.

* 函数

system.get\_foreground\_app()

* 返回

app包名: str

* 示例

*# 案例:获取当前运行的app包名*  
*#导包*  
from ascript.android import system  
  
app\_package = system.get\_foreground\_app()  
print(app\_package)

### Root指令[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "root指令" \o "Direct link to Root指令)

重启当前程序

Root设备

只有Root 设备可用

* 函数

system.shell(adb\_root:str)

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| adb\_root | str | 否 | 可执行的 shell指令 |

* 返回值

[Result](F:/cs01/cs01/cj2/1extracted_div_content.html)

包含两个属性: res:shell指令的回执消息. error:如果命令行指令出错,返回的错误信息

from ascript.android import system  
  
r = system.shell("pm list packages")  
  
*# 打印 命令行回执消息*  
if r.res:  
 for x in r.res:  
 print(x)  
  
*# 打印 命令行错误回执消息*  
if r.error:  
 print("命令行指令出错了")  
 for x in r.error:  
 print(x)

### 云控监听[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "云控监听" \o "Direct link to 云控监听)

云控端,发送自定义消息后,在此监听消息

* 函数

system.channel(fun)

| **参数** | **类型** | **必须** | **备注** |
| --- | --- | --- | --- |
| fun | python 函数 | 必填 | 需要一个形式参数,作为消息回调 |

*# 导入消息通道*  
from ascript.android import system  
  
*# 创建一个Python函数,作为消息通道回调*  
def airws(msg):  
 print(msg)  
  
*# 建立消息回调通道*  
system.channel(airws)

# 设备信息

from ascript.android.system import Device

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 设备ID[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备id" \o "Direct link to 设备ID)

获取设备的唯一ID

* 函数

Device.id()

* 示例

*# 获取设备ID*  
from ascript.android.system import Device  
device\_id = Device.id()  
print(device\_id)

### 设备名称[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备名称" \o "Direct link to 设备名称)

获取手机名称

* 函数

Device.name()

* 示例

*# 获取设备名称*  
from ascript.android.system import Device  
device\_name = Device.name()  
print(device\_name)

### 屏幕信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "屏幕信息" \o "Direct link to 屏幕信息)

获取手机设备的屏幕分辨率与屏幕密度

* 函数

Device.display()

* 返回值

| **DisplayMetrics** | **属性** | **备注** |
| --- | --- | --- |
|  | heightPixels | 屏幕高度 |
|  | widthPixels | 屏幕宽度 |
|  | density | 屏幕密度 |

* 示例

*# 获取屏幕信息*  
from ascript.android.system import Device  
display = Device.display()  
*# 屏幕宽度*  
print(display.widthPixels)  
*# 屏幕高度*  
print(display.heightPixels)  
*# 屏幕密度*  
print(display.density)

### 设备品牌[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备品牌" \o "Direct link to 设备品牌)

获取设备品牌名称

* 函数

Device.brand()

* 示例

*# 获取设备品牌*  
from ascript.android.system import Device  
device\_brand = Device.brand()  
print(device\_brand)

### 设备型号[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备型号" \o "Direct link to 设备型号)

获取设备型号

* 函数

Device.model()

* 示例

*# 获取设备型号*  
from ascript.android.system import Device  
device\_model = Device.model()  
print(device\_model)

### 设备SDK版本[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备sdk版本" \o "Direct link to 设备SDK版本)

获取SDK版本

* 函数

Device.sdk()

* 示例

*# 设备SDK版本*  
from ascript.android.system import Device  
device\_sdk = Device.sdk()  
print(device\_sdk)

### 设备Android版本[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备android版本" \o "Direct link to 设备Android版本)

设备Android版本

* 函数

Device.version()

* 示例

*# 设备Android版本*  
from ascript.android.system import Device  
device\_version = Device.version()  
print(device\_version)

### 设备本地IP地址[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备本地ip地址" \o "Direct link to 设备本地IP地址)

设备本地IP地址

* 函数

Device.ip()

* 示例

*# 设备本地IP地址*  
from ascript.android.system import Device  
device\_ip = Device.ip()  
print(device\_ip)

### 设备运行信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设备运行信息" \o "Direct link to 设备运行信息)

获取当前设备运行的APP信息

* 函数

Device.current\_appinfo()

* 返回值

| **DeviceInfo** | **属性** | **备注** |
| --- | --- | --- |
|  | name | 当前运行的APP名称 |
|  | packageName | 当前运行APP的包名称 |
|  | activity | 当前运行APP的Activity |

* 示例

*# 获取当前设备运行的APP信息*  
from ascript.android.system import Device  
device\_info = Device.current\_appinfo()  
print(device\_info.name)  
print(device\_info.packageName)  
print(device\_info.activity)

### 已安装程序列表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "已安装程序列表" \o "Direct link to 已安装程序列表)

获取所有已安装APP的信息

* 函数

Device.apps()

* 返回值

AppBean[]

| **AppBean** | **属性** | **备注** |
| --- | --- | --- |
|  | appName | 当前运行的APP名称 |
|  | appSize | app 大小 |
|  | isSd() | 是否在SD卡中 |
|  | isSystem() | 是否为系统应用 |
|  | appPackageName | app包名 |
|  | apkPath | 安装路径 |

* 示例

*#获取所有已安装APP的信息*  
from ascript.android.system import Device  
apps = Device.apps()  
print(len(apps)) *#共安装了多个APP*  
for app in apps:  
 print(app.appName) *# app名称*  
 print(app.appSize) *# app大小*  
 print(app.isSd()) *# 是否在sd卡*  
 print(app.isSystem()) *# 是否为系统应用*  
 print(app.appPackageName) *# app包名*  
 print(app.apkPath) *# app 安装地址*

### 电池电量[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "电池电量" \o "Direct link to 电池电量)

获取当前电池电量数

* 函数

Device.battery()

* 返回值

int类型 电池电量 1-100 之间

* 示例

*# 获取当前设备运行的APP信息*  
from ascript.android.system import Device  
num = Device.battery()  
print(num)

### 应用内存[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "应用内存" \o "Direct link to 应用内存)

获取当前APP内存信息

* 函数

Device.memory()

* 返回值

list 数组,其中包含3个元素 剩余内存,已用内存,总共内存 ,单位字节

* 示例

*# 获取当前设备运行的APP信息*  
from ascript.android.system import Device  
info = Device.memory()  
*# 返回单位是字节*  
print(f"剩余内存:{info[0]},已用内存{info[1]},总共内存{info[2]}")

# 资源与环境

from ascript.android.system import R

## 常量[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "常量" \o "Direct link to 常量)

### Android上下文[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "android上下文" \o "Direct link to Android上下文)

* 获取上下文 **Android Context** ,在android系统中,上下文有很多的作用. 打开意图 获取资源 等

R.context

### 工程名称[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "工程名称" \o "Direct link to 工程名称)

* 获取当前工程名称

R.name

### 当前小程序ID[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当前小程序id" \o "Direct link to 当前小程序ID)

* 获取当前运行的小程序ID

如果当前运行的非线上小程序,返回0

R.id

### 当前APP包名[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当前app包名" \o "Direct link to 当前APP包名)

* 获取当前程序的包名 可根据此id判断是否为打包小程序

R.package\_name

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 获取SD卡目录[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取sd卡目录" \o "Direct link to 获取SD卡目录)

获取手机SD卡目录

* 函数

R.sd(path =None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| path | int | 否 | 子文件路径 默认:None,返回当前工程路径 |

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html) : SD卡地址 或 SD卡和子目录的拼接地址

* 示例

*# 获取SD卡根路径*  
from ascript.android.system import R  
path = R.sd()  
print(path)

*# 获取SD卡下的 1.png 文件路径*  
from ascript.android.system import R  
path = R.sd("1.png")  
print(path)

### 获取工程目录[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取工程目录" \o "Direct link to 获取工程目录)

获取当前运行的工程目录

* 函数

R.root(path =None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| path | int | 否 | 子文件路径 默认:None,返回当前工程路径 |

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html) : 根目录 或 根目录和子目录的拼接地址

* 示例

*# 获取系统根目录*  
from ascript.android.system import R  
path = R.root()  
print(path)

*# 获取根目录下的子文件1.txt*  
from ascript.android.system import R  
path = R.root("1.txt")  
print(path)

### 获取res目录[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取res目录" \o "Direct link to 获取res目录)

获取 \*\*当前工程/res/\*\*资源目录路径

* 函数

R.res(childpath =None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| childpath | int | 否 | 相对于res的子目录,默认,返回资源目录 |

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html) : 文件地址

* 示例

*# 获取资源文件根目录*  
from ascript.android.system import R  
path = R.res()  
print(path)

*# 获取资源文件*  
from ascript.android.system import R  
path = R.res("img/a.png")  
print(path)

### 获取相对路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取相对路径" \o "Direct link to 获取相对路径)

获取 **当前文件**的相对路径

* 函数

R.rel(\_\_file\_\_,path):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| path | str | 是 | 获取相对于当前文件的相对路径,不能以 “/”开头,否则就是绝对路径. |

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html) : 文件地址

*#获取上级目录下res/img/logo.png*  
path = R.rel(\_\_file\_\_,"../res/img/logo.png")  
print(path)

*#获取同级目录的文件绝对路径*  
path = R.rel(\_\_file\_\_, "res/img/logo.png")  
print(path)

### 获取img目录[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取img目录" \o "Direct link to 获取img目录)

**当前工程/res/img**

* 函数

R.img(childpath =None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| childpath | int | 否 | 相对于res/img的子目录,默认,返回res/img目录 |

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html) : 文件地址

* 示例

*# 获取工程下的图片资源根目录*  
from ascript.android.system import R  
path = R.img()  
print(path)

*# 获取工程下的图片资源*  
from ascript.android.system import R  
path = R.img("logo.png")  
print(path)

### 获取ui目录[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取ui目录" \o "Direct link to 获取ui目录)

**当前工程/res/ui**

* 函数

R.ui(childpath =None):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| childpath | int | 否 | 相对于res/ui/childpath,默认,返回res/ui |

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html) : 文件地址

* 示例

*# 获取工程下的UI资源路径*  
from ascript.android.system import R  
path = R.ui()  
print(path)

*# 获取工程目录下的 UI 子文件路径*  
from ascript.android.system import R  
path = R.ui("a.html")  
print(path)

# 数据缓存

from ascript.android.system import KeyValue

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 存储数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "存储数据" \o "Direct link to 存储数据)

将数据 通过 键值对的形式存储在本地, **程序关闭后数据不会丢失**

* 函数

KeyValue.save(key:str,value)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| colors | str | 是 | 数据的id |
| value | str,int,float,bool | 是 | 数据的值 |

* 示例

*# 将数据存储在本地*  
from ascript.android.system import KeyValue  
  
KeyValue.save('name','自在老师')  
KeyValue.save('age',18)

### 存储获取[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "存储获取" \o "Direct link to 存储获取)

通过数据键(Key) 获取Value值

* 函数

KeyValue.get(key:str,default\_value)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| key | str | 是 | 数据的id |
| default\_value | str,int,float,bool | 是 | 如果本地没有存储数据,获取到的默认值,默认值的数据类型必须和 存入值一致 |

* 示例

*# 从本地获取数据*  
from ascript.android.system import KeyValue  
name = KeyValue.get('name','')  
print(name)  
age = KeyValue.get('age',0)  
print(age)

### 清空数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "清空数据" \o "Direct link to 清空数据)

通过数据键(Key) 清空Value值.

* 函数

KeyValue.remove(key:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| key | str | 是 | 数据的id |

* 示例

*# 从本地获取数据*  
from ascript.android.system import KeyValue  
name = KeyValue.remove('name')

# 剪贴板

from ascript.android.system import Clipboard

## 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "方法" \o "Direct link to 方法)

### 存数据至剪贴板[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "存数据至剪贴板" \o "Direct link to 存数据至剪贴板)

将数据存入剪贴板

* 函数

Clipboard.put(msg:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| msg | str | 是 | 要存入剪贴板的数据 |

* 示例

*# 将文本存入剪贴板*  
from ascript.android.system import Clipboard  
Clipboard.put("我是剪贴板中的文字")

### 读剪贴板数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "读剪贴板数据" \o "Direct link to 读剪贴板数据)

从剪贴板读取数据 支持跨应用

* 函数

Clipboard.get()

* 返回值

[str](F:/cs01/cs01/cj2/1extracted_div_content.html): 剪贴板中的数据

* 示例

*# 从剪贴板中读取信息*  
from ascript.android.system import Clipboard  
msg = Clipboard.get()  
print(msg)

# 系统通知

from ascript.android.system import Notification

## 通知监听[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "通知监听" \o "Direct link to 通知监听)

warning

调用该方法,会触发权限请求.请在设备上开启监听权限.

您也可以在APP设置中,开启系统监听.

获取手机SD卡目录

* 函数

Notification.listen(tunner):

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| tunner | fun(notiobj) | 是 | python函数,必须带有1个形参 |

* 示例

*# 监听系统通知*  
from ascript.android.system import Notification  
from android.app import Notification as AndroidNoti  
  
def noti\_callback(n):  
 print(n)  
 print("隶属包名",n.getPackageName())  
 print("标签",n.getTag())  
 print("ID",n.getId())  
 print("时间",n.getPostTime())  
 print("通知",n.getNotification())  
 *# 获取通知对象*  
 *# 获取 通知-额外数据对象*  
 android\_noti = n.getNotification()  
 bundle = android\_noti.extras  
 title = bundle.getString(AndroidNoti.EXTRA\_TITLE)  
 print("标题",title)  
 text = bundle.getString(AndroidNoti.EXTRA\_TEXT)  
 print("内容",text)  
 *# 获取通知意图*  
 intent = android\_noti.contentIntent  
 print(intent)  
  
Notification.listen(noti\_callback)

# Root指令

当您使用模拟器或Root的设备时,可以使用该指令

## 发送shell指令[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发送shell指令" \o "Direct link to 发送shell指令)

* 函数

system.shell(command: str, callback: ShellListener = None)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| command | str | 是 | Root指令 |
| callback | ShellListener接口 | 否 | 回调对象,请创建一个类继承 ascript.android.system.ShellListener |

* + ShellListener

class ShellListener(ABC):  
 @abstractmethod  
 def commandOutput(self, i: int, s: str):  
 *#当命令有输出值时*  
 pass  
  
 @abstractmethod  
 def commandTerminated(self, i: int, s: str):  
 *# 当命令被中断时*  
 pass  
  
 def commandCompleted(self, i: int, i1: int):  
 *# 当命令执行完毕时*  
 pass

* 示例

*# 修改分辨率*  
from ascript.android import system  
from ascript.android.system import ShellListener  
  
class L(ShellListener):  
 def commandOutput(self, i: int, s: str):  
 print('?',s)  
  
 def commandTerminated(self, i: int, s: str):  
 pass  
  
 def commandCompleted(self, i: int, i1: int):  
 pass  
  
  
res = system.shell('wm size reset',L())

# Shizuku指令

当使用Shizuku工具激活时.

我们可以在未越狱的设备上,模拟执行adb-shell指令.

*# 导包*  
from ascript.android import shizuku

## 发送shell指令[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发送shell指令" \o "Direct link to 发送shell指令)

shell 指令,其实是指adb指令

* 函数

shizuku.shell(command: Union[str, list])

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| command | Union[str, list] | 是 | shell指令,可以是一条指令,或多条指令,多条指令用数组传递 |

* 示例

*# 模拟点击home按键*  
from ascript.android import shizuku  
shizuku.shell("input keyevent 3")

# 使用插件

from ascript.android import plug

插件系统允许作者上传自己的工程作为插件存在, 其他开发者可以 **像导包一样,引入插件**, 使用插件中的方法

## 加载插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "加载插件" \o "Direct link to 加载插件)

* 函数

plug.load(plug\_and\_version:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| plug\_and\_version | str | 是 | 线上插件名称与版本,名称和版本之间用:连接, 也可以不写版本号(拉取最新版本) 如:"first","first:1.0" |

注意

在加载插件时,**建议写版本号**

否则,作者的版本更新可能会影响到您程序的运行.

* 示例

*# 加载线上插件,并导入插件*  
from ascript.android import plug  
plug.load("first")  
import first

## 导入插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导入插件" \o "Direct link to 导入插件)

当我们 **加载完毕插件以后** ,插件就作为我们的本地模块存在了.

我们可以通过 import 导入或 from 插件.子模块 import 子方法 这样的形式导入

* 示例

*# 我们试着加载 first 插件,并导入它的子模块方法,*  
*# 已知first插件包含user模块,user模块中存在两个方法 exit() login(name,pwd)*  
  
from ascript.android import plug  
*# 从线上加载模块*  
plug.load("first:2.0")  
  
*# 导入模块*  
from first import user  
  
*# 调用模块中的函数方法*  
user.login("自在老师","998")  
user.exit()

## 多文件中,插件如何加载?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "多文件中插件如何加载" \o "Direct link to 多文件中,插件如何加载?)

**我们只需要在 \_\_init\_\_.py** 中 使用 plug.load("") 一次 ,其他页面不需要再次加载,直接使用导包语句即可.

# 插件列表

列表中的插件,均为开发者上传.

其中部分插件可能包含收费服务,使用前务必了解详细.

## 插件列表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件列表-1" \o "Direct link to 插件列表)

Loading...

# 发布插件

我们开发的 **任意AS工程**,都可以当作一个插件发布.

## 导出工程为插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "导出工程为插件" \o "Direct link to 导出工程为插件)

任意工程都可以导出为插件.

## 上传发布插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "上传发布插件" \o "Direct link to 上传发布插件)

### 填写描述信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "填写描述信息" \o "Direct link to 填写描述信息)

[点这里也可前往上传插件页面](https://py.airscript.cn/admin/plug/list/create" \t "F:/cs01/cs01/cj2/_blank)

下面是一个填写信息的模版

### 发布[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发布" \o "Direct link to 发布)

### 更新插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "更新插件" \o "Direct link to 更新插件)

更新步骤

导出程序包

前往已上传的插件页面,点击编辑

重新上传新的插件包,更改版本号

点击提交即可

# Python部分

## 1.创建插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1创建插件" \o "Direct link to 1.创建插件)

在AS中,创建一个新的图色插件很容易,只需要继承 GPTask 即可.

继承后需要补充 name属性 , ui\_path属性 , run方法 的内容

*#自定义插件案例*  
from ascript.android.screen.gp import GPStack,GPTask,Result  
from ascript.android.system import R  
import numpy as np  
class MyPlug(GPTask):  
 name = "插件名称"  
 ui\_path = "插件UI路径" *# 这里使用的是相对路径,该UI路径下必须包含 index.html,这将作为插件的入口*  
  
 def run(self,cv\_image: np.ndarray,offset\_x:int=0,offset\_y:int=0,data=None) -> Result:  
 *# 在这里要写你的插件逻辑*  
 return Result(cv\_image,offset\_x,offset\_y,data)

### 插件名称[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件名称" \o "Direct link to 插件名称)

* **name** 字段

插件的名称,必须配置. 发布插件后,会显示在插件列表中.

### 插件UI路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件ui路径" \o "Direct link to 插件UI路径)

* **ui\_path**

制作插件UI是 图形处理插件中 比较繁琐的一部分内容. 其中涉及到 html,css,javascript 的知识. 同时我们也给你提供了相关js的API、

在这里,我们需要在 工程中 创建一个任意目录,**目录中必须包含 index.html**

我们只需要将这个 index.html 的 **上级目录相对路径配置给 ui\_path** 即可.

这里 我们推荐你使用 R.rel(\_\_file\_\_,"res/你的ui路径") 这种形式来填写路径.

#### 配置案例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "配置案例" \o "Direct link to 配置案例)

### 插件的核心实现[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件的核心实现" \o "Direct link to 插件的核心实现)

插件的核心,是实现 GPTask中的run方法

在run方法中,你可以绘制图片,或者根据图像算法得到一个结果.

def run(self,cv\_image: np.ndarray,offset\_x:int=0,offset\_y:int=0,data=None) -> Result:  
 *# 在这里要写你的插件逻辑*  
 return Result(cv\_image,offset\_x,offset\_y,data)

* 必填部分说明

| **字段或方法** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| cv\_image | np | 是 | 一张3通道的 cv numpy image |
| offset\_x,offset\_y | str | 是 | 上级插件处理图片后,返回的偏移量.  比如:上级插件裁剪了图片,会把裁剪区域距原图顶点的偏移量传入. |
| data | obj | 是 | 上级插件处理图片后,返回的数据 |
| Result | object | 是 | 当run方法内部,处理好图片,和数据后,需要传封装进 Result中返回 |

## 2.build.as配置插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2buildas配置插件" \o "Direct link to 2.build.as配置插件)

在 build.as文件中配置gp属性

gp是个json数组,我们需要写一个json对象,对象包含两个属性 id,des

{  
 "id":"这里写继承GPTask类的路径",  
 "des":"这是我的第一个图色插件"  
}

* 属性介绍

| **字段或方法** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| id | str | 是 | 插件的相对路径: 以 “.”开头, “.”代表当前工程根目录 如:你在工程根\_\_init\_\_.py 中有一个MyPlug插件,则可以写为 .MyPlug  如:你在工程下有一个user.py,在user.py中有一个MyPlug插件,则可以写为 .user.MyPlug |
| des | str | 是 | 插件功能简短的介绍 |

* **id**: 插件的路径

路径如何书写

* **des**: 插件的简短功能描述

### 配置案例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "配置案例-1" \o "Direct link to 配置案例)

# 调试

## 如何调试插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "如何调�试插件" \o "Direct link to 如何调试插件)

调试开发插件很简单哦

### 1.您已经创建了一个插件?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "1您已经创建了一个插件" \o "Direct link to 1.您已经创建了一个插件?)

您需在您的工程中, 创建一个插件继承GPTask

* 例如

### 2.增加调试路径[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "2增加调试路径" \o "Direct link to 2.增加调试路径)

* 点击插件列表中的更多

# UI部分

### UI显示在哪儿?[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "ui显��示在哪儿" \o "Direct link to UI显示在哪儿?)

## 创建UI目录[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "创建ui目录" \o "Direct link to 创建UI目录)

可以在工程下,任意目录创建 任意名称的目录

但目录下 **必须包含 index.html 页面 作为插件的入口**

然后将 **目录** 相对路径配置进您插件的 ui\_path 字段即可. 配置方法详见Python部分.

## js 方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "js-方法" \o "Direct link to js 方法)

我们已经为您封装好了,很多JavaScript API

可以获取当前用户括选的范围,获取获取当前用户取色的坐标RGB颜色,删除设备文件,拷贝文件. 插入图形处理流程,等操作

## js 回调[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "js-回调" \o "Direct link to js 回调)

当插件被加载,或用户操作后回调您插件中 js的方法.

#### 插件加载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "插件加载" \o "Direct link to 插件加载)

当插件被加载的时候调用该函数,一般用来初始化操作, 该方法在 document.ready 后执行,因此您可以安心在此方法中操作DOM元素

* 函数

var gp;  
var api;  
var task = {  
 data: {}, *//这里是要保存的相关数据*  
 params:"" *//这里是传递给Python的参数*  
}  
  
function on\_load(p\_gp,gp\_api) {  
 gp = p\_gp;  
 api = gp\_api;  
 *//在这里初始化 DOM,等操作*  
}

* 参数

| **名称** | **类型** | **说明** |
| --- | --- | --- |
| p\_gp | dict | 插件对象,包含插件id,name,等属性 |
| gp\_api | dict | api对象,所有图形处理API都通过该属性访问,建议赋值为全局变量 |

#### 重新修改数据[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "重新修改数据" \o "Direct link to 重新修改数据)

当插件加入流程处理以后,再次点击修改,回进入到对应插件.并携带data数据

* 函数

function on\_reload(p\_gp) {  
 *// 这里的 p\_gp 已经携带了 data数据. data数据 等于 api.stack.add 加入的 数据*  
}

* 参数

| **名称** | **类型** | **说明** |
| --- | --- | --- |
| p\_gp | dict | 插件对象,包含插件id,name, **data** 属性 |

#### 当括选范围[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当括选范围" \o "Direct link to 当括选范围)

当主图选中范围后,回触发该方法

* 函数

function on\_rect(rect) {  
   
}

* 参数

| **名称** | **类型** | **说明** |
| --- | --- | --- |
| rect | list | rect是一个数组,包含了 [left,tip,right,bottom] 4个属性 是一个矩形范围 |

#### 当按下取色按键[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当按下取色按键" \o "Direct link to 当按下取色按键)

当按下1-9数字按键,取色后,回调改方法

* 函数

function on\_color(key, color) {  
  
}

* 参数

| **名称** | **类型** | **说明** |
| --- | --- | --- |
| key | str | 按键值 |
| color | str | rgb颜色值 |

#### 当橡皮擦除后[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当橡皮擦除后" \o "Direct link to 当橡皮擦除后)

当橡皮擦除图片后,回调该方法

* 函数

function on\_wipe() {  
  
}

## js API[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "js-api" \o "Direct link to js API)

在回调函数 on\_load() 中我们可以拿到 api对象,所有的 **api方法都通过该对象调用**.

var gp;  
var api;  
var task = {  
 data: {}, *//这里是要保存的相关数据*  
 params:"" *//这里是传递给Python的参数*  
}  
  
function on\_load(p\_gp,gp\_api) {  
 gp = p\_gp;  
 api = gp\_api;  
 *//在这里初始化 DOM,等操作*  
}

### 文件处理[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "文件处理" \o "Direct link to 文件处理)

api.file.path

通过api 操控设备中的文件

#### 工作空间[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "工作空间" \o "Direct link to 工作空间)

| **属性** | **类型** | **说明** |
| --- | --- | --- |
| api.api.file.path.workspace | str | 工作空间 |

#### 获取文件url地址[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取文件url地址" \o "Direct link to 获取文件url地址)

通过文件存储路径,获取url类型的地址

api.file.url(file\_path)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| file\_path | str | 是 | 文件存储地址 |

* 返回值 文件的URL类型字符串

#### 读取文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "读取文件" \o "Direct link to 读取文件)

读取文件内容

api.file.read(file\_path,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| file\_path | str | 是 | 文件存储地址 |
| listener | function | 是 | 读取文件后的回调函数 |

#### 拷贝文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "拷贝文件" \o "Direct link to 拷贝文件)

拷贝文件

api.file.read(source\_file,target\_file,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| source\_file | str | 是 | 源文件地址 |
| target\_file | str | 是 | 目标文件地址 |
| listener | function | 是 | 拷贝结束后的回调函数 |

#### 获取文件树[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取文件树" \o "Direct link to 获取文件树)

传入一个存储路径文件夹地址,获取该目录下的所有文件列表

api.file.dir(dir\_path,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| dir\_path | str | 是 | 源文件夹地址 |
| listener | function | 是 | 得到结果后的回调函数 |

#### 重命名[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "重命名" \o "Direct link to 重命名)

传入一个文件的路径,重命名该文件

api.file.rname(file\_path,rname,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| file\_path | str | 是 | 源文件地址 |
| rname | str | 是 | 重命名的名称 |
| listener | function | 是 | 得到结果后的回调函数 |

#### 删除文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "删除文件" \o "Direct link to 删除文件)

删除该文件或文件夹

api.file.delete(file\_path,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| file\_path | str | 是 | 源文件地址 |
| listener | function | 是 | 得到结果后的回调函数 |

#### 写文件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "写文件" \o "Direct link to 写文件)

将内容写入文件

api.file.write(file\_path,content,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| file\_path | str | 是 | 源文件地址 |
| content | str | 是 | 写入的内容 |
| listener | function | 是 | 写入口的回调函数 |

### 图形处理栈[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图形处理栈" \o "Direct link to 图形处理栈)

图形处理栈中,包含关于图形处理插件的管理

api.stack

#### 当前处理任务信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当前处理任务信息" \o "Direct link to 当前处理任务信息)

* 字段

| **字段名** | **类型** | **说明** |
| --- | --- | --- |
| api.stack.info.name | str | 当前任务栈名称 |
| api.stack.info.path | str | 当前任务栈存储路径 |

#### 当前任务栈结果[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "当前任务栈结果" \o "Direct link to 当前任务栈结果)

图形处理任务,运行后产生的结果数据.

* 字段

| **字段名** | **类型** | **说明** |
| --- | --- | --- |
| api.stack.result | str | 当前任务栈名称 |

#### 加入任务栈[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "加入任务栈" \o "Direct link to 加入任务栈)

插入本插件到任务栈用

api.stack.add(data)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| data | str | 否 | 插件参数与数据,见下方示例 如果你需要给本插件传递python参数,需在数据对象中包含 parmas参数,该参数将原封不动的传递进python插件对象的init 初始化函数中 |

* 关于data 数据的约束

var task = {  
 data: {}, *//这里推荐您保存插件的处理数据,当用户二次修改时,可恢复html页面数据*  
 params:"" *//这是传递给插件的init函数,初始化参数.*  
}

#### 测试插件[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "测试插件" \o "Direct link to 测试插件)

临时将插件存放进任务栈中,进行测试.

api.stack.test(data,listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| data | str | 是 | 插件参数与数据,见下方示例 如果你需要给本插件传递python参数,需在数据对象中包含 parmas参数,该参数将原封不动的传递进python插件对象的init 初始化函数中 |
| listener | function | 是 | 测试完毕后的回调函数,可打印查看数据结构 |

* 关于data 数据的约束

var task = {  
 data: {}, *//这里推荐您保存插件的处理数据,当用户二次修改时,可恢复html页面数据*  
 params:"" *//这是传递给插件的init函数,初始化参数.*  
}

#### 通过相对路径获取文件信息[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "通过相对路径获取文件信息" \o "Direct link to 通过相对路径获取文件信息)

传入一个GP工程的相对路径,返回当前文件的url地址,和真实文件存储路径.

api.stack.file(rel\_path)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| rel\_path | str | 是 | 相对路径 如 "res/img/a.png" |

* 返回值 一个 字典对象,包含两个属性 path:当前文件的存储路径, url:当前文件的url地址

### 标记[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "标记" \o "Direct link to 标记)

在主图中绘制标记

api.marks

#### 增加点标记[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "增加点标记" \o "Direct link to 增加点标记)

在主图中绘制点标记

api.marks.point.add(id,xy\_list)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| id | str | 是 | 标记id,一般传数字标记 |
| xy\_list | array | 是 | 点的坐标,数组必须包含两个元素,代表x,y 如:[10,34] |

#### 清空点标记[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "清空点标记" \o "Direct link to 清空点标记)

清空主图中的所有点标记

api.marks.point.clear()

#### 增加矩形标记[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "增加矩形标记" \o "Direct link to 增加矩形标记)

在主图中绘制点矩形标记

api.marks.rect.add(id,rect)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| id | str | 是 | 标记id,一般传数字标记 |
| rect | list | 是 | 矩形的范围,数组必须包含4个元素,代表[left,top,right,bottom] 如:[10,20,50,60] |

#### 清空矩形标记[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "清空矩形标记" \o "Direct link to 清空矩形标记)

清空主图中的所有矩形标记

api.marks.rect.clear()

### 图像[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "图像" \o "Direct link to 图像)

对主图进行操作

#### 展示图片[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "展示图片" \o "Direct link to 展示图片)

让主图位 显示一张图

api.image.show(path)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| path | str | 是 | 图片的存储路径 |

#### 设置括选范围[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "设置括选范围" \o "Direct link to 设置括选范围)

让主图括选一个范围

api.image.set\_rect(rect)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| rect | list | 是 | 括选的范围数组,必须包含4个元素[left,top,right,bottom] 例如:[100,200,500,600] |

#### 获取括选范围[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取括选范围" \o "Direct link to 获取括选范围)

获取主图当前括选的范围

api.image.get\_rect()

* 返回值

一个数组,包含了4个元素,代表一个矩形范围

#### 裁剪一个小图[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "裁剪一个小图" \o "Direct link to 裁剪一个小图)

从主图中截取一张小图

api.image.crop(listener,p\_rect,target\_path)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| listener | function | 是 | 截图的回调函数 |
| p\_rect | list | 是 | 截图的范围数组,必须包含4个元素,例如[100,200,300,300] |
| target\_path | str | 是 | 截图后要保存的的文件存储路径 |

### 工程[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "工程" \o "Direct link to 工程)

获取设备的工程列表信息,其中包含了工程名称,存储路径地址等信息

#### 获取工程列表[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取工程列表" \o "Direct link to 获取工程列表)

获取已创建的工程列表

api.module.get\_all(listener)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| listener | function | 是 | 数据回调函数,可打印查看数据结构 |

### 系统[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "系统" \o "Direct link to 系统)

#### 存入剪贴板[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "存入剪贴板" \o "Direct link to 存入剪贴板)

存入电脑剪贴板,方便用户拷贝

api.sys.copy(msg)

* 参数

| **参数名** | **类型** | **必填** | **说明** |
| --- | --- | --- | --- |
| msg | str | 是 | 存入的数据信息 |

# 加载apk,arr,aab,jar

AS 可以将一个apk,arr,aab,jar 文件导入,加载并执行APK文件中的类

apk中包含arr?

如果您加载的apk中包含arr,arr中包含so 那么需要将arr中的so拷贝一份放在apk的libs中即可正常加载.

**如果apk,arr中包含so库,也会一并加载.**

整个过程通过**反射实现** 如果您掌握一点java的反射知识,那将更容易上手~

## 加载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "加载" \o "Direct link to 加载)

此方法会将apk,arr,jar,aab等文件加载进一个DexLoader中,方便后续通过反射拿到其中的类.

from ascript.android import plug

* 函数

plug.load\_apk(file\_path:str)

* 参数

| **参数** | **类型** | **是否必填** | **说明** |
| --- | --- | --- | --- |
| file\_path | str | 是 | apk文件的绝对路径 |

* 返回值

dalvik.system.DexClassLoader<?> //class 对象.

拿到该对象,我们可以通过java 反射,创建类的实例,并运行其内部方法等等..

* 示例:

from java.lang import Class  
from ascript.android import plug  
  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);  
*# 获取 获取公共(无参数)构造*   
constructor = clazz.getConstructor()  
*# 创建类的实例*  
instance = constructor.newInstance();  
*# 获取成员方法run*  
method = clazz.getMethod("run");  
*# 执行成员方法*  
method.invoke(instance);

## 反射参数与类型转换[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "反射参数与类型转换" \o "Direct link to 反射参数与类型转换)

### 反射参数类型[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "反射参数类型" \o "Direct link to 反射参数类型)

在通过反射获取java构造函数或某个方法时,通常需要需要写入的参数类型.

在python 中如何写入这些类型呢?

| **java参数类型** | **Python导入包** | **Python反射获取java方法参数类型写法** |
| --- | --- | --- |
| String | from java.lang import String | clazz.getMethod("java方法名称", String) |
| int | from java.lang import Integer | clazz.getMethod("java方法名称", Integer.TYPE) |
| float | from java.lang import Float | clazz.getMethod("java方法名称", Float.TYPE) |
| double | from java.lang import Double | clazz.getMethod("java方法名称", Double.TYPE) |
| long | from java.lang import Long | clazz.getMethod("java方法名称", Long.TYPE) |

### Python类型转Java实参类型[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "python类型转java实参类型" \o "Direct link to Python类型转Java实参类型)

一个Python的布尔型（bool）、整型（int）、浮点型（float）或字符串（str）对象可以直接传递给任何兼容的Java方法参数或字段。

然而，Java拥有比Python更多的基本数据类型，因此，当方法调用存在多个适用的整型或浮点型重载时，将使用参数类型最长的那个。类似地，当将一个单字符字符串传递给一个同时拥有String和char类型重载的方法时，将使用String类型的重载。

* 如果上述规则不能得到期望的结果，可以使用以下包装类来选择您想要使用的Java基本数据类型：

*# 转java boolean*  
class java.jboolean(value)  
  
*# 转java byte*  
class java.jbyte(value, truncate=False)  
  
*# 转java short*  
class java.jshort(value, truncate=False)  
  
*# 转java int*  
class java.jint(value, truncate=False)  
  
*# 转java long (python int 默认会转为java long)*  
class java.jlong(value, truncate=False)  
  
*# 转java float*  
class java.jfloat(value, truncate=False)  
  
*# 转java double*  
class java.jdouble(value, truncate=False)  
  
*# 转java jchar*  
class java.jchar(value)  
  
*# 转java jvoid 无法被实例化，但在定义静态代理时可以作为返回类型使用。*  
class java.jvoid

数值类型包装器接受一个可选的truncate参数。如果设置了该参数，那么给定值的任何多余的高位将被丢弃，就像在Java中进行类型转换一样。否则，向包装器类传递一个超出范围的值将导致OverflowError。

当使用这些包装器时，将应用Java的重载解析规则来处理被包装的参数。例如，一个jint只适用于Java的int或更大的类型，并且会使用最短的可适用重载。

* 案例

from java import jint,jboolean,jlong,jdouble  
  
a = jint(10)  
  
b = jboolean(True)  
  
c = jlong(10)  
  
d = jdouble(10.2222)

## 获取构造函数[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取构造函数" \o "Direct link to 获取构造函数)

java 创建一个类实例的函数

java 因为参数的不同,可能会有多个构造函数.

* 先获取某个apk下的java类实例

*# 获取 Apk中的类*  
from java.lang import Class  
from ascript.android import plug  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);

from java.lang import Class, String, Float, Integer

### 获取无参构造与创建实例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取无参构造与创建实例" \o "Direct link to 获取无参构造与创建实例)

from java.lang import Class  
from ascript.android import plug  
  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);  
*# 获取 获取公共(无参数)构造*   
constructor = clazz.getConstructor()  
*# 创建类的实例*  
instance = constructor.newInstance();

### 获取有参构造函数[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取有参构造函数" \o "Direct link to 获取有参构造函数)

from java.lang import Class  
from ascript.android import plug  
from java.lang import String,Integer  
from java import jfloat, jint, jarray  
  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);  
*# 获取 获取公共(无参数)构造*   
constructor = clazz.getDeclaredConstructor(String, Integer.TYPE)  
*# 创建类的实例*  
instance = constructor.newInstance("张三",jint(18));

## 获取属性与方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取属性与方法" \o "Direct link to 获取属性与方法)

静态属性,成员属性,静态方法,成员方法 的获取方式一致.

**区别在于执行该方法时,传入的**第一个参数.

如果第一个参数传入为None,则代表获取静态属性或执行静态方法.

如果第一个参数传入为类实例对象,则代表获取成员属性或成员方法.

### 获取-某个属性或方法[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取-某个属性或方法" \o "Direct link to 获取-某个属性或方法)

*# 获取 Apk中的类*  
from java.lang import Class  
from ascript.android import plug  
from java.lang import String,Integer  
from java import jfloat, jint, jarray  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);

* 假如方法和属性是静态的,执行案例

*# 反射某个静态方法,并执行. 静态方法的第一个参数为None(代表无实例)*  
*# 这里的反射类型,请见 #反射参数与类型转换*  
clazz.getDeclaredMethod("save",String,Integer.TYPE)  
clazz.invoke(None,"张三",jint(18))  
*# 反射某个静态属性*  
static\_field = clazz.getDeclaredField("name")  
name\_val = static\_field.get(None)

* 假如方法和属性为对象的,执行案例

*# 获取 获取公共(无参数)构造,创建类的实例*  
constructor = clazz.getConstructor()  
instance = constructor.newInstance()  
*# 反射某个静态方法,并执行. 成员方法或属性,第一个参数为对象实例*  
*# 这里的反射类型,请见 #反射参数与类型转换*  
clazz.getDeclaredMethod("save",String,Integer.TYPE)  
clazz.invoke(instance,"张三",jint(18))  
*# 反射某个静态属性*  
static\_field = clazz.getDeclaredField("name")  
name\_val = static\_field.get(instance)

### 获取-所有属性案例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取-所有属性案例" \o "Direct link to 获取-所有属性案例)

*# Method[] methods = clazz.getDeclaredFields();*  
from java.lang import Class  
from ascript.android import plug  
from java.lang import String,Integer  
from java import jfloat, jint, jarray  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);  
  
*# 获取类中的所有静态属性,返回值为数组[属性字段,属性字段]*  
fields = clazz.getDeclaredFields()  
  
*# 将属性数组转换为一个 字典{属性名:属性对象,....},方便后期使用*  
py\_fields = {}  
for field in fields:  
 py\_fields[field.getName()] = field

* 假如类中有一个name 静态属性,执行案例

name = field["name"].get(None)

* 假如类中有一个name 成员属性,执行案例

*# 获取 获取公共(无参数)构造,创建类的实例*  
constructor = clazz.getConstructor()  
instance = constructor.newInstance()  
name = field["name"].get(instance)  
  
*# 打印获取到的name属性*  
pritn(name)

### 获取-所有方法案例[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "获取-所有方法案例" \o "Direct link to 获取-所有方法案例)

*# Method[] methods = clazz.getDeclaredMethods();*  
from java.lang import Class  
from ascript.android import plug  
from java.lang import String,Integer  
from java import jfloat, jint, jarray  
*# 加载apk,获取 DexClassLoader 对象*  
clazz\_loader = plug.load\_apk("sdcard/my.apk")  
*# 获取 Apk中的类*  
clazz = Class.forName("com.aojoy.apkplug.TestPlug", True, clazz\_loader);  
  
*# 获取类中的所有静态方法,返回值为数组[方法,方法]*  
methods = clazz.getDeclaredMethods()  
  
*# 将方法转换为一个 字典{方法名:方法,....},方便后期使用*  
py\_methods = {}  
for m in methods:  
 py\_methods[m.getName()] = m

* 假如类中有一个静态save(String a,int b)方法,执行案例

py\_methods["save"].invoke(None,"张三",jint(18))

* 假如类中有一个成员save(String a,int b)方法,执行案例

*# 获取 获取公共(无参数)构造,创建类的实例*  
constructor = clazz.getConstructor()  
instance = constructor.newInstance()  
py\_methods["save"].invoke(instance,"张三",jint(18))

# 发布与打包程序

将研发完成的小程序上线,并打包为可独立运行的App文件

## 发布[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "发布" \o "Direct link to 发布)

### 上传[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "上传" \o "Direct link to 上传)

* 1. 在代码编辑器中导出后缀名为as的工程包
  2. 打开 [开发者后台](http://py.airscript.cn/admin/apply/list" \t "F:/cs01/cs01/cj2/_blank),进入小程序列表
  3. 点击新增,填写资料,提交发布
  4. 在Airscript App中 通过id检索

上传后,在小程序列表中 会有对应小程序ID

我们可以在Airscript首页通过这个ID搜索并运行小程序.

### 打包独立APP[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "打包独立app" \o "Direct link to 打包独立APP)

* 1. 您需要发布一款小程序才可打包
  2. 在 [开发者后台](http://py.airscript.cn/admin/apply/list" \t "F:/cs01/cs01/cj2/_blank) 中找到一款发布的小程序,点击后方打包小程序
  3. 填写打包信息,点击开始打包等待完成即可.
  + 打包需要10分钟左右,可以离开界面

# HID软件

部分APP拦截屏蔽无障碍点击动作,这时候你可能需要HID工具辅助执行动作, **HID点击,滑动,输入,不可被拦截屏蔽**.

Hid(虚拟外设动作),需要电脑端连接手机,**更真实的模拟手指点击**.

理论情况下,一台电脑可以连接120台移动设备

## Windows版下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "windows版下载" \o "Direct link to Windows版下载)

Windows 软件

**需先安装usb驱动,后安装客户端程序**

* [win-usb驱动](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/libusbK-inf-wizard.exe" \t "F:/cs01/cs01/cj2/_blank)
* [win-64客户端](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/AirScript-win-x64-vpn.exe.zip" \t "F:/cs01/cs01/cj2/_blank)

## Mac版下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "mac版下载" \o "Direct link to Mac版下载)

Mac 软件

**需先安装usb驱动,后安装客户端程序**

*#驱动安装*  
brew install libusb

* [mac-64客户端](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/AirScript-mac-x64-vpn.tar.gz" \t "F:/cs01/cs01/cj2/_blank)
* [mac-arm64客户端](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/AirScript-mac-arm-vpn.tar.gz" \t "F:/cs01/cs01/cj2/_blank)

## Linux版下载[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "linux版下载" \o "Direct link to Linux版下载)

Linux 软件

**驱动请自行搜索**

* [linux-64客户端](https://airscript.oss-cn-hangzhou.aliyuncs.com/hid/ASHID-linux-x64" \t "F:/cs01/cs01/cj2/_blank)

# 投屏软件

为何一些投屏软件开了会影响AS?

由于AS app 包含图色检索,需要屏幕图色权限,因此会和一些投屏软件抢占图色权限.

因此需要通过 **adb权限来投屏幕**,同时这种方案延迟更低,稳定性更强.

## Windows 用户[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "windows-用户" \o "Direct link to Windows 用户)

Windows投屏步骤

1. [下载Windows 端的投屏软件](https://nsres.oss-cn-hangzhou.aliyuncs.com/sharescreen/scrcpy-win64-v1.24.zip" \t "F:/cs01/cs01/cj2/_blank)
2. 手机打开开发者模式,并通过USB线插入电脑
3. 打开已下载的 scrcpy.exe 即可投屏.

## Mac 用户[​](F:/cs01/cs01/cj2/1extracted_div_content.html" \l "mac-用户" \o "Direct link to Mac 用户)

苹果电脑投屏步骤

1. 确认您的电脑已经安装了HomeBrew
2. 安装 scrcpy

brew install scrcpy

1. 在电脑上安装 ADB 根据不同版本，使用不同命令安装 ADB

# Homebrew >= 2.6.0  
brew install --cask android-platform-tools  
  
# Homebrew < 2.6.0  
brew cask install android-platform-tools

1. 启动 scrcpy 如果以上都搞定，用数据线连接电脑和手机，现在来启动软件 scrcpy，很简单，在终端输入

scrcpy