JavaScript 对指令字母的大小写是敏感的，应该注意。

Keys

按键模拟部分提供了一些模拟物理按键的全局函数，包括 Home、音量键、照相键等，首字母小写的函数依赖于无障碍服务，首字母大写的函数都依赖于 root 权限。执行此类函数时，如果没有 root 权限，则函数执行后没有效果，并会在控制台输出一个警告。

---

back()

返回<boolean>

模拟按下返回键。返回是否执行成功。

home()

返回<boolean>

模拟按下 Home 键。返回是否执行成功。

recents()

返回<boolean>

显示最近任务。返回是否执行成功。

notifications()

返回<boolean>

拉出通知栏。返回是否执行成功。

quickSettings()

返回<boolean>

显示快速设置(下拉通知栏到底)。返回是否执行成功。

powerDialog()

返回<boolean>

弹出电源键菜单。返回是否执行成功。

splitScreen()

返回<boolean>

分屏。返回是否执行成功。需要系统自身功能的支持。

以上函数依赖于无障碍服务。

以下函数依赖于 root 权限。

Home()

模拟按下 Home 键。

Menu()

模拟按下菜单键。

Camera()

模拟按下照相键。

Back()

模拟按下返回键。

Power()

模拟按下电源键。

OK()

模拟按下物理按键确定。

VolumeUp()

按下音量上键。

Up()

模拟按下物理按键上。

Left()

模拟按下物理按键左。

VolumeDown()

按键音量上键。

Down()

模拟按下物理按键下。

Right()

模拟按下物理按键右。

Text(text)

text<string>要输入的文字，只能为英文或英文符号(**不能输入汉字**)

例如 `Text("aaa");`

KeyCode(code)

code<number>要按下的按键的数字代码或名称。模拟物理按键。

例如 `KeyCode(29)`和 `KeyCode("KEYCODE_A")`是按下 A 键。

KeyCode 对照表

名称	数字代码	名称	数字代码
KEYCODE_MENU	1	KEYCODE_O	43
KEYCODE_SOFT_RIGHT	2	KEYCODE_P	44
KEYCODE_HOME	3	KEYCODE_Q	45
KEYCODE_BACK	4	KEYCODE_R	46
KEYCODE_CALL	5	KEYCODE_S	47
KEYCODE_ENDCALL	6	KEYCODE_T	48
KEYCODE_0	7	KEYCODE_U	49
KEYCODE_1	8	KEYCODE_V	50
KEYCODE_2	9	KEYCODE_W	51
KEYCODE_3	10	KEYCODE_X	52
KEYCODE_4	11	KEYCODE_Y	53
KEYCODE_5	12	KEYCODE_Z	54
KEYCODE_6	13	KEYCODE_COMMA	55
KEYCODE_7	14	KEYCODE_PERIOD	56
KEYCODE_8	15	KEYCODE_ALT_LEFT	57
KEYCODE_9	16	KEYCODE_ALT_RIGHT	58
KEYCODE_STAR	17	KEYCODE_SHIFT_LEFT	59
KEYCODE_POUND	18	KEYCODE_SHIFT_RIGHT	60
KEYCODE_DPAD_UP	19	KEYCODE_TAB	61
KEYCODE_DPAD_DOWN	20	KEYCODE_SPACE	62
KEYCODE_DPAD_LEFT	21	KEYCODE_SYM	63
KEYCODE_DPAD_RIGHT	22	KEYCODE_EXPLORER	64
KEYCODE_DPAD_CENTER	23	KEYCODE_ENVELOPE	65
KEYCODE_VOLUME_UP	24	KEYCODE_ENTER	66
KEYCODE_VOLUME_DOWN	25	KEYCODE_DEL	67
KEYCODE_POWER	26	KEYCODE_GRAVE	68
KEYCODE_CAMERA	27	KEYCODE_MINUS	69
KEYCODE_CLEAR	28	KEYCODE_EQUALS	70
KEYCODE_A	29	KEYCODE_LEFT_BRACKET	71
KEYCODE_B	30	KEYCODE_RIGHT_BRACKET	72
KEYCODE_C	31	KEYCODE_BACKSLASH	73
KEYCODE_D	32	KEYCODE_SEMICOLON	74
KEYCODE_E	33	KEYCODE_APOSTROPHE	75
KEYCODE_F	34	KEYCODE_SLASH	76
KEYCODE_G	35	KEYCODE_AT	77
KEYCODE_H	36	KEYCODE_NUM	78
KEYCODE_I	37	KEYCODE_HEADSETHOOK	79
KEYCODE_J	38	KEYCODE_FOCUS	80
KEYCODE_K	39	KEYCODE_PLUS	81
KEYCODE_L	40	KEYCODE_MENU	82
KEYCODE_M	41	KEYCODE_NOTIFICATION	83
KEYCODE_N	42	KEYCODE_SEARCH	84

全局变量与函数

全局变量和函数在所有模块中均可使用。 但以下变量的作用域只在模块内，详见 module 模块

exports

module

require() 以下的对象是特定于 Auto.js 的。 有些内置对象是 JavaScript 语言本身的一部分，它们也是全局的。

一些模块中的函数为了使用方便也可以直接全局使用，这些函数在此不再赘述。例如 timers 模块的 setInterval, setTimeout 等函数。

---

sleep(n)

n <number> 毫秒数

暂停运行 n 毫秒的时间。1 秒等于 1000 毫秒。

```
//暂停 5 毫秒
sleep(5000);
```

exit()

立即停止脚本运行。

立即停止是通过抛出 ScriptInterruptedException 来实现的，因此如果用 try...catch 把 exit()函数的异常捕捉，则脚本不会立即停止，仍会运行几行后再停止。

waitForActivity(activity[, period = 200])

activity Activity 名称

period 轮询等待间隔（毫秒）

等待指定的 Activity 出现，period 为检查 Activity 的间隔。

waitForPackage(package[, period = 200])

package 包名

period 轮询等待间隔（毫秒）

等待指定的应用出现。例如

```
waitForPackage("com.tencent.mm")
```

为等待当前界面为微信。

currentActivity()

返回 <string>

返回最近一次监测到的正在运行的 Activity 的名称，一般可以认为就是当前正在运行的 Activity 的名称。

此函数依赖于无障碍服务，如果服务未启动，则抛出异常并提示用户启动。

currentPackage()

返回 <string>

返回最近一次监测到的正在运行的应用的包名，一般可以认为就是当前正在运行的应用的包名。

此函数依赖于无障碍服务，如果服务未启动，则抛出异常并提示用户启动。

setClip(text)

text <string> 文本

设置剪贴板内容。此剪贴板即系统剪贴板，在一般应用的输入框中"粘贴"既可使用。

```
setClip("剪贴板文本");
```

getClip()

返回 <string>

返回系统剪贴板的内容。

```
toast("剪贴板内容为:" + getClip());
```

toast(message)

message <string> 要显示的信息

以气泡显示信息 message 几秒。(具体时间取决于安卓系统，一般都是 2 秒)

高版本的安卓系统（Android 6 或 7 之后的系统）由于系统对部分 toast 消息进行了过滤，所以导致在其他应用中无法显示 toast 提示。

注意，信息的显示是"异步"执行的，并且，不会等待信息消失程序才继续执行。如果在循环中执行该命令，可能出现脚本停止运行后仍然有不断的气泡信息出现的情况。 例如:

```
for(var i = 0; i < 100; i++){
    toast(i);
}
```

运行这段程序以后，会很快执行完成，且不断弹出消息，在任务管理中关闭所有脚本也无法停止。 要保证气泡消息才继续执行可以用:

```
for(var i = 0; i < 100; i++){
    toast(i);
    sleep(2000);
}
```

或者修改 toast 函数:

```
var _toast_ = toast;
toast = function(message){
    _toast_(message);
    sleep(2000);
}
for(var i = 0; i < 100; i++){
    toast(i);
}
```

toastLog(message)

message <string> 要显示的信息

相当于 toast(message);log(message)。显示信息 message 并在控制台中输出。参见 console.log。

random(min, max)

min <number> 随机数产生的区间下界

max <number> 随机数产生的区间上界

返回 <number>

返回一个在[min...max]之间的随机数。例如 random(0, 2)可能产生 0, 1, 2。

random()

返回 <number>

返回在[0, 1)的随机浮点数。

context

全局变量。一个 android.content.Context 对象。

注意该对象为 ApplicationContext，因此不能用于界面、对话框等的创建。

App

app 模块提供一系列函数，用于使用其他应用、与其他应用交互。例如发送意图、打开文件、发送邮件等。

同时提供了方便的进阶函数 startActivity 和 sendBroadcast，用他们可完成 app 模块没有内置的和其他应用的交互。

app.launchApp(appName)

appName <string> 应用名称

通过应用名称启动应用。如果该名称对应的应用不存在，则返回 false；否则返回 true。如果该名称对应多个应用，则只启动其中某一个。

该函数也可以作为全局函数使用。

launchApp("Auto.js");

app.launch(packageName)

packageName <string> 应用包名

通过应用包名启动应用。如果该包名对应的应用不存在，则返回 false；否则返回 true。

该函数也可以作为全局函数使用。

//启动微信

launch("com.tencent.mm");

app.launchPackage(packageName)

packageName <string> 应用包名

相当于 app.launch(packageName)。

app.getPackageName(appName)

appName <string> 应用名称

获取应用名称对应的已安装的应用的包名。如果该找不到该应用，返回 null；如果该名称对应多个应用，则只返回其中某一个的包名。

该函数也可以作为全局函数使用。

var name = getPackageName("QQ"); //返回 "com.tencent.mobileqq"

app.getAppName(packageName)

packageName <string> 应用包名

获取应用包名对应的已安装的应用的名称。如果该找不到该应用，返回 null。

该函数也可以作为全局函数使用。

var name = getAppName("com.tencent.mobileqq"); //返回"QQ"

app.openAppSetting(packageName)

packageName <string> 应用包名

打开应用的详情页(设置页)。如果找不到该应用，返回 false；否则返回 true。

该函数也可以作为全局函数使用。

app.uninstall(packageName)

packageName <string> 应用包名

卸载应用。执行后会会弹出卸载应用的提示框。如果该包名的应用未安装，由应用卸载程序处理，可能弹出"未找到应用"的提示。

//卸载 QQ

app.uninstall("com.tencent.mobileqq");

app.viewFile(path)

path <string> 文件路径

用其他应用查看文件。文件不存在的情况由查看文件的应用处理。

如果找不出可以查看该文件的应用，则抛出 ActivityNotException。

```
//查看文本文件
app.viewFile("/sdcard/1.txt");
```

app.editFile(path)

path <string> 文件路径

用其他应用编辑文件。文件不存在的情况由编辑文件的应用处理。

如果找不出可以编辑该文件的应用，则抛出 ActivityNotException。

```
//编辑文本文件
app.editFile("/sdcard/1.txt/);
```

app.openUrl(url)

url <string> 网站的 Url，如果不以"http://"或"https://"开头则默认是"http://"。

用浏览器打开网站 url。

如果没有安装浏览器应用，则抛出 ActivityNotException。

app.sendEmail(options)

options <Object> 发送邮件的参数。包括:

email <string> | <Array> 收件人的邮件地址。如果有多个收件人，则用字符串数组表示

cc <string> | <Array> 抄送收件人的邮件地址。如果有多个抄送收件人，则用字符串数组表示

bcc <string> | <Array> 密送收件人的邮件地址。如果有多个密送收件人，则用字符串数组表示

subject <string> 邮件主题(标题)

text <string> 邮件正文

attachment <string> 附件的路径。

根据选项 options 调用邮箱应用发送邮件。这些选项均是可选的。

如果没有安装邮箱应用，则抛出 ActivityNotException。

```
//发送邮件给 10086@qq.com 和 10001@qq.com。
app.sendEmail({
    email: ["10086@qq.com", "10001@qq.com"],
    subject: "这是一个邮件标题",
    text: "这是邮件正文"
});
```

app.startActivity(name)

name <string> 活动名称，可选的值为:

console 日志界面

settings 设置界面

启动 Auto.js 的特定界面。该函数在 Auto.js 内运行则会打开 Auto.js 内的界面，在打包应用中运行则会打开打包应用的相应界面。

```
app.startActivity("console");
```

进阶：意图 Intent

Intent(意图) 是一个消息传递对象，您可以使用它从其他应用组件请求操作。尽管 Intent 可以通过多种方式促进组件之间的通信，但其基本用例主要包括以下三个：

启动活动(Activity)： Activity 表示应用中的一个"屏幕"。例如应用主入口都是一个 Activity，应用的功能通常也以 Activity 的形式独立，例如微信的主界面、朋友圈、聊天窗口都是不同的 Activity。通过将 Intent 传递给 startActivity()，您可以启动新的 Activity 实例。Intent 描述了要启动的 Activity，并携带了任何必要的数据。

启动服务(Service)： Service 是一个不使用用户界面而在后台执行操作的组件。通过将 Intent 传递给 startService()，您可以启动服务执行一次性操作（例如，下载文件）。Intent 描述了要启动的服务，并携带了任何必要的数据。

传递广播： 广播是任何应用均可接收的消息。系统将针对系统事件（例如：系统启动或设备开始充电时）传递各种广播。通过将 Intent 传递给 sendBroadcast()、sendOrderedBroadcast() 或 sendStickyBroadcast()，您可以将广播传递给其他应用。

本模块提供了构建 Intent 的函数(app.intent()), 启动 Activity 的函数 app.startActivity(), 发送广播的函数 app.sendBroadcast()。使用这些方法可以用来方便的调用其他应用。例如直接打开某个 QQ 号的个人卡片页，打开某个 QQ 号的聊天窗口等。

app.startActivity(options)

options <Object> 选项

根据选项构造一个 Intent，并启动该 Activity。

app.sendBroadcast(options)

options <Object> 选项

根据选项构造一个 Intent，并发送该广播。

app.intent(options)

options <Object> 选项，包括：

action <string> 意图的 Action，指意图要完成的动作，是一个字符串常量，比如"android.intent.action.SEND"。当 action 以"android.intent.action"开头时，可以省略前缀，直接用"SEND"代替。常见的 action 参见常用的意图动作。

type <string> 意图的 MimeType，表示和该意图直接相关的数据的类型，比如"text/plain"为纯文本类型。

data <string> 意图的 Data，表示和该意图直接相关的数据，是一个 Uri，可以是文件路径或者 Url 等。例如要打开一个文件, action 为"android.intent.action.VIEW", data 为"file:///sdcard/1.txt"。

category <Array> 意图的类别。比较少用。

packageName <string> 目标包名

className <string> 目标 Activity 或 Service 等组件的名称

extras <Object> 以键值对构成的这个 Intent 的 Extras(额外信息)。提供该意图的其他信息，例如发送邮件时的邮件标题、邮件正文。

根据选项，构造一个意图 Intent 对象。

例如：//打开应用来查看图片文件

```
var i = app.intent({
        action: "VIEW",
        type: "image/png",
        data: "file:///sdcard/1.png"
});
app.startActivity(i);
```