**Android模拟产生事件**

05 January 2015 on [Android](https://www.race604.com/tag/android/)

我们有时候需要使用程序产生一些输入事件。例如一些“丧心病狂”的App，让你发疯的狂点屏幕来抽奖，程序来帮你，包你中奖。当然，本文并不是为了来作弊，而是开发过程中的一些实用工具。

给系统模拟注入输入事件有如下几种方式：

**1. 使用shell命令**

Android中自带一个input工具，使用方法如下：

adb shell #进入系统

输入如下命令：

input keyevent KEYCODE\_BACK 或者 input keyevent 3 #模拟按返回键

input keyevent KEYCODE\_HOME #模拟按Home键

上面的命令就给Android系统输入了相应的按键，使用方法是input keyevent <按键名或者按键值>。按键名或者值可以从[官方文档](http://developer.android.com/reference/android/view/KeyEvent.html)查询，只要是以KEYCODE\_开头的都可以。

如果系统的焦点在输入框内，还可以直接输入文本：

input text hello

直接在输入框内输入*hello*。输入的文本不能带空格，也不能是中文，真是很遗憾。

还可以直接输入点击屏幕的事件，模拟点击屏幕

input tap 100 200 #在屏幕坐标(100, 200)处点击

使用方法是input tap <x> <y>。坐标从屏幕左上角开始算。 也可以模拟长按，滑动等等，详细的用法如下：

Usage: input [<source>] <command> [<arg>...]

The sources are:

trackball

joystick

touchnavigation

mouse

keyboard

gamepad

touchpad

dpad

stylus

touchscreen

The commands and default sources are:

text <string> (Default: touchscreen)

keyevent [--longpress] <key code number or name> ... (Default: keyboard)

tap <x> <y> (Default: touchscreen)

swipe <x1> <y1> <x2> <y2> [duration(ms)] (Default: touchscreen)

press (Default: trackball)

roll <dx> <dy> (Default: trackball)

tmode <tmode>

**2. 使用Instrumentation**

Instrumentation本身是Android用来做测试的工具，可以通过它监测系统与应用程序之间的交互。详情可以参考官方文档[Android测试](http://developer.android.com/tools/testing/index.html)。我们这里只关注怎么使用Instrumentation产生发送按键或者触屏事件。

发送按键：

Instrumentation mInst = new Instrumentation();

mInst.sendKeyDownUpSync(KeyEvent.KEYCODE\_CAMERA);

sendKeyDownUpSync()从名字上就能看出，这是同步发送一个按键按下和弹起事件。

发送触屏事件：

Instrumentation mInst = new Instrumentation();

mInst.sendPointerSync(MotionEvent.obtain(SystemClock.uptimeMillis(),

SystemClock.uptimeMillis(), MotionEvent.ACTION\_DOWN, x, y, 0);

mInst.sendPointerSync(MotionEvent.obtain(SystemClock.uptimeMillis(),

SystemClock.uptimeMillis(), MotionEvent.ACTION\_UP, x, y, 0);

同样，sendPointerSync()是同步发送发送触摸事件。

与Shell工具一样，还有类似sendStringSync()发送文本，sendTrackballEventSync()发送轨迹球事件等方法。

注意，使用上面的方法，在AndroidManifast.xml中申明如下权限：

<uses-permission android:name="android.permission.INJECT\_EVENTS"/>

遗憾的是android.permission.INJECT\_EVENTS是一个System权限,折腾起来有点麻烦，可以参考[这里](http://stackoverflow.com/a/7328055/1260562)。

**3. 使用Android内部API**

在Android系统中，有些内部的API提供注入事件的方法。因为是内部API，在不同版本上可能变化比较大。使用如果想在普通App中使用，可能需要通过反射机制来调用。

在Android API 16之前，WindownManager有相应的方法提供注入事件的方法，如下：

IBinder wmbinder = ServiceManager.getService("window");

IWindowManager wm = IWindowManager.Stub.asInterface(wmbinder); //pointer

wm.injectPointerEvent(myMotionEvent, false); //key

wm.injectKeyEvent(new KeyEvent(KeyEvent.ACTION\_DOWN, KeyEvent.KEYCODE\_A), false);

wm.injectKeyEvent(new KeyEvent(KeyEvent.ACTION\_UP, KeyEvent.KEYCODE\_A), false); //trackball

wm.injectTrackballEvent(myMotionEvent, false);

在API 15之后，引入了InputManager，把上面的哪些injectXXXEvent()方法从WindowManager中移除了。使用方法类似：

IBinder imBinder = ServiceManager.getService("input");

IInputManager im = IInputManager.Stub.asInterface(imBinder);

//inject key event

final KeyEvent keyEvent = new KeyEvent(downTime, eventTime, action,

code, repeatCount, metaState, deviceId, scancode,

flags | KeyEvent.FLAG\_FROM\_SYSTEM |KeyEvent.FLAG\_KEEP\_TOUCH\_MODE | KeyEvent.FLAG\_SOFT\_KEYBOARD,

source);

event.setSource(InputDevice.SOURCE\_ANY)

im.injectInputEvent(keyEvent, InputManager.INJECT\_INPUT\_EVENT\_MODE\_WAIT\_FOR\_FINISH);

//inject pointer event

motionEvent.setSource(InputDevice.SOURCE\_TOUCHSCREEN);

im.injectInputEvent(motionEvent, InputManager.INJECT\_INPUT\_EVENT\_MODE\_WAIT\_FOR\_FINISH);

从API 16开始，[InputManager](http://developer.android.com/reference/android/hardware/input/InputManager.html)就成了一个公开的类了，可以通过如下方法获得InputManager实例：

InputManager im = (InputManager) getSystemService(Context.INPUT\_SERVICE);

注意，使用injectEvent()同样需要申明android:name="android.permission.INJECT\_EVENTS"权限。

**4. 使用Nativ/JNI调用**

察看Android设备的/dev/input/目录下的设备：

shell@mako:/dev/input $ ll

crw-rw---- root input 13, 64 2013-08-11 18:00 event0

crw-rw---- root input 13, 65 2013-08-11 18:00 event1

crw-rw---- root input 13, 66 2013-08-11 18:00 event2

crw-rw---- root input 13, 67 2013-08-11 18:00 event3

crw-rw---- root input 13, 68 2013-08-11 18:00 event4

crw-rw---- root input 13, 69 2013-08-11 18:00 event5

可以看到有一些输入设备节点，同时也提供了一些shell工具来操作这些设备，例如上面第1节中提到的input命令，另外还有getevent和sendevent工具分别来监听和发送事件。这些方法，都可以通过JNI的方式调用。这里需要注意的时间eventX设备都是input的用户组，要直接使用，需要root设备。

特别的是，这里有一个开源项目[android-event-injector](https://code.google.com/p/android-event-injector/)，使用JNI方法注入事件。当然设备需要root。

参考文章： 1 <http://blog.djodjo.org/?p=628>   
2 <http://www.pocketmagic.net/2012/04/injecting-events-programatically-on-android/#.VKn42M2UfCI>