chen825919148的专栏

个人资料

chen825919148

访问： 1141486次
积分： 11241
等级： BLOG 7
排名： 第1456名

原创： 58篇 转载： 456篇
译文： 0篇 评论： 118条

文章搜索

android测试之getevent/sendevent

2014-01-24 12:51 7116人阅读 评论(0)

分类： Android开发（135）

关于在android平台上捕获事件的资料互联网上已经铺天盖地，但个人觉得可用性都不太大，大部仅

针对特定设备，而对于其他设备引导性不强，故整理处本文，望对初学者有个有力的帮助：

getevent 旨在获取android设备的事件信息，具体参考详细用法(本人亦初学者一枚，[illegible])

sendevent 则可以向设备发送模拟事件，其中包括touch和keypress

详细用法如下：

```
源码 复制 打印 ?
01. Usage: getevent [-t] [-n] [-s switchmask] [-S] [-v [mask]] [-d] [-p] [-i] [-l] [-q] [-
c count] [-r] [device]
02.     -t: show time stamps
03.     -n: don't print newlines
04.     -s: print switch states for given bits
05.     -S: print all switch states
06.     -v: verbosity mask (errs=1, dev=2, name=4, info=8, vers=16, pos. events=32,
props=64)
```

关闭

```
    -d: show HID descriptor, if available
    -p: show possible events (errs, dev, name, pos. events)
    -i: show all device info and possible events
    -l: label event types and names in plain text
    -q: quiet (clear verbosity mask)
    -c: print given number of events then exit
    -r: print rate events are received

Usage: sendevent <device> <type> <code> <value>
```

首先，adb shell进入android设备，运行getevent命令得到如下信息，为各类事件的驱动设备，每部

硬件可能都不一样，可以说无规律可循，个人觉得掠过吧

源码 复制 打印 ?

```
add device 1: /dev/input/event1
  name:     "pmic8xxx pwrkey"
add device 2: /dev/input/event3
  name:     "apq8064-tabla-snd-card Headset Jack"
add device 3: /dev/input/event2
  name:     "apq8064-tabla-snd-card Button Jack"
add device 4: /dev/input/event5
  name:     "gpio-keys"
could not get driver version for /dev/input/mice, Not a
add device 5: /dev/input/event0
  name:     "atmel_mxt_ts"
add device 6: /dev/input/event4
  name:     "mhl_rcp"
```

在点击设备屏幕之后得到：

源码 复制 打印 ?

```
/dev/input/event0: 0003 0039 000002a5
/dev/input/event0: 0003 0030 00000004
/dev/input/event0: 0003 0035 0000017b
/dev/input/event0: 0003 0036 000001cf
```

文章分类

文档操作 (12)
VC++学习 (102)
目标跟踪 (4)
面试相关 (132)
MFC (46)
图像处理 (19)
Android开发 (136)
多线程 (32)
linux面试 (3)
电脑相关 (2)
iphone开发教程 (1)
行政测试 (1)
Java学习 (7)
ubuntu系统 (6)
mysql (4)
OCJP (1)
web (1)
CI框架 (2)
技术小谈 (1)
nodejs (1)

文章存档

2015年07月 (1)
2014年10月 (4)
2014年08月 (6)
2014年07月 (1)

关闭

2014年06月 (1)

展开

阅读排行

ImageView.ScaleType设 (56296)
sift是图像匹配的非常经典 (42717)
RGB与YUV转换 (39123)
ShareSDK 微信及其朋友 (34308)
Android下Activity结束（f (29558)
C++中反正切atan2(y,x)与 (23374)
解决Android Studio中调i (22175)
adb shell input keyevent (22047)
Android Studio中添加重要 (20775)
C语言移位运算符 (19036)

评论排行

sift是图像匹配的非常经典 (15)
Android下Activity结束（f (13)
C++ 多线程编程实例 (6)
ImageView.ScaleType设 (5)
自绘对话框标题栏 (5)
Android开源项目汇总 (5)
Android Studio中添加重要 (5)
MFC子窗口向父窗口发送 (4)

```
/dev/input/event0: 0003 003a 0000001c
/dev/input/event0: 0000 0000 00000000
/dev/input/event0: 0003 0039 ffffffff
/dev/input/event0: 0000 0000 00000000
```

很难辨认，加-l参数后得到（getevent -l）

源码 复制 打印 ？

```
/dev/input/event0: EV ABS       ABS MT TRACKING ID     000002a6
/dev/input/event0: EV ABS       ABS MT TOUCH MAJOR   00000004
/dev/input/event0: EV ABS       ABS MT POSITION X      0000017b
/dev/input/event0: EV ABS       ABS MT POSITION Y      000001cf
/dev/input/event0: EV ABS       ABS MT PRESSURE         0000001c
/dev/input/event0: EV SYN       SYN REPORT                  00000000
/dev/input/event0: EV ABS       ABS MT TRACKING ID     ffffffff
/dev/input/event0: EV_SYN       SYN_REPORT                  00000000
```

格式为 device: type code value，即 设备、输入设备类型、按键扫描码、附加码，具体定义可从kernel/include/lin

中获得，至于这个头文件，途径之一是从google官网源码中获取：

type: 输入设备类型，在手机系统中经常使用的键盘（keyboard）和小键盘（kaypad）属于按键设

备EV_KEY，轨迹球属于相对设备EV_REL，触摸屏属于绝对设备EV_ABS

code: 按键扫描码，区别于ASCII码和SDK中KeyEvent的键码

value: 附加码，1/0 down/up

第一行：可理解为一次touch的开始

第2行：可理解为点击开始

第3行：触摸点x坐标

第4行：触摸点y坐标

第5行：可理解为触摸点大小

第6行：事件同步(点击结束)

五年程序员谈软件工程师 (4)
RGB与YUV转换 (3)

第7行：一次touch结束

第8行：事件同步(事件结束)

一次touch此8行是必须的，如果是longTouch呢，在touch的基础上，重复若干次第2~6行，即看起来可能是这样子：

源码 复制 打印 ？

```
01. /dev/input/event0: EV ABS       ABS MT TRACKING ID     000002a6
02. /dev/input/event0: EV ABS       ABS MT TOUCH MAJOR   00000004
03. /dev/input/event0: EV ABS       ABS MT POSITION X      0000017b
04. /dev/input/event0: EV ABS       ABS MT POSITION Y      000001cf
05. /dev/input/event0: EV ABS       ABS MT PRESSURE         0000001c
06. /dev/input/event0: EV_SYN       SYN_REPORT                  00000000
07.
08. /dev/input/event0: EV ABS       ABS MT TOUCH MAJOR   00000004
09. /dev/input/event0: EV ABS       ABS MT POSITION X      0000017b
10. /dev/input/event0: EV ABS       ABS MT POSITION Y      000001cf
11. /dev/input/event0: EV ABS       ABS MT PRESSURE         0000001c
12. /dev/input/event0: EV_SYN       SYN_REPORT                  00000000
13.
14. /dev/input/event0: EV ABS       ABS MT TOUCH MAJOR   00000004
15. /dev/input/event0: EV ABS       ABS MT POSITION X      0000017b
16. /dev/input/event0: EV ABS       ABS MT POSITION Y      000001cf
17. /dev/input/event0: EV ABS       ABS MT PRESSURE         0000001c
18. /dev/input/event0: EV_SYN       SYN_REPORT                  00000000
19.
20. ......
21.
22. /dev/input/event0: EV ABS       ABS MT TRACKING ID     ffffffff
23. /dev/input/event0: EV_SYN       SYN_REPORT                  00000000
```

关闭

如果是drag呢，在longTouch的基础上，xy坐标从起点到终点是渐变的，其他可认为一样。

对于按键，HOME为例，一目了然：

源码 复制 打印 ？

```
/dev/input/event0: 0001 0066 00000001
/dev/input/event0: 0000 0000 00000000
/dev/input/event0: 0001 0066 00000000
/dev/input/event0: 0000 0000 00000000

/dev/input/event0: EV KEY       KEY HOME             DOWN
/dev/input/event0: EV SYN       SYN REPORT           00000000
/dev/input/event0: EV KEY       KEY HOME             UP
/dev/input/event0: EV_SYN       SYN_REPORT           00000000
```

所以清楚了之后，使用sendevent进行事件模拟就很轻松了，关键是从input.h中获取按键对应的扫

描码，也许每个版本的系统中input.h的内容都稍有不同，这个很头疼。

值得注意的是使用getevent获取的数值都是16进制的，而sendevent使用的是10进制的，需要进行

转换。

如点击坐标: 120,254

关闭

源码 复制 打印 ？

```
sendevent /dev/input/event1 0003 0057 00000000      <---事件开始
sendevent /dev/input/event1 0003 0048 00000010      <---点击开始
sendevent /dev/input/event1 0003 0058 00000070      <---   触摸范围
sendevent /dev/input/event1 0003 0053 00000120      <---   x坐标
sendevent /dev/input/event1 0003 0054 00000254      <---   y坐标
sendevent /dev/input/event1 0000 0000 00000000      <---点击结束(同步)
sendevent /dev/input/event1 0003 0057 4294967295        <---事件结束
sendevent /dev/input/event1 0000 0000 00000000      <---事件同步
```

翻译为：

type code value

EV_ABS ABS_MT_TRACKING_ID 00000000 <---事件开始

EV_ABS ABS_MT_TOUCH_MAJOR 00000010 <---点击开始

EV_ABS ABS_MT_PRESSURE 00000070 <--- 触摸范围

EV_ABS ABS_MT_POSITION_X 00000120 <--- x坐标

EV_ABS ABS_MT_POSITION_Y 00000254 <--- y坐标

EV_SYN SYN_REPORT 00000000 <---点击结束(同步)

EV_ABS ABS_MT_TRACKING_ID 4294967295 <---事件结束

EV_SYN SYN_REPORT 00000000 <---事件同步

对于longTouch，重复几次上面提到的步骤，而drag再弄个坐标渐变。

按键方面(MENU) DOWN：1 UP：0

源码 复制 打印 ？

```
sendevent /dev/input/event0 0001 0102 0000000001
sendevent /dev/input/event0 0000 0000 0000000000
sendevent /dev/input/event0 0001 0102 0000000000
sendevent /dev/input/event0 0000 0000 0000000000

EV KEY       KEY MENU            DOWN
EV SYN       SYN REPORT         00000000
EV KEY       KEY MENU            UP
EV_SYN       SYN_REPORT         00000000
```

关闭

此外还可以模拟可见字符按键。

到这里是否觉得getevent好用好理解，但sendevent却超级麻烦。没关系，如果真不想用sendevent

模拟事件的话，可以转向input命令。

个人觉得这个input命令是个重量级的东东，调用时间非常长，貌似低系统版本和高系统版本的input提

供的功能还不一样，但它毕竟好用啊，如下：

源码 复制 打印 ？

```
usage: input ...
       input text <string>
       input keyevent <key code number or name>
       input tap <x> <y>
       input swipe <x1> <y1> <x2> <y2>
```

可见除了longTouch无法模拟之外，其他的都很简便了

输入文本： input text abcdefg

按键 ： input keyevent KEYCODE_MENU

点击 ： input tap 100 300

拖拽 ： input swipe 100 600 500 600

①对于文本输入，以下字符需要加 \ 进行转义：

` ' " ~ # & ( ) | \ ; < >

②对于longTouch，使用input如何模拟？

关闭

③最后，在android中不管使用sendevent或者input，如何能模拟输入非ASCII字符呢，比如中文？

这是个难题

上一篇 adb shell input keyevent值所对应的字符

下一篇 基础总结篇之二：Activity的四种launchMode

查看评论

暂无评论

您还没有登录,请[登录]或[注册]

* 以上用户言论只代表其个人观点，不代表CSDN网站的观点或立场

关闭