

## chen825919148的专栏

目录视图

摘要视图

RSS 订阅

## 个人资料



chen825919148

访问：1141486次

积分：11241

等级：BLOG &gt; ?

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

异步赠书：9月重磅新书升级，本本经典 程序员9月书讯 每周荐书：ES6、虚拟现实、物联网（评论送书）

## 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，希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，见谅/不吝指教)

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

## 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)

```
07. -d: show HID descriptor, if available
08. -p: show possible events (errs, dev, name, pos. events)
09. -i: show all device info and possible events
10. -l: label event types and names in plain text
11. -q: quiet (clear verbosity mask)
12. -c: print given number of events then exit
13. -r: print rate events are received
14.
15. 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 ?

```
01. add device 1: /dev/input/event1
02.   name:      "pmic8xxx pwrkey"
03. add device 2: /dev/input/event3
04.   name:      "apq8064-tabla-snd-card Headset Jack"
05. add device 3: /dev/input/event2
06.   name:      "apq8064-tabla-snd-card Button Jack"
07. add device 4: /dev/input/event5
08.   name:      "gpio-keys"
09. could not get driver version for /dev/input/mice, Not :
10. add device 5: /dev/input/event0
11.   name:      "atmel mxt ts"
12. add device 6: /dev/input/event4
13.   name:      "mhl_rcp"
```

在点击设备屏幕之后得到 :

源码 复制 打印 ?

```
01. /dev/input/event0: 0003 0039 000002a5
02. /dev/input/event0: 0003 0030 00000004
03. /dev/input/event0: 0003 0035 0000017b
04. /dev/input/event0: 0003 0036 000001cf
```

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 (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)点 (23374)
- 解决Android Studio中调i (22175)
- adb shell input keyevent (22047)
- Android Studio中添加重i (20775)
- C语言移位运算符 (19036)

评论排行

- sift是图像匹配的非常经 (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i (5)
- MFC子窗口向父窗口发这 (4)

```
05. /dev/input/event0: 0003 003a 0000001c
06. /dev/input/event0: 0000 0000 00000000
07. /dev/input/event0: 0003 0039 ffffffff
08. /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 ?

```
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. /dev/input/event0: EV_ABS      ABS MT TRACKING ID      ffffffff
08. /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/linux/input.h 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 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)

关闭

[五年程序员谈软件工程师](#) (4)[RGB与YUV转换](#) (3)

### 推荐文章

[\\* CSDN新版博客feed流内测用户征集令](#)[\\* Android检查更新下载安装](#)[\\* 动手打造史上最简单的Recycleview 侧滑菜单](#)[\\* TCP网络通讯如何解决分包粘包问题](#)[\\* SDCC 2017之大数据技术实战线上峰会](#)[\\* 快速集成一个视频直播功能](#)

### 最新评论

[ImageView.ScaleType设置图解](#)  
冰霜雨露: 个人技术博客 欢迎拍砖。 <http://wangchuangshi.com>[JAVA反射机制作用是什么](#)  
冰霜雨露: 个人技术博客, 欢迎共同讨论。  
<http://wangchuangshi.com>[Android多个Activity切换时其生命](#)  
fengyu09: @bright\_future\_: 阈值太低。[计算n!中结尾零的个数——上海](#)  
jiayuechao: 您好 刚来上海一直想去先锋工作, 您在先锋工作吗, 能指点一下吗 c/c++ 方面的知识呢[OCJP考试介绍](#)

rc447516551: 您好, 我想考ocjp, 请问你是自学考试还是通过培训机构的?

[Android多个Activity切换时其生命](#)

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

```

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:

关闭

让你们占用我的昵称: 看头像可耻地硬了。

解决Android Studio中调试总出现Pradator: mac版的怎么解决大神求教啊

adb shell input keyevent值所对应的qq\_37892127: 感谢, 有帮助!

C++ 多线程编程实例

looking\_: @cristinaSophia:试试CreateMutexA

五年程序员谈软件工程师的职业流年的往返: 额, 这是本人写的么? 还有, 头像是本人么!!! (づー3ー)づ

源码 复制 打印 ?

```
01. /dev/input/event0: 0001 0066 00000001
02. /dev/input/event0: 0000 0000 00000000
03. /dev/input/event0: 0001 0066 00000000
04. /dev/input/event0: 0000 0000 00000000
05.
06. /dev/input/event0: EV_KEY      KEY_HOME      DOWN
07. /dev/input/event0: EV_SYN      SYN_REPORT    00000000
08. /dev/input/event0: EV_KEY      KEY_HOME      UP
09. /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

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

源码 复制 打印 ?

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

翻译为:

关闭

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 ?

```
01. sendevent /dev/input/event0 0001 0102 0000000001
02. sendevent /dev/input/event0 0000 0000 0000000000
03. sendevent /dev/input/event0 0001 0102 0000000000
04. sendevent /dev/input/event0 0000 0000 0000000000
05.
06. EV_KEY      KEY_MENU      DOWN
07. EV_SYN      SYN_REPORT    00000000
08. EV_KEY      KEY_MENU      UP
09. EV_SYN      SYN_REPORT    00000000
```

此外还可以模拟可见字符按键。

关闭

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 ?
01.  usage: input ...
02.      input text <string>
03.      input keyevent <key code number or name>
04.      input tap <x> <y>
05.      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

顶

1

踩

0

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](#)

#### 相关文章推荐

- [Android输入法框架系统\(下\)](#)
- [Android 通过ADB模拟按键、点击、滑动等事件](#)
- [Presto的服务治理与架构在京东的实践与应用--王...](#)
- [Retrofit 从入门封装到源码解析](#)
- [getevent/sendevent 使用说明](#)
- [android系统中sendevent的应用](#)
- [深入掌握Kubernetes应用实践--王渊命](#)
- [自然语言处理工具Word2Vec](#)
- [android adb sendevent 模拟点击](#)
- [Android getevent/sendevent详解](#)
- [Python基础知识汇总](#)
- [常用adb shell命令：getevent和sendevent](#)
- [Android sendevent/getevent 用法](#)
- [linux input输入子系统分析《四》：input子系统整...](#)
- [Android核心技术详解](#)
- [如何防止android程序被kill掉](#)

#### 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

[公司简介](#) | [招贤纳士](#) | [广告服务](#) | [联系方式](#) | [版权声明](#) | [法律顾问](#) | [问题报告](#) | [合作伙伴](#) | [论坛反馈](#)

网站客服

杂志客服

微博客服

webmaster@csdn.net

400-660-0108

| 北京创新乐知信息技术有限公司 版权所有 | 江苏知之为计算机有限公司 | 江苏乐知网络技术有限公司

京 ICP 证 09002463 号 | Copyright © 1999-2017, CSDN.NET, All Rights Reserved



关闭