TUTORIAL: Remote Control Touch Screen

By Twin0ne, Junior Member on 7th September 2012, 05:46 PM

1 posts
Thanks: 28

Tweet Like +1

Reply Search Thread

Do you want to activate the screen wirelessly, or play some music at your phone over SSH because you don't know where you left it?
THIS IS YOUR PLACE!!!

This tutorial will show you how to create a file, that simulates a keypress, swipe, button press.
You can even draw something, save a file of data, and then replay it so android draws the same something.
You can simulate somebody writing text with the default android keyboard, over SSH, etc.

FORUMS

Android Software and Hacking General [Developers Only]

Android General

Android Q&A, Help & Troubleshooting

Android Software Development

Miscellaneous Android Development

[More]

Remove All Ads from XDA

Let's start by the beginning: I will add a zip package with my working directory at my phone, all tools are included there.
The dropbear ssh server is included because it's practical because it can be turned on with your android terminal.
You can also access the other tools using another ssh server.

The media player control is a very simple script that takes an argument and then translates it to the android "input keyevent" command.

And the low level screen events is what i'm going to talk about most.

From here on, this will be a tutorial that will focus on screen events
I will cover the following topics:

1) Understanding and getting screen events.
2) Simulating keypressess, creating and sending screen events.
3) Automating the tasks.

UNDERSTANDING AND GETTING SCREEN EVENTS

Android provides two commandline tools for managing screen events: getevent and sendevent.

These commands are kinda cryptic:

We'll take a look at the "getevent" command. This command outputs all events of a given device (or all if no device is given)

TOP FORUM DISCUSSIONS

How to Root the Samsung Galaxy Note 8 (Exynos)
September 25, 2017

How to Enable Fingerprint Selfie on the Xiaomi Redmi 3s
September 25, 2017

“Can’t Talk” Lets You Auto-Reply to WhatsApp, Slack and More
September 20, 2017

Google Nexus 4 Receives a Port of Android 8.0 Oreo
September 19, 2017

Transcriber Converts Voice Notes in WhatsApp to Text
September 10, 2017

Install the H2OS Boot Animation from the OnePlus 5 using a Magisk Module
September 6, 2017

OpenSans and Roboto Fonts for Unrooted Huawei and Honor Devices on EMUI 4.0+
September 6, 2017

More Forum Links

I just use getevent (hint, press Control+C to terminate the command and stop getting events):

Code:

```
root@android:/ # getevent
add device 1: /dev/input/event8
  name:     "compass_sensor"
add device 2: /dev/input/event7
  name:     "sec_touchkey"
add device 3: /dev/input/event6
  name:     "gyro_sensor"
add device 4: /dev/input/event5
  name:     "light_sensor"
add device 5: /dev/input/event4
  name:     "proximity_sensor"
add device 6: /dev/input/event3
  name:     "sii9234_rcp"
add device 7: /dev/input/event2
  name:     "sec_touchscreen"
```

You may notice the "add device X" number is not matching the /dev/input/eventX number. I will use the latter to refer to an event, as we will use this command a lot, believe me.

I will be only talking about devices 1, 2 and 7, i don't care about the rest.

/dev/input/event1 (gpio-keys): This are the hardware keys.
On my Samsung Galaxy SII, these keys are POWER, HOME, VOLUP, and VOLDOWN keys.

/dev/input/event2 (sec_touchscreen): is the touch-screen (they're not very creative at android developing

/dev/input/event7 (sec_touckey): The touch keys.
On my Galaxy, i have the MENU and BACK buttons as touchkeys. This touchkeys are not part of the touchscreen, thats why its another device.
Note though that the touch_screen and the touck_keys devices ARE connected, i'll later have you remember this line.

The numbers of the devices may change from device to device, as not all phones have touchkeys, or a touch screen.
Well, now you know how to list the devices.

The next step is to see how each device works:

Device 1: HWKEYS

I recommend following these steps on your device, and comment any changes at this thread, thank you.

Let's try out short-pressing just the POWER button of our phone (or unlock button)
When i refer to short-pressing, i mean pressing and immediately releasing the button. You'll soon understand why.
Use getevent, press the key, and finish with Control + C.

Code:

```
root@android:/ # getevent /dev/input/event1
0001 0074 00000001
0000 0000 00000000
0001 0074 00000000
0000 0000 00000000
^C
```

WTF is this?! Yes, i said that too when i first saw this, (and you havent seen a swipe with 4 fingers yet, haha)

What can we observe? There are 3 fields, and all numbers.

The first field will always be 1 except in separator lines (see the second line)
The second field is the identifier for the POWER key.
The third field is a boolean value, it will be 1 when the key is pressed and 0 when it is released.

0000 0000 00000000
This is a simple separator, it it means android just processed everything ultil here.

0001 0074 00000000
This is a KEYUP line, as you can see, the identifier field is 1, the keycode field is 74, and the third boolean field is set to 0, what indicates the key is released.

0000 0000 00000000
And another separator so android process it.

^C
This is the Control+C character, it is used to terminate the running program in linux terminal.

Actually quite simple, isn't it? Lets take a look at the touchkeys device (it's simpler as the screen)

Device 7: MENU and BACK

Yes, android has set up a whole device driver only for 2 stupid keys.
Let's see them in action. I will first press the MENU key, and then the BACK key.

Note that now i use "getevent /dev/input/event7", and not event1.
Oh, i will make some comments on the output so you can see it more clear.

Code:
```
0000 0000 00000000    [S]  SEPARATOR
0001 008b 00000000    [FU] KEYUP (8b)
```

```
0001 009e 00000000    [FU] KEYUP (9e)
0000 0000 00000000    [S]  SEPARATOR
^C
130|root@android:/ #
```

Wow, hex!!, the POWER button we pressed before, was also hex.
Remember that getevents are all in hex.

Explanation:
[FD] Is (in my syntax), a FINGERDOWN event, that means the finger touches the screen.
[FU] FINGERUP event, the finger leaves the screen.
[S] Separator, processess the event.

The parentheses i put after the FU and FD events are the ID of the key.

Remember i told you that the touch_keys and touch_screen devices were connected?
I dont know why but the first fields of the non-separator events are also 1 (same as the HWKEYS)
You'l see that with touchscreen events this is different (it uses number 3 always)

Well, the syntax of the events in the touchkeys are the same as the hardware keys.
Actually, i consider this touch keys to be hardware, with the consideration that they only work when the screen is on.

Device 2: SCREEN TOUCHES AND SWIPES (The interesting part)

Now it's getting really interesting.
Let's get a simple tap somewhere random at the screen:

Code:

```
0003 0036 000001dd  [Y]  Y coordinate

// NOTE LINE

0000 0000 00000000  [S]  Separator

0003 0039 ffffffff  [FU] FingerUp (24f)
```

NOTE: At the NOTE LINE you MAY get 2 other events, 0030 and 003a. I have no idea what these two are for.
Besides, to simulate a click we don't need them, so if anyone know what these two events are, you're welcome to comment.
If you enable the Show touches & Pointer location, you see that it has some fields: PRS and SIZE, Those are for pressure and size, of course, but i don't know if they have something to do with this 0030 and 003a.

> Tip for those of you that are debugging: if you increase pressure, your finger's contact area with the device grows, as your sking gets down.
> I realised this thinking of somebody pushing his face to a glass surface and somebody watching at the other side.

NOTE 2: You are recommended to enable Settings -> Developer Options -> Show touches & Pointer location

Understanding touchscreen events is more complicated. All buttons from HWKEYS and TOUCHKEYS, were quite simple: they only have a key identifier, and a boolean state: Pressed (1) or not pressed (0).

However, when we use the touchscreen, every single tap has the following elements (events):

NOTE: The third field at the figerdown event, is a sort of simple counter, across different taps, swipes, etc. I dont know yet when it is resetted, but for sure at reboot as this counter will probably reside in ram.

[X] Number of pixels counted from left side.
[Y] Number of pixels counted from top.
[S] Separator / process trigger.

[FU] Finger Up (The finger exits the screen completely)
[S] Another separator.

Notice that the 0030 and 003a events are not in my list. I've made a tool to simulate keypressess, and those two events are NEVER included, but my script works always. So something tells me these two events are worthless.

Introduction to swipe
With the pointer location enabled in settins, try swiping somewhere random.
As you may notice, you aren't perfect, so you can't draw a straight line.
Everytime your finger changes direction, a new point is added. this means that if you want to make a square, you would need 4 points, imagining all lines are straight.

If you take a close look at the screen, the POINTER LOCATION shows a colored dot, everytime you set some coordinates (this means, by a single click, or by a direction change with a swipe, or with a double tap it would make two dots).

I will unlock my screen with my finger, and then analyze the output of getevent:
Lets analize this simple swipe, and quicly move on to the last teorichal part: Multiple taps.

Code:
```
root@android:/ # getevent /dev/input/event2
0003 0039 000002ba [FD]
0003 0035 000000e6 [X]
```

```
0000 0000 00000000 [S]

0003 0035 000000fb [X]
0003 0036 00000296 [Y]
0000 0000 00000000 [S]

0003 0035 00000120 [X]
0003 003a 00000004 [?]
0000 0000 00000000 [S]

0003 0035 00000161 [X]
0003 0036 00000292 [Y]
0003 003a 00000003 [?]
0000 0000 00000000 [S]

0003 0035 000001aa [X]
0003 0036 0000029a [Y]
```

Take a look at the above code. You should already get the point, at every separator the events are processed. You can use a separator to update coordinates (this includes when a finger enters the screen), and to release a finger (at the last coordinates inputted for that finger).

This that the 2nd to 5th block, only update the coordinates of the finger that is pressing the screen.

Multiple taps
Multiple taps introduces something new: Identificators [ID]
For this i will use the following: a thriple tap.
There is a order in which the fingers enter and leave the screen, while one finger is still pressed, another one can

```
130|root@android:/ # getevent /dev/input/event2
0003 0039 000002c0 [FD]
0003 0035 00000057 [X]
0003 0036 00000129 [Y]
0000 0000 00000000 [S]

0003 002f 00000001 [ID] 1
0003 0039 000002c1 [FD]
0003 0035 00000169 [X]
0003 0036 000001bb [Y]
0000 0000 00000000 [S]

0003 002f 00000002 [ID] 2
0003 0039 000002c2 [FD]
0003 0035 000000de [X]
0003 0036 00000187 [Y]
0000 0000 00000000 [S]

0003 0039 ffffffff [FU] // NOTE 1
0000 0000 00000000 [S]

0003 002f 00000000 [ID] 0 // NOTE 2
```

Well, its actually quite simple. As there are multiple fingers, android adds ID's to this new fingers.
The default ID = 0, android doesn't know if you're gonna press 1 finger or 200, so it doesnt add a ID to the first.
The first finger's ID is always 0.
When you add a finger, android adds ID=1
If you release a finger, ID=0 is available again.

just learned to combine swipe with multiple touches)

Thats why at the line i marked with NOTE 1, there is no ID, because the first finger i released, was finger ID=2 (the last that touched the screen)

Important: When you release a finger, you MUST set an ID, because android only keeps track of the CID (Current ID), but not of the previus.
So if you release a finger, and then set some coordinates, they will affect an ID that is no longer pressed, and it could either do NOTHING, or output some error message :P

SIMULATING KEYPRESSESS, CREATING AND SENDING EVENTS

NOTE: When we use "getevent", we get some hex codes. When we use "sendevent" we have to use decimal values.
Why? I don't know, maybe bothering is the point. Anyway, it's like that, and we'll have to manage, right?

Sendevent takes the EXACT same input that getevent outputs, except that GETEVENT outputs [HEX]adecimal, and Sendevents wants [DEC]imal characters

To simulate the POWER button key, for one single click, it would take this steps:

STEP 1

STEP 2

STEP 3

Edit: I'm currently making the SENDEVENT TUTORIAL part, in about 2 hours it will be done.

Vote for newsworthy if you think it was!!!

trevd
Inactive Recognized Developer

Thanks: 1,268

8th September 2012, 01:40 AM | #2

> Quote:
> Originally Posted by **Twin0ne**
>
> Do you want to activate the screen wirelessly, or play some music at your phone over SSH because you don't know where you left it?
> THIS IS YOUR PLACE!!!

Welcome to XDA, This is a great way to "roll into to town"

I've been planning to research getevent/sendevent command input for a while, ever since one of my tablet touchscreens decided to die, Mainly to simulate long pressing.

If you don't want to write to the /dev nodes directly you can always step up a level of abstraction and use the /system/bin/input commnd. Prior to jelly bean this used to only support KEYEVENT codes and text input. The JB version however now supports tap and swipe aswell.

Code:

```
input keyevent <key code number or name>
input tap <x> <y>
```

Here (androidxref) is a list of Valid KeyCodes for use with the keyevent parameter.

I've gone a bit nuts with a custom adb that I use so i've wrapped a lot of commnds into one word adb commands like "adb power" , "adb back" etc

Once again thanks for sharing, low level android tricks are always welcome!

The Following 5 Users Say Thank You to trevd For This Useful Post: [View] Gift trevd Ad-Free

Rupert Rawnsley
Junior Member

Thanks: 4

4th January 2013, 10:52 AM | #3

Very useful description thank you. I've found some other relevant info...

Adding -l to the command line of getevent adds the event names, also adding -d preserves the HID descriptor. A single touch event looks like this...

```
root@android:/ # getevent -d -l /dev/input/event2
EV_ABS ABS_MT_TRACKING_ID 00000071
EV_ABS ABS_MT_POSITION_X 0000020d
EV_ABS ABS_MT_POSITION_Y 00000160
EV_SYN SYN_REPORT 00000000
```

Also, there is a description of the linux multi-touch protocol here: http://www.kernel.org/doc/Documentat...h-protocol.txt

The Following 3 Users Say Thank You to Rupert Rawnsley For This Useful Post: [View] **Gift Rupert Rawnsley Ad-Free**

adfree
Senior Member

Thanks: 3,554

22nd August 2014, 02:00 AM | **#4**

Thank you very much for this thread.

This helps me better understand how Android works.

And I have chance to use Touch "simulation"...
http://forum.xda-developers.com/show...&postcount=701

Best Regards

adfree
Senior Member

Thanks: 3,554

27th August 2014, 12:20 PM | **#5**

Only as info or example...

Code:

```
adb shell sendevent /dev/input/event3 3 48 28
adb shell sendevent /dev/input/event3 0 2 0
adb shell sendevent /dev/input/event3 0 0 0
adb shell sendevent /dev/input/event3 3 57 0
adb shell sendevent /dev/input/event3 3 53 4
adb shell sendevent /dev/input/event3 3 54 327
adb shell sendevent /dev/input/event3 3 48 32
adb shell sendevent /dev/input/event3 0 2 0
adb shell sendevent /dev/input/event3 0 0 0
adb shell sendevent /dev/input/event3 3 57 0
adb shell sendevent /dev/input/event3 3 53 5
adb shell sendevent /dev/input/event3 3 54 327
adb shell sendevent /dev/input/event3 3 48 48
adb shell sendevent /dev/input/event3 0 2 0
adb shell sendevent /dev/input/event3 0 0 0
adb shell sendevent /dev/input/event3 3 57 0
adb shell sendevent /dev/input/event3 3 53 6
adb shell sendevent /dev/input/event3 3 54 327
adb shell sendevent /dev/input/event3 3 48 49
adb shell sendevent /dev/input/event3 0 2 0
```

I have created simple **Batch** file... ***.bat**

I can unlock Screenlock/Screensaver with this...
Tested on my S8600 with I9001 Firmware... because unsupported Hardware and Touch not working yet....

Also on I8150 tested successfully... because taken from I8150...
Here Log, which I have converted into Dec Values by my little brain... line by line...

Code:

```
0003 0036 00000146
0003 0030 0000001c
0000 0002 00000000
0000 0000 00000000
0003 0039 00000000
0003 0035 00000004
0003 0036 00000147
0003 0030 00000020
0000 0002 00000000
0000 0000 00000000
0003 0039 00000000
0003 0035 00000005
0003 0036 00000147
0003 0030 00000030
0000 0002 00000000
0000 0000 00000000
0003 0039 00000000
0003 0035 00000006
0003 0036 00000147
0003 0030 00000031
```

I am using old Android Version, where only **input text** and **input keyevent** is supported...

I can't install Android 4.x during incompatibility on my GT-S8600 yet...

Best Regards

LOGIN REGISTER search plus

19th April 2015, 12:18 AM | #6

Hi,
what i understand so far.
for each device you have different "targets" to simulate e.g. the press/hold of a key. Actually i found it for my device and managed to simulate my hardware/soft keys with tasker-run shell and sendevent...
Question1: how do key-remapper handle that ? Do they question the system e.g. what device the menu-key is ?
But more important to me is **Question2:** How can i simulate CTRL-C assuming that it is copy (and CTRL-V,CTRL-a) for a floating addition for my soft keyboard. I already managed to have a taker-scene, which floats as an overlay an by pressing my button called "test" the scene(overlay) hides and i can simulate a (sequence of keys or i can sendevent-stuff)...and there i would call Select-All,copy,paste.... Any idea (and no, the standard bar popping up when editing text with these functions does not pop up with swiftkey or in other situations i might want to use that). The keyboard "programmer keyboard" has this as buttons..so the function must be there somewhere...
In short: how to emulate CTRL-C ?

RuggedHunter
Senior Member

Thanks: 1,088

12th January 2016, 07:01 PM | #7

I need to emulate a swipe up and down the side of my screen, in a single touch. I'm trying to go from x=1350 y=1350 to x=1350 y=750 and back down to x=1350 y=1350

I can do this in two distinct touches with:
input swipe 1350 1350 1350 750
input swipe 1350 750 1350 1350

But I need it completed in one single touch, so I'm trying to use sendevent for the first time. I thought this would

```
sendevent /dev/input/event1 3 54 1350
sendevent /dev/input/event1 0 2 0
sendevent /dev/input/event1 0 0 0
sendevent /dev/input/event1 3 57 0
sendevent /dev/input/event1 3 53 1350
sendevent /dev/input/event1 3 54 750
sendevent /dev/input/event1 0 2 0
sendevent /dev/input/event1 0 0 0
sendevent /dev/input/event1 3 57 0
sendevent /dev/input/event1 3 53 1350
sendevent /dev/input/event1 3 54 1350
sendevent /dev/input/event1 0 2 0
sendevent /dev/input/event1 0 0 0
```

When I do this, I don't get any swipe at all, and my touchscreen gets extremely laggy, then unresponsive to single taps or swipe gestures...but double taps work.

I need to reboot by Vol Down + Power to rescue the device.

Sent from my S6 Edge+ g928c
BobCatRom

Pwallius
Junior Member

Thanks: 4

30th July 2016, 05:34 PM | #8

Quote:

Were you able to simulate a long press? I'm currently stuck on that with my project. I want to simulate a long press in Chrome so I can open the share dialog for a link and extract the URL. I was hoping to be able to do this through the Accessibility service but it seems long press isn't supported. Thanks!

Reply Subscribe to Thread

Previous Thread Next Thread

Top Threads in Android Software and Hacking General [Developers Only] by ThreadRank

[Port]Pixel launcher for Marshmallow & Lollipop 2017-04-30
Android Software and Hacking General [Developers Only]

[mod] new ultimate performance and battery mod -stonedmod- 2017-05-12
Android Software and Hacking General [Developers Only]

[KITANA Tweak V5.0.1] [Power Of Your Android] [BIG RELEASE] 2017-07-01
Android Software and Hacking General [Developers Only]

LOGIN ▸ REGISTER ▸ search plus

[MOD] [KK+] Android O Beta Redesigned Emoji Set v2 [3 Aug 2017] 2017-05-17
Android Software and Hacking General [Developers Only]

[MOD][Port] MIEZU™ FlymeOS 6 Experience Package On All Devices [LP/MM/N] 2017-06-10
Android Software and Hacking General [Developers Only]

[TUTORIAL] Vernee Mars Pro Root + TWRP 2017-08-06
Android Software and Hacking General [Developers Only]

[Build.prop][Tweaks] Emperor Tweaks (God Mode) for 5.0+ - 8.0.x 2017-09-14
Android Software and Hacking General [Developers Only]

xda-developers ➞ Android Development and Hacking ➞ Android Software and Hacking General [Developers Only] ➞ TUTORIAL: Remote Control Touch Screen by Twin0ne