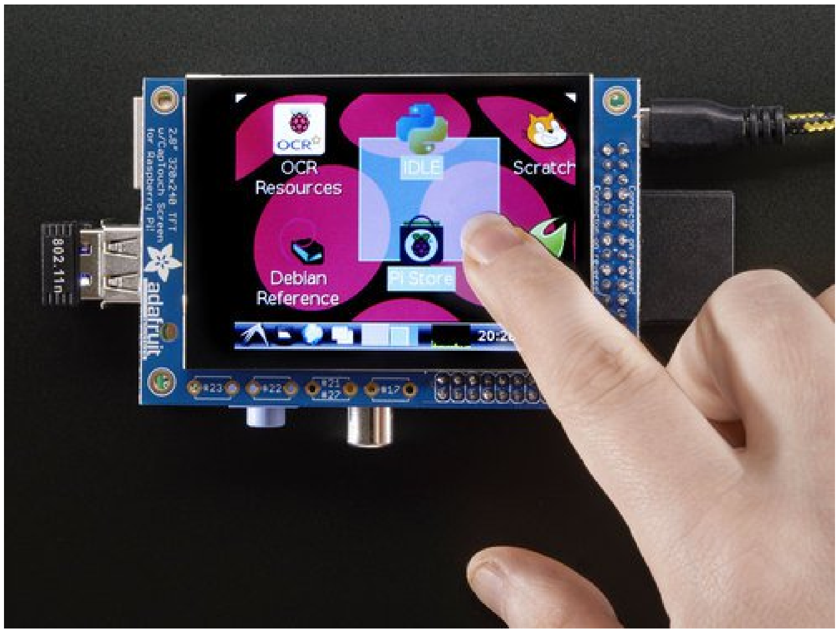
Adafruit 2.8" PiTFT - Capacitive Touch

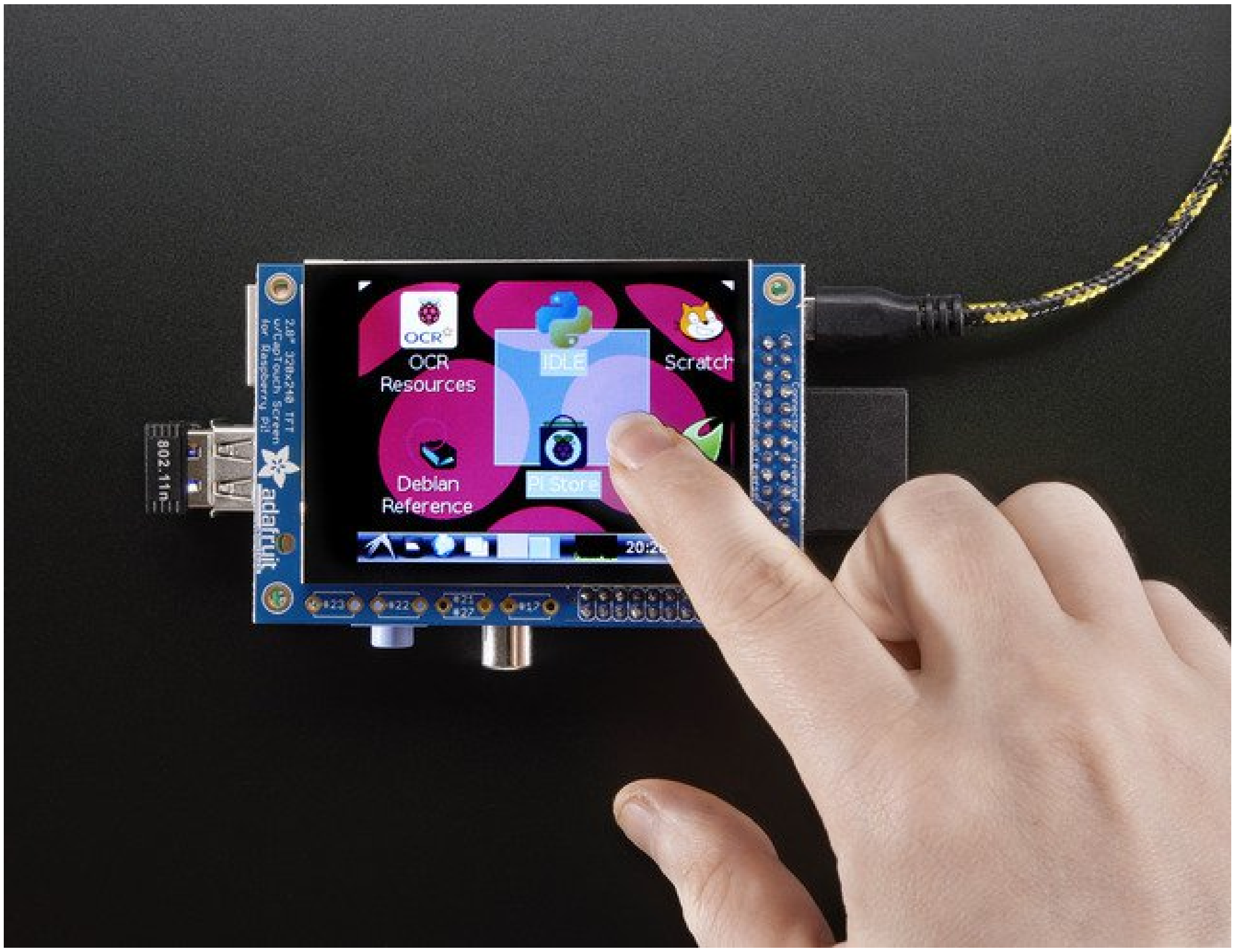
Created by lady ada



Last updated on 2016-11-26 04:43:44 AM UTC

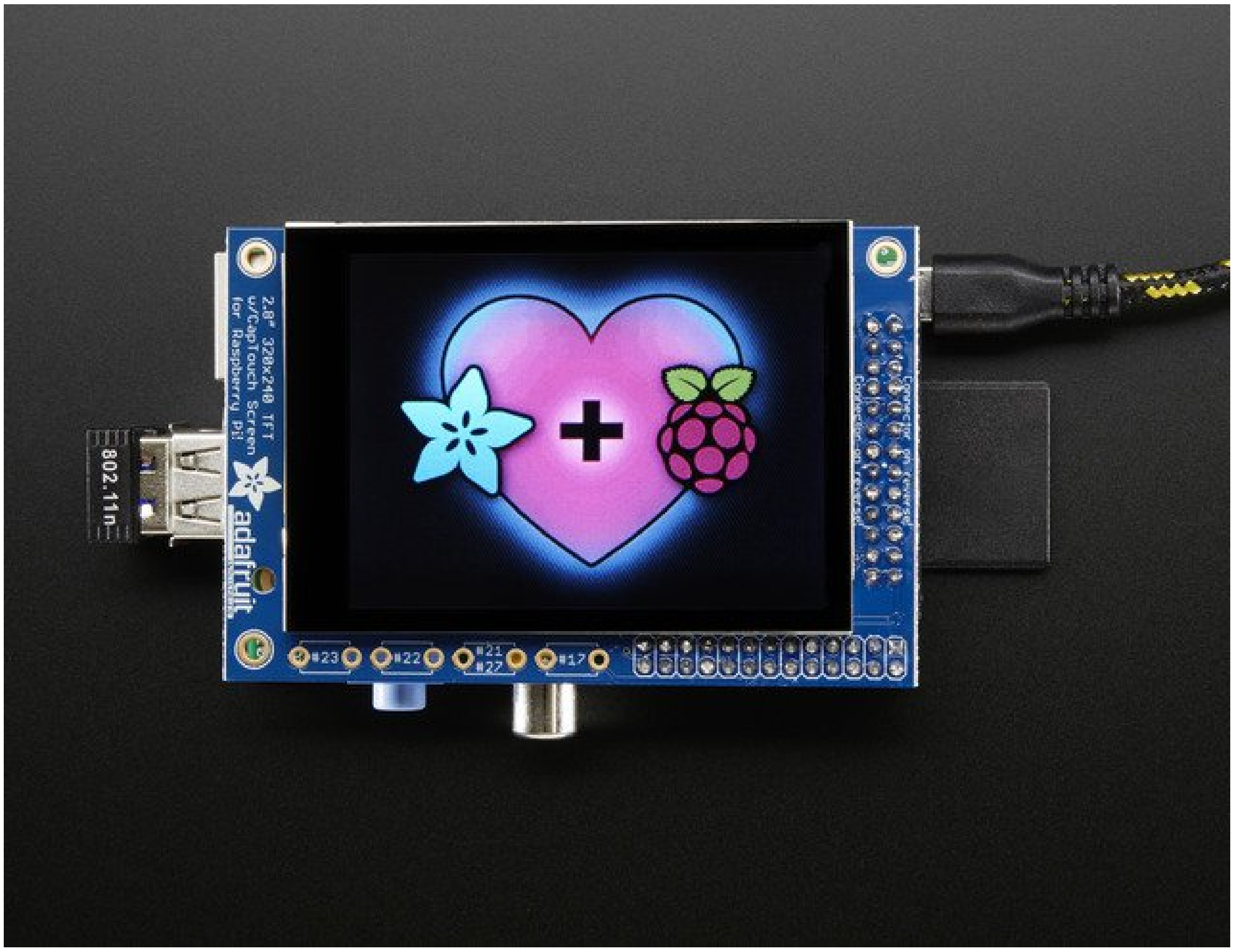
Guide Contents

Overview

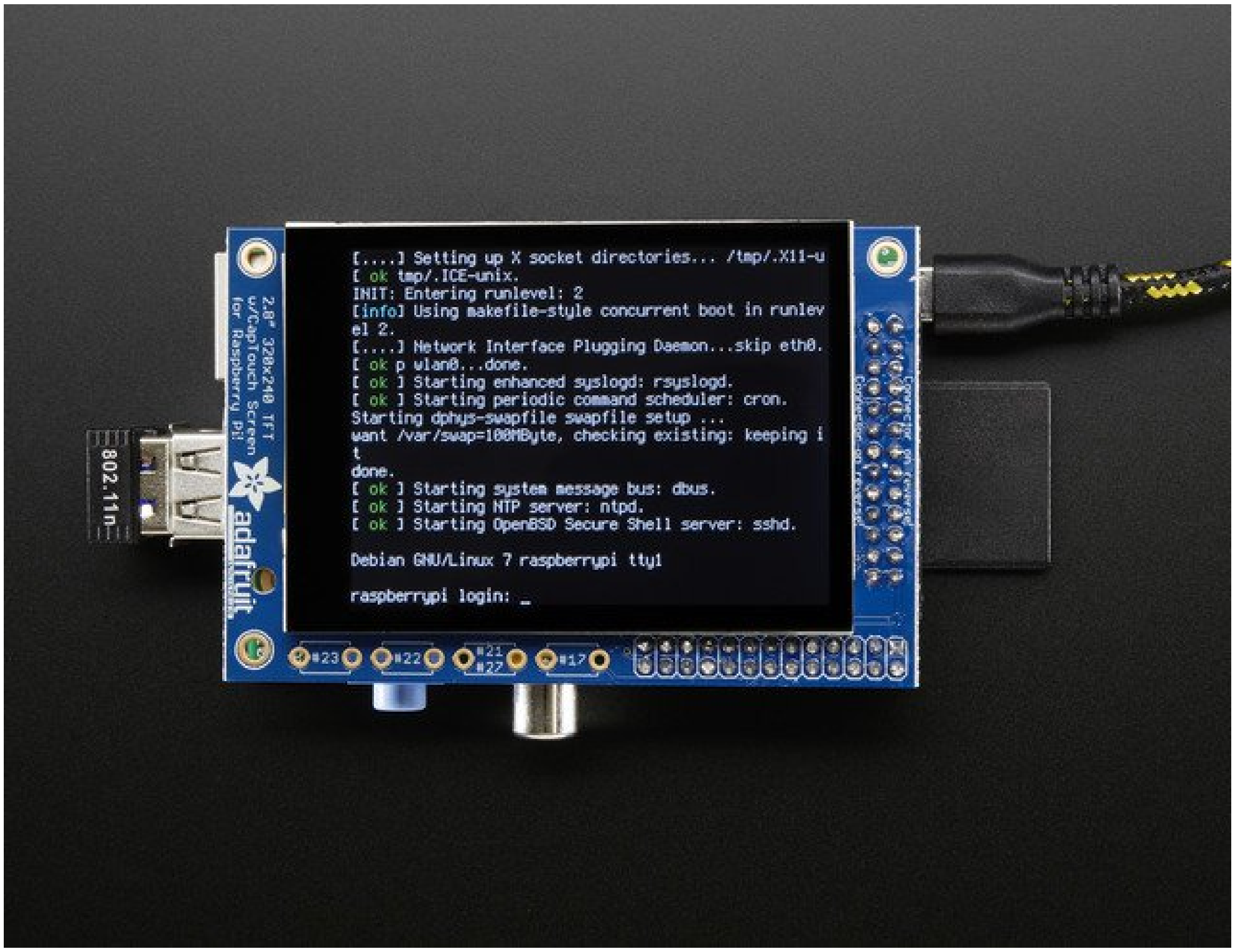


Our best-selling PiTFT just got a fancy upgrade, now we have a version with a **capacitive touchscreen!** That's right, instead of a resistive touchscreen, which requires a fingernail or stylus, you can now use a fingerpad. The screen looks much nicer, with a black bezel and glass overlay.

Featuring a 2.8" display with 320x240 16-bit color pixels and a capacitive touch overlay. The plate uses the high speed SPI interface on the Pi and can use the mini display as a console, X window port, displaying images or video etc. Best of all it plugs right in on top!



Uses the hardware I2C Pins (SDA & SCL), SPI pins (SCK, MOSI, MISO, CE0) as well as GPIO #25 and #24. All other GPIO are unused. Since we had a tiny bit of space, there's 4 spots for optional slim tactile switches wired to four GPIOs, that you can use if you want to make a basic user interface. For example, you can use one as a power on/off button. See below for the link to get the optional tact switches, they're not included.

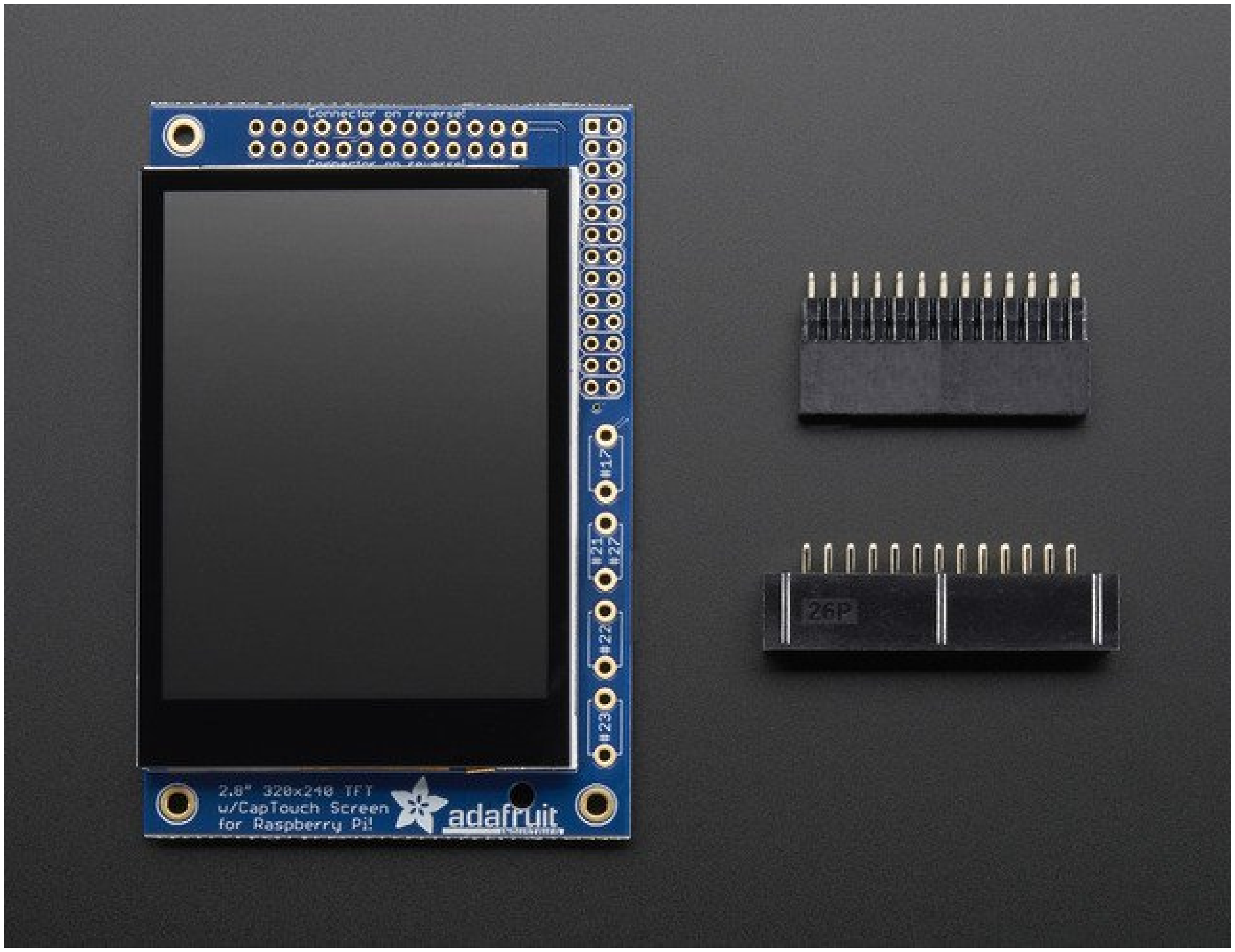


```
[....] Setting up X socket directories... /tmp/.X11-u
[ ok tmp/.ICE-unix.
INIT: Entering runlevel: 2
[info] Using makefile-style concurrent boot in runlev
el 2.
[....] Network Interface Plugging Daemon...skip eth0.
[ ok p wlan0...done.
[ ok ] Starting enhanced syslogd: rsyslogd.
[ ok ] Starting periodic command scheduler: cron.
Starting dphys-swapfile swapfile setup ...
want /var/swap=100MByte, checking existing: keeping i
t
done.
[ ok ] Starting system message bus: dbus.
[ ok ] Starting NTP server: ntpd.
[ ok ] Starting OpenBSD Secure Shell server: sshd.

Debian GNU/Linux 7 raspberrypi tty1

raspberrypi login: _
```

The screen is the same size as the resistive type so you can use this with the PiTFT PiBow or any other enclosure you may already have. We also use the same SDL device and signals so PyGame and X11 based programs can be swapped in with no changes in code.

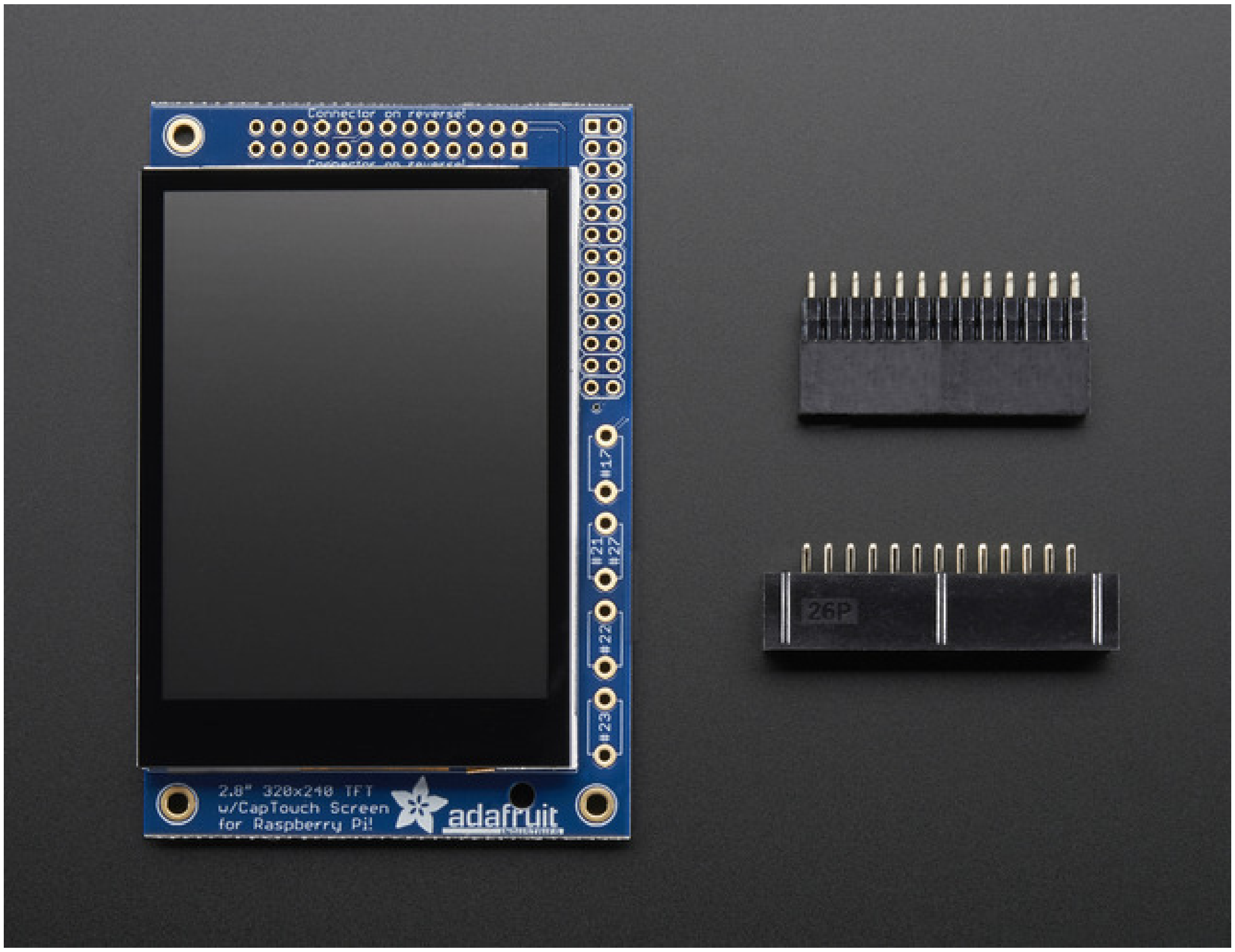
It's designed to fit nicely onto the Pi Model A or B rev 2 but also works perfectly fine with the Model B+ as long as you don't mind the PCB overhangs the USB ports by 5mm, see the photos above. Model B rev 1 have an older layout for the I2C pins and won't be able to use the touch screen

This version comes as a mini-kit, with a 2x13 extra-tall female header (to connect the plate to the Pi) and a 2x13 male header that can be used to connect an IDC cable or cobbler from the side. The photos above also show the optional installed slim tactile buttons. The tactile buttons are not included, but you can pick up a pack of 20 here. (http://adafru.it/1489) Some basic soldering is required to install the headers. You can also pick up an extra-long Pi stacking header if you want to install that instead of the 2x13 female header installed. (http://adafru.it/1112)


Connector on reverse!
#17
#21
#27
#22
#23
2.8" 320x240 TFT
w/CapTouch Screen
for Raspberry Pi!
adafruit
26P

Assembly

We are now selling these displays pre-assembled - skip this step if your PiTFT is not a mini-kit



This section is identical to the PiTFT Resistive 2.8" so please visit that page to complete assembly of this Pi Plate

Visit the 2.8" Resistive PiTFT Assembly Page

http://adafru.it/dDQ

Easy Install

The PiTFT requires kernel support and a couple other things to make it a nice stand-alone display. We have a detailed step-by-step setup for hackers who want to tweak, customize or understand the PiTFT setup. If you just want to get going, check out the following for easy-install instructions!

Ready to go image

If you want to start with a fresh image, we have two for Raspbian. There's the larger 'classic Jessie' image that will boot into X by default, and requires a 8G image, it has a lot more software installed. There's also the smaller 'Jessie Lite' that will boot into the command line, and can be burned onto a 2G card! Click below to download and install into a new SD card. Unzip and follow the classic SD card burning tutorials (http://adafru.it/aMW)

This image is customized for the CAPACITIVE touch 2.8" TFT, also known as PID #1983! Not for PID #1601
Download Jessie-based PiTFT 2.8" Capacitive Image for Pi 1, Pi 2 and Pi 3 (Sept 23, 2016)
http://adafru.it/saM
Download Jessie Lite-based PiTFT 2.8" Capacitive Image for Pi 1, Pi 2 and Pi 3 (Sept 23, 2016)
http://adafru.it/saN

Older images:

- Raspbian Jessie 2016-03-25-based image (http://adafru.it/mAc)
- Raspbian Jessie Lite 2016-03-25-based image (http://adafru.it/mAd)
- Raspbian Jessie 2015/09/24-based image (http://adafru.it/iDy)
- Raspbian Wheezy 2015/09/24-based image (http://adafru.it/idz)
- Raspbian 2014/09/18-based image (http://adafru.it/e11)
- Raspbian 2014/06/20-based image (http://adafru.it/dSO)
- Raspbian image from 2015/03/03 (http://adafru.it/eUI)

DIY Installer script

If you don't want to download an image, you can run our installation package helper from inside your existing Raspbian install. It will download the kernel add-ons, and configure your

Pi for PiTFT joy

The helper is available for perusal here (http://adafru.it/eIn) if you are interested in how it works

Step 1. Expand Filesystem

Start by expanding the filesystem **This is required!!!**

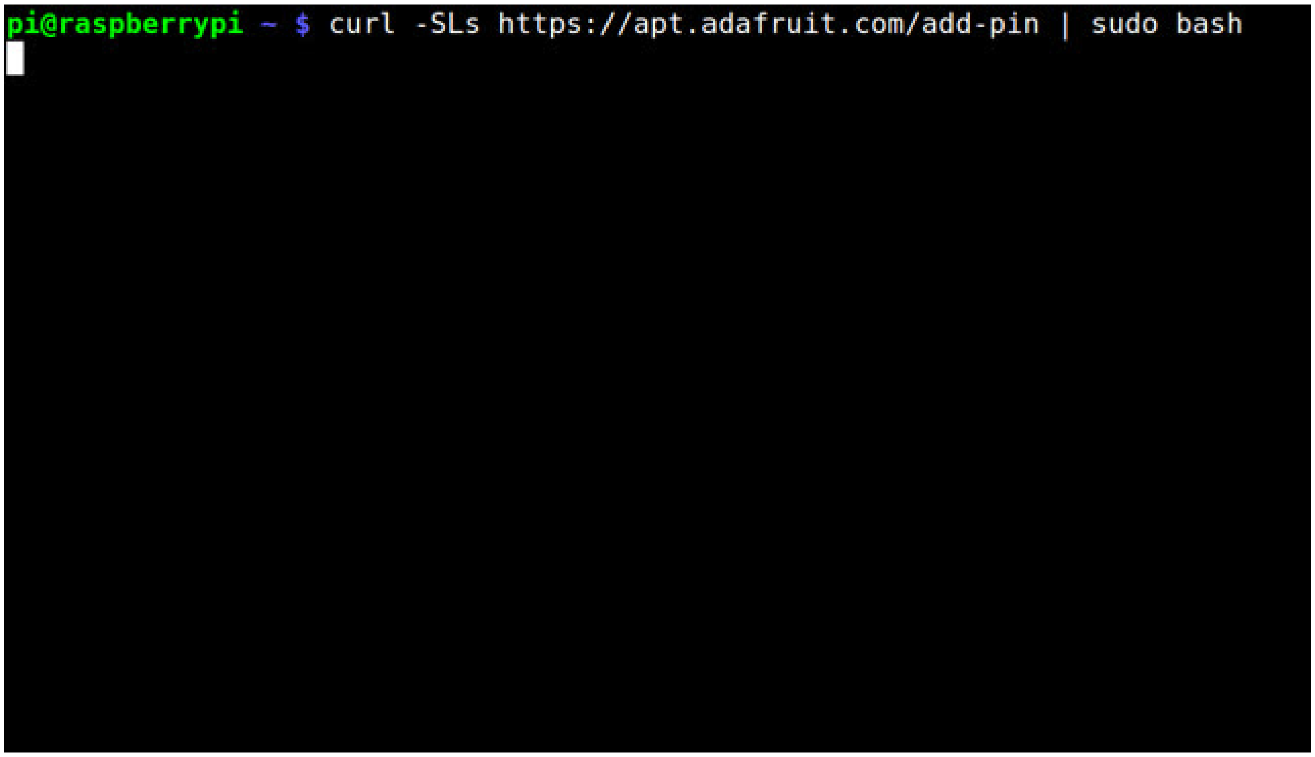
```
sudo raspi-config
(expand filesystem)
sudo reboot
```

Step 2. Install new Kernel

Then, once the filesys is expanded, download and install t he new kernel by running the following commands:

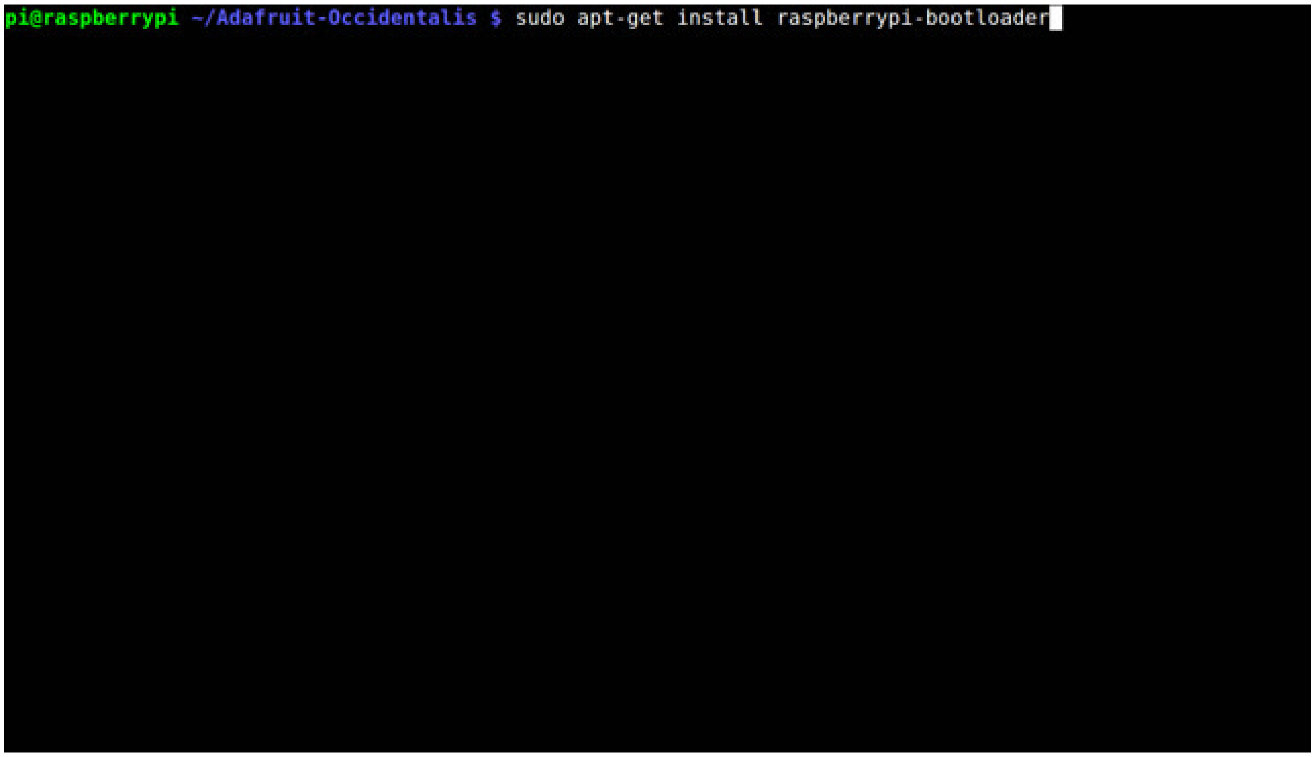
```
curl -SLs https://apt.adafruit.com/add-pin | sudo bash
sudo apt-get install -y raspberrypi-bootloader adafruit-pitft-helper raspberrypi-kernel
```

The first command adds **apt.adafruit.com** to your repository list, so you can grab code directly from Adafruit's servers, and tells apt that it should give a very high priority to packages installed there.



```
pi@raspberrypi ~ $ curl -SLs https://apt.adafruit.com/add-pin | sudo bash
```

The next line does the actual download and installation, it'll take a while because there's a lot of software to replace for PiTFT support.



```
pi@raspberrypi ~/Adafruit-Occidentalis $ sudo apt-get install raspberrypi-bootloader
```

It's normal for the Pi to pause and/or take a while at this step for many minutes, theres a lot of kernel software to replace

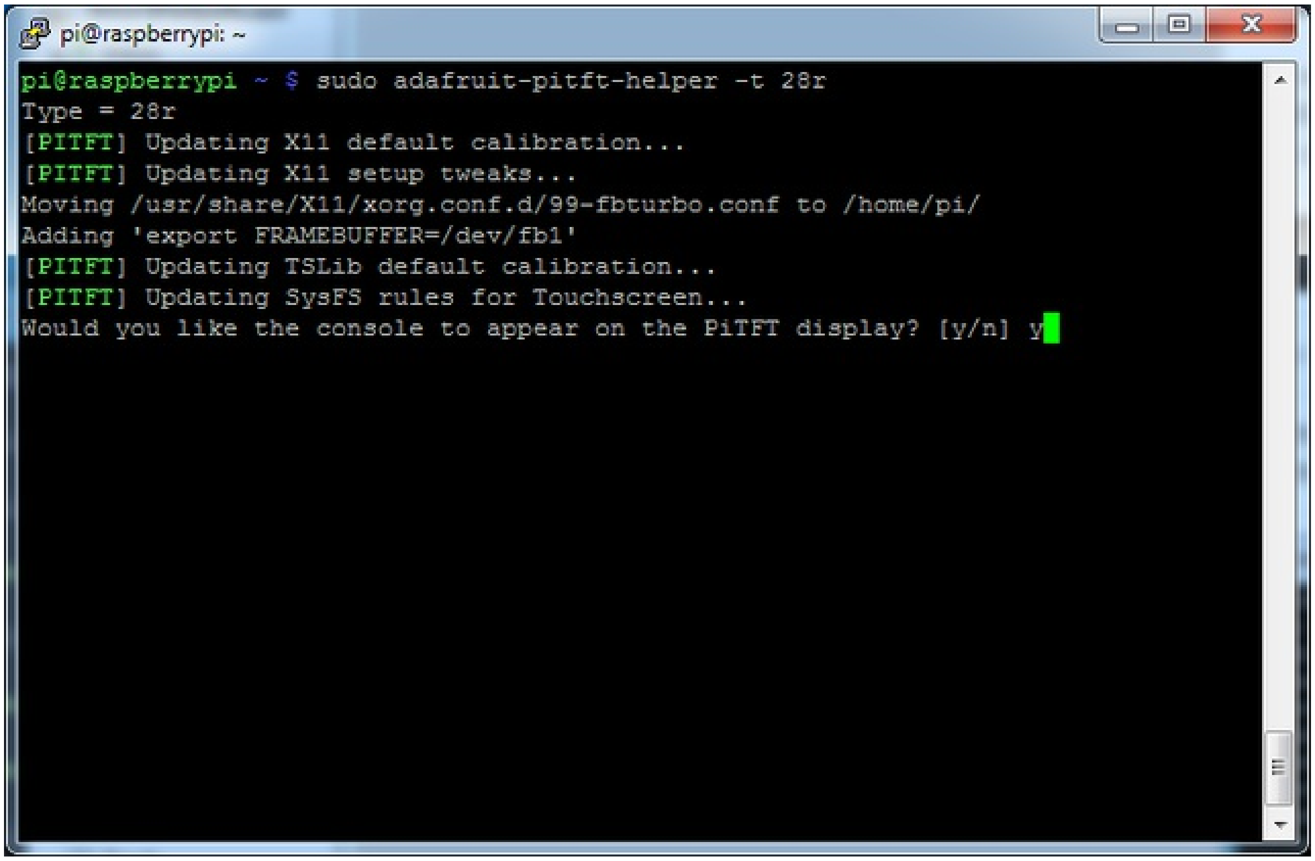
Step 3. Enable & Configure the PiTFT

OK now the kernel and helper are installed, all you have to do is run the helper which will configure the kernel device tree overlays and add the few configurations to make the console show up, etc.

```
sudo adafruit-pitft-helper -t 28c
```

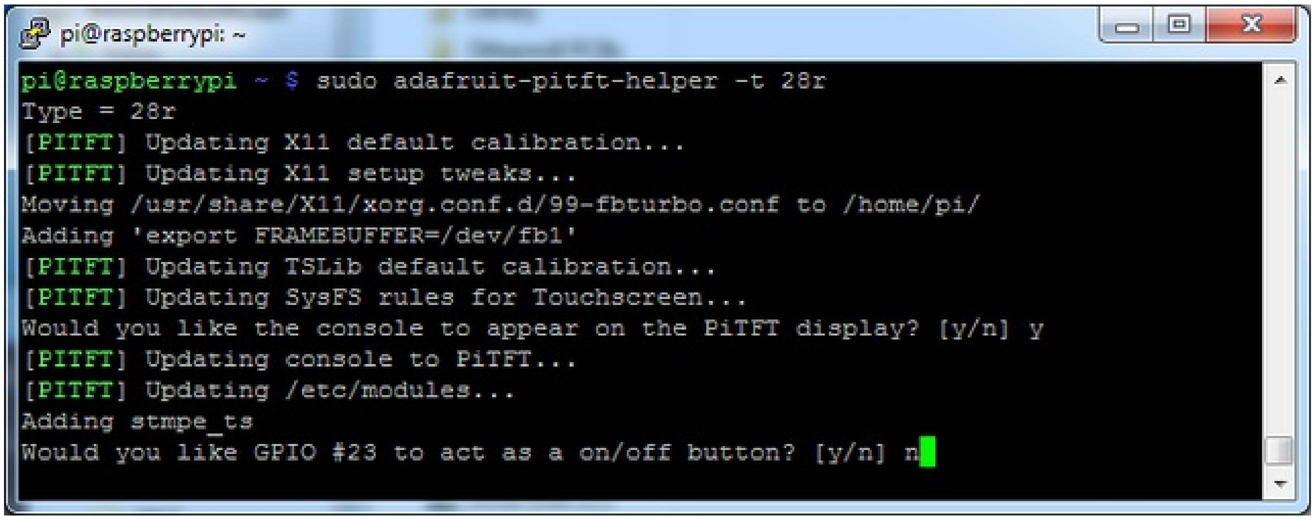
This will install the "2.8 Capacitive" type of PiTFT into the current install.

At the end you will be prompted on whether you want the text console to appear on the PiTFT. Answer Y or N depending on your personal desires!



```
pi@raspberrypi ~ $ sudo adafruit-pitft-helper -t 28r
Type = 28r
[PITFT] Updating X11 default calibration...
[PITFT] Updating X11 setup tweaks...
Moving /usr/share/X11/xorg.conf.d/99-fbturbo.conf to /home/pi/
Adding 'export FRAMEBUFFER=/dev/fb1'
[PITFT] Updating TSLib default calibration...
[PITFT] Updating SysFS rules for Touchscreen...
Would you like the console to appear on the PiTFT display? [y/n] y
```

You will also be prompted on whether you want one of the tactile buttons to act as an 'on off' switch. Answer Y or N depending on your personal desires!



```
pi@raspberrypi ~ $ sudo adafruit-pitft-helper -t 28r
Type = 28r
[PITFT] Updating X11 default calibration...
[PITFT] Updating X11 setup tweaks...
Moving /usr/share/X11/xorg.conf.d/99-fbturbo.conf to /home/pi/
Adding 'export FRAMEBUFFER=/dev/fb1'
[PITFT] Updating TSLib default calibration...
[PITFT] Updating SysFS rules for Touchscreen...
Would you like the console to appear on the PiTFT display? [y/n] y
[PITFT] Updating console to PiTFT...
[PITFT] Updating /etc/modules...
Adding stmpe_ts
Would you like GPIO #23 to act as a on/off button? [y/n] n
```

Thats it!

Run **sudo reboot** to try out your fancy new PiTFT :)

Detailed Installation

If you've grabbed our Easy Install image, or use the script, this step is not required, it's already done! This is just for advanced users who are curious on how to configure and customize the kernel install

In the next few steps we'll cover the**detailed** installation procedure. Chances are, you should grab the Easy Install image or script. If you have some interest in the details of how we install the PiTFT setup, read on!

In order to add support for the 2.8" TFT and capacitive touchscreen, we'll need to install a new Linux Kernel. Lucky for you, we created a kernel package that you can simply install *over* your current Raspbian (or Raspbian-derived) install instead of needing a whole new image. This makes it easier to keep your install up-to-date.

To use our kernel .deb files you must be using Raspbian or derivative. This wont work with Arch or other Linux flavors. As Raspbian is the official OS for the Pi, that's the only Linux we will support! Others can recompile their own kernel using our patchfile (http://adafru.it/cY2), but we have no tutorial or support or plans for such.

Before you start

You'll need a working install of Raspbian with network access.If you need help getting that far, check out our collection of Pi tutorials (http://adafru.it/aWq).

We'll be doing this from a console cable connection, but you can just as easily do it from the direct HDMI/TV console or by SSH'ing in. Whatever gets you to a shell will work!

Also, run **sudo apt-get update** !
To run these all the setup and config commands you'll need to be logged into a proper Terminal - use ssh, a console cable, or the main text console (on a TV). The WebIDE console may not work.

Download & Install Kernel

The only way we're distributing the PiTFT kernel packages right now is thru apt.adafruit.com so you'll still need to run:

```
curl -SLs https://apt.adafruit.com/add-pin | sudo bash
```

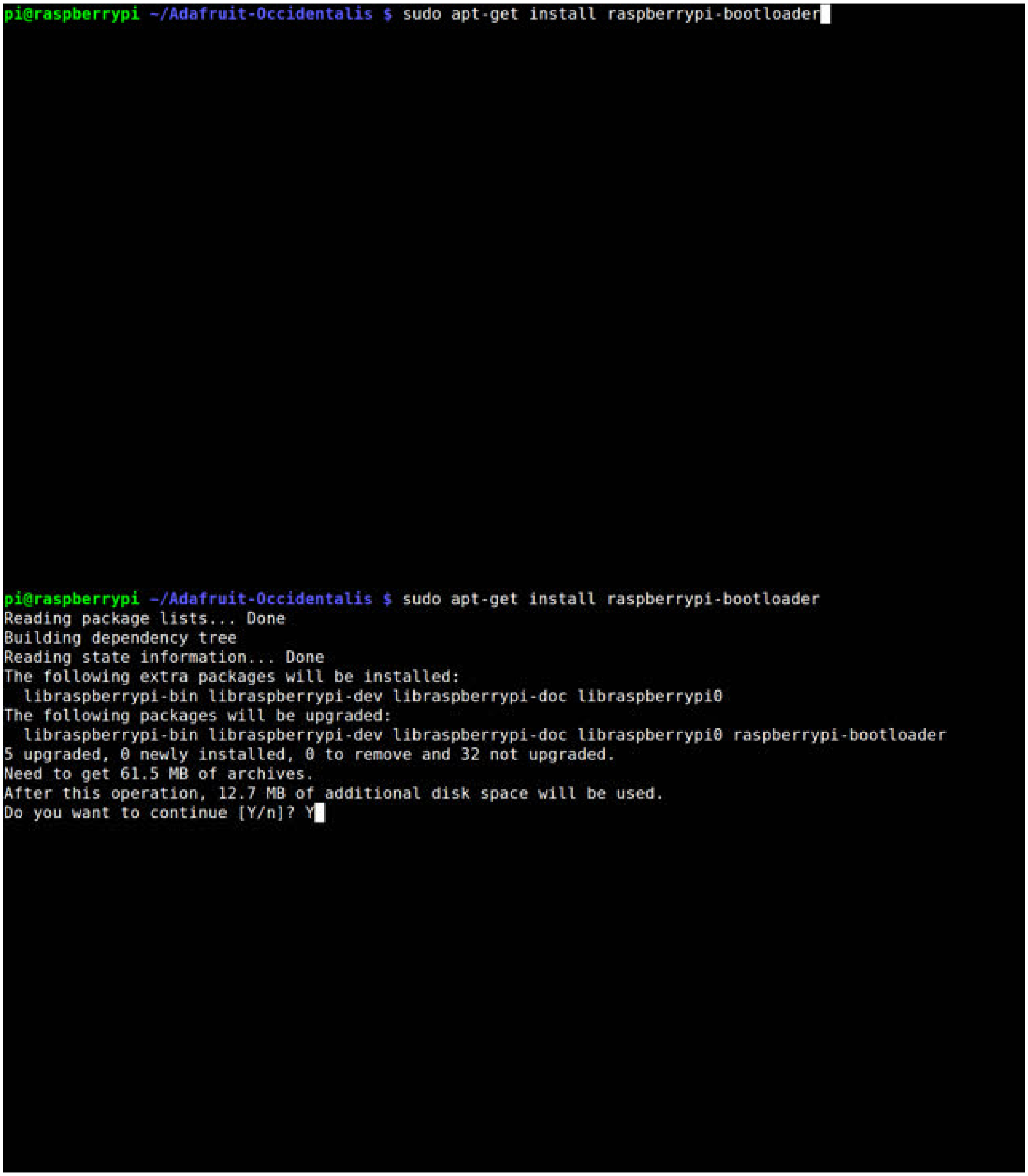
To add apt.adafruit.com to your list of software sources and make it the default source for packages it hosts

```
pi@raspberrypi ~ $ curl -SLs https://apt.adafruit.com/add-pin | sudo bash
```

Then install the kernel with

```
sudo apt-get install raspberrypi-bootloader
```

This will take a up to 20 minutes so go make a sandwich or coffee. It takes longer than it used to because there's now 2 kernels (v6 and v7 arm) and 2 kernel module directories.



```
pi@raspberrypi ~/Adafruit-Occidentalis $ sudo apt-get install raspberrypi-bootloader
```

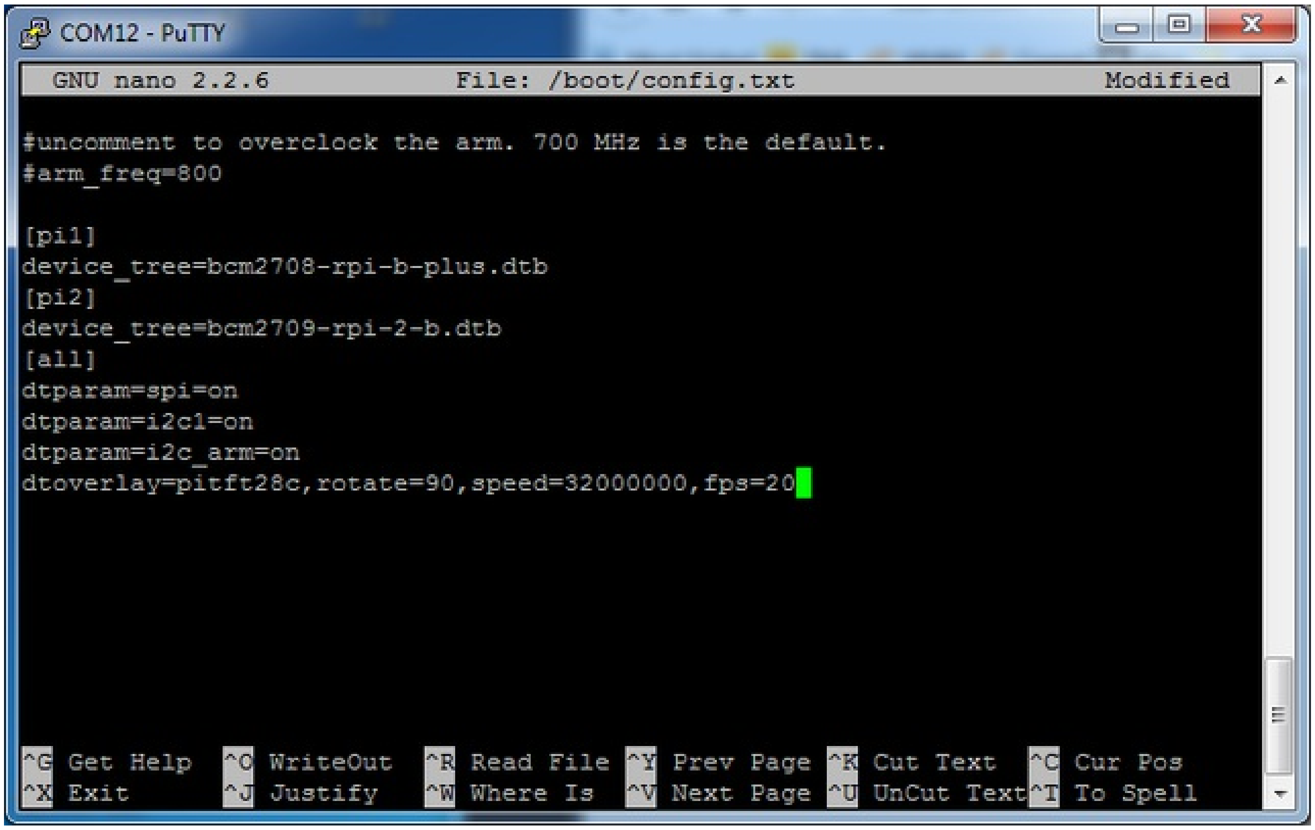
```
pi@raspberrypi ~/Adafruit-Occidentalis $ sudo apt-get install raspberrypi-bootloader
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc libraspberrypi0
The following packages will be upgraded:
  libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc libraspberrypi0 raspberrypi-bootloader
5 upgraded, 0 newly installed, 0 to remove and 32 not upgraded.
Need to get 61.5 MB of archives.
After this operation, 12.7 MB of additional disk space will be used.
Do you want to continue [Y/n]? Y
```

OK since you're not going to run the helper, lets add the device tree overlay manually. Edit /boot/config.txt with

sudo nano /boot/config.txt

and add the following lines at the end:

```
[pi1]
device_tree=bcm2708-rpi-b-plus.dtb
[pi2]
device_tree=bcm2709-rpi-2-b.dtb
[all]
dtparam=spi=on
dtparam=i2c1=on
dtparam=i2c_arm=on
dtoverlay=pitft28c,rotate=90,speed=32000000,fps=20
```



```
#uncomment to overclock the arm. 700 MHz is the default.
#arm_freq=800

[pi1]
device_tree=bcm2708-rpi-b-plus.dtb
[pi2]
device_tree=bcm2709-rpi-2-b.dtb
[all]
dtparam=spi=on
dtparam=i2c1=on
dtparam=i2c_arm=on
dtoverlay=pitft28c,rotate=90,speed=32000000,fps=20
```

The **rotate=** variable tells the driver to rotate the screen**0 90 180** or **270** degrees.
0 is portrait, with the bottom near the "Adafruit Logo"
90 is landscape, with the bottom of the screen near the buttons.
180 is portrait, with the top near the "Adafruit Logo"
270 is landscape, with the top of the screen near the buttons.
You can change this file with**nano** and reboot to make the change stick.

The **speed=** variable tells the driver how to fast to drive the display. 32MHz (**32000000**) is a pretty nice 20 FPS rate but if your screen is acting funny, try taking it down to 16MHz (**16000000**)

Save the file. Now we'll just reboot to let it all sink in.

sudo shutdown -h now (if you don't have the TFT installed, shutdown, place the TFT on the Pi and re-power)

or

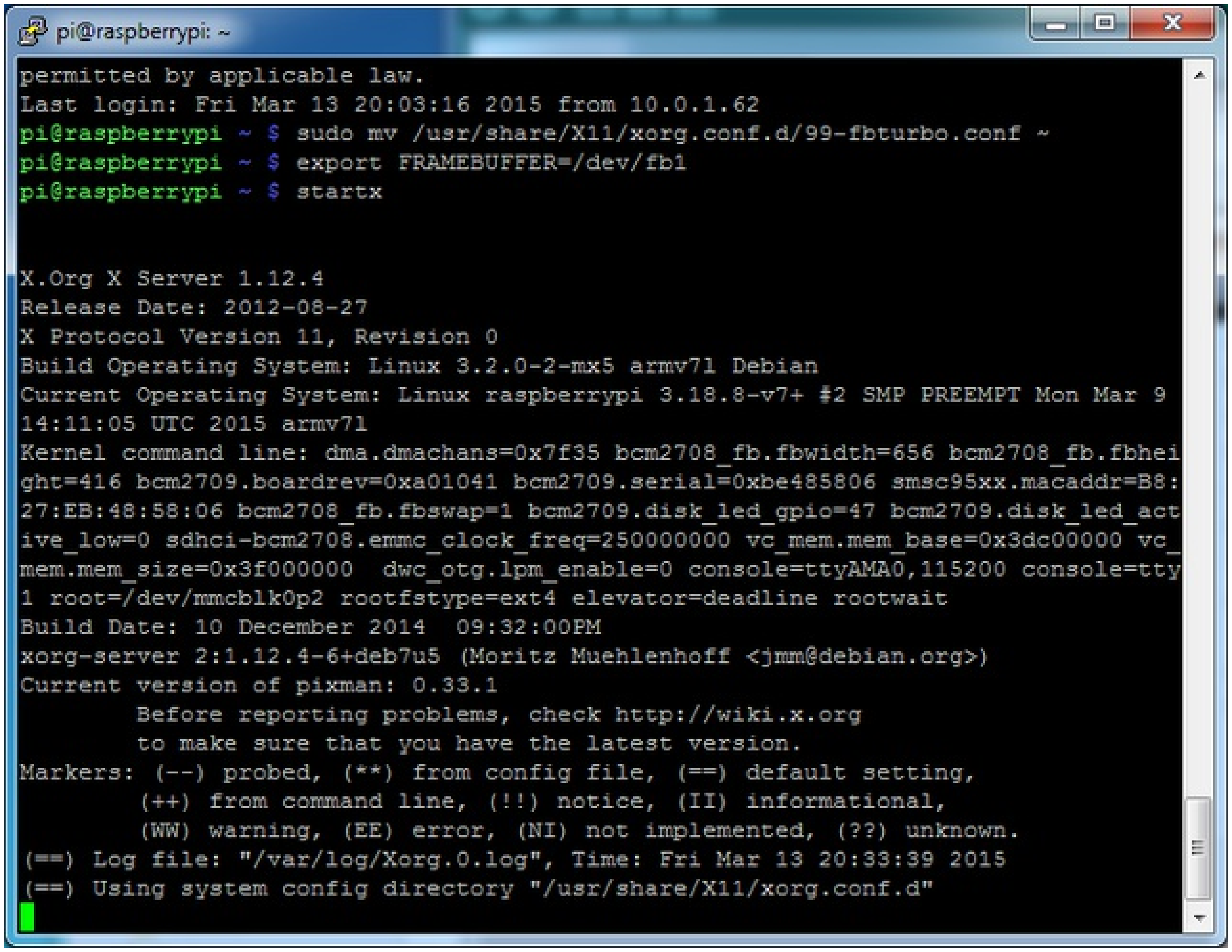
sudo reboot (if you have the TFT plate installed already)

When the Pi restarts, the attached PiTFT should start out all white and then turn black. That means the kernel found the display and cleared the screen. If the screen did not turn black, that means that likely there's something up with your connection or kernel install. Solder anything that needs resoldering!

Now that you're rebooted, log back in on the console/TV/SSH. There's nothing displayed on the screen yet, we'll do a test to make sure everything is perfect first!

Run the following commands to startx on the/**dev/fb1** framebuffer, a.k.a PiTFT screen:

```
sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~
export FRAMEBUFFER=/dev/fb1
startx
```



```
permitted by applicable law.
Last login: Fri Mar 13 20:03:16 2015 from 10.0.1.62
pi@raspberrypi ~ $ sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~
pi@raspberrypi ~ $ export FRAMEBUFFER=/dev/fb1
pi@raspberrypi ~ $ startx


X.Org X Server 1.12.4
Release Date: 2012-08-27
X Protocol Version 11, Revision 0
Build Operating System: Linux 3.2.0-2-mx5 armv7l Debian
Current Operating System: Linux raspberrypi 3.18.8-v7+ #2 SMP PREEMPT Mon Mar 9
14:11:05 UTC 2015 armv7l
Kernel command line: dma.dmachans=0x7f35 bcm2708_fb.fbwidth=656 bcm2708_fb.fbhei
ght=416 bcm2709.boardrev=0xa01041 bcm2709.serial=0xbe485806 smsc95xx.macaddr=B8:
27:EB:48:58:06 bcm2708_fb.fbswap=1 bcm2709.disk_led_gpio=47 bcm2709.disk_led_act
ive_low=0 sdhci-bcm2708.emmc_clock_freq=250000000 vc_mem.mem_base=0x3dc00000 vc_
mem.mem_size=0x3f000000  dwc_otg.lpm_enable=0 console=ttyAMA0,115200 console=tty
1 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait
Build Date: 10 December 2014  09:32:00PM
xorg-server 2:1.12.4-6+deb7u5 (Moritz Muehlenhoff <jmm@debian.org>)
Current version of pixman: 0.33.1
        Before reporting problems, check http://wiki.x.org
        to make sure that you have the latest version.
Markers: (--) probed, (**) from config file, (==) default setting,
        (++) from command line, (!!) notice, (II) informational,
        (WW) warning, (EE) error, (NI) not implemented, (??) unknown.
(==) Log file: "/var/log/Xorg.0.log", Time: Fri Mar 13 20:33:39 2015
(==) Using system config directory "/usr/share/X11/xorg.conf.d"
```

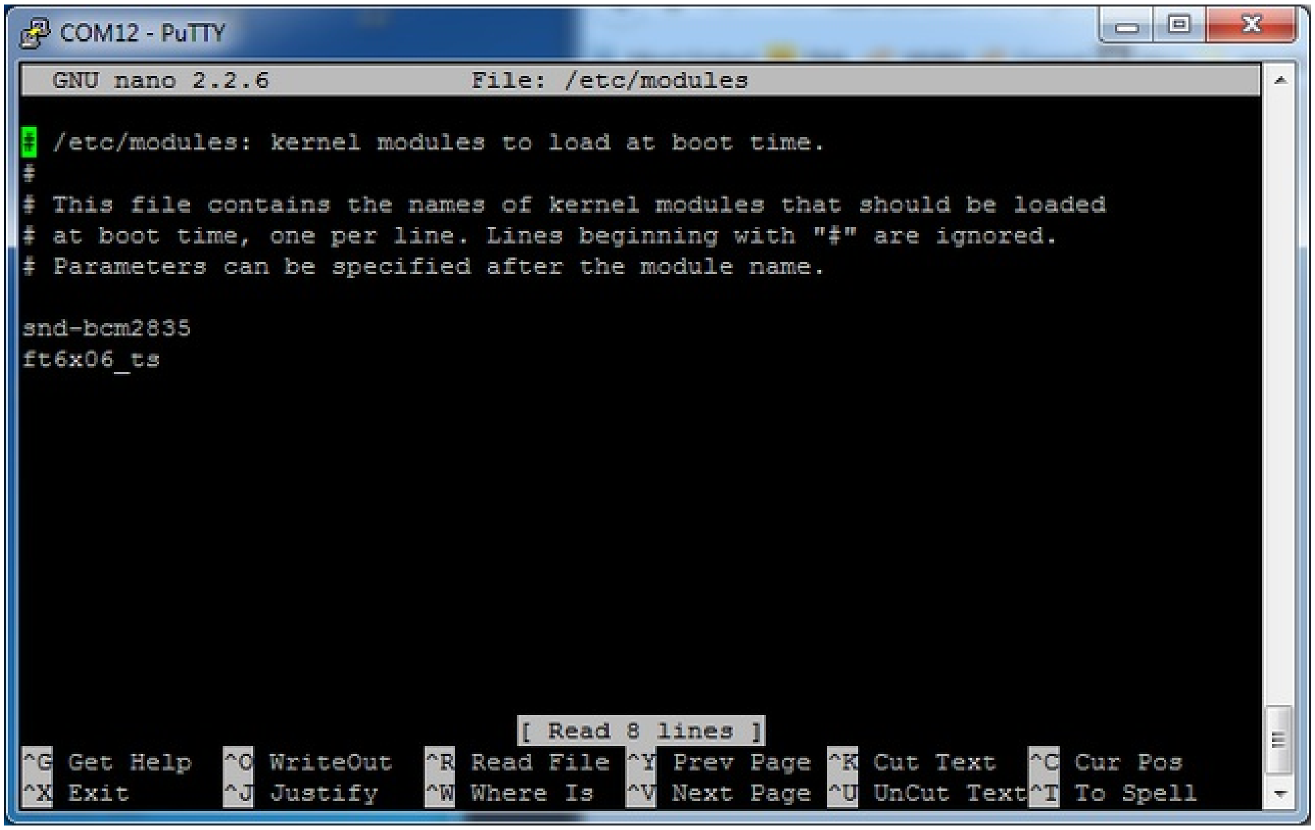
You should see the Pi desktop show up on the TFT! Congrats, you've completed the first test perfectly.

Hit Control-C in the console to quit the X server so we can continue configuration

Next up we'll add support for the touch screen automatically on boot. Edit the module list with

sudo nano /etc/modules

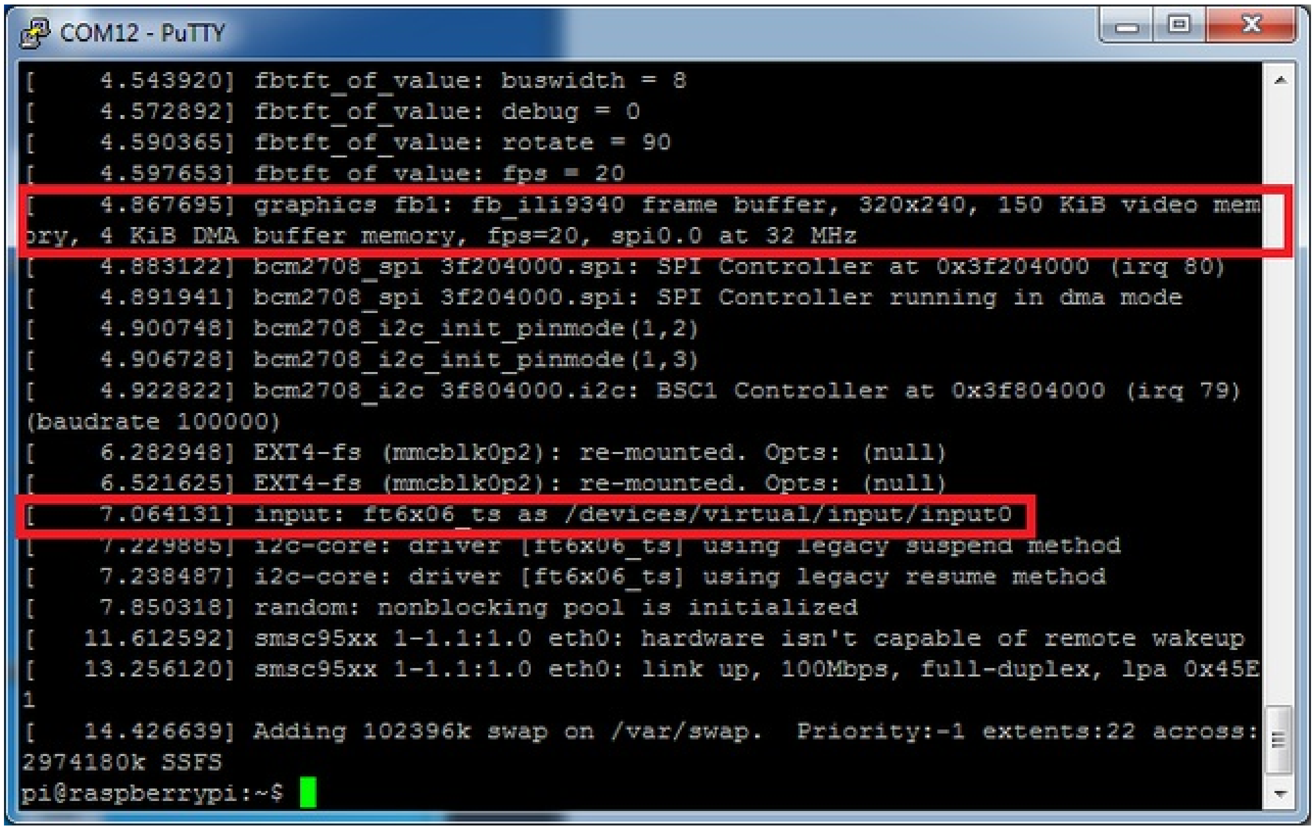
and add **ft6x06_ts** on a line at the end



COM12 - PuTTY

```
  GNU nano 2.2.6            File: /etc/modules

# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
ft6x06_ts




                           [ Read 8 lines ]
^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Save the file and reboot the Pi with**sudo reboot** and look at the console output (or run **dmesg** in the console window after logging in) you will see the modules install. Look in particular for the FT6206 (a.k.a. FT6x06) detection and the ILI9340 screen frequency as highlighted here



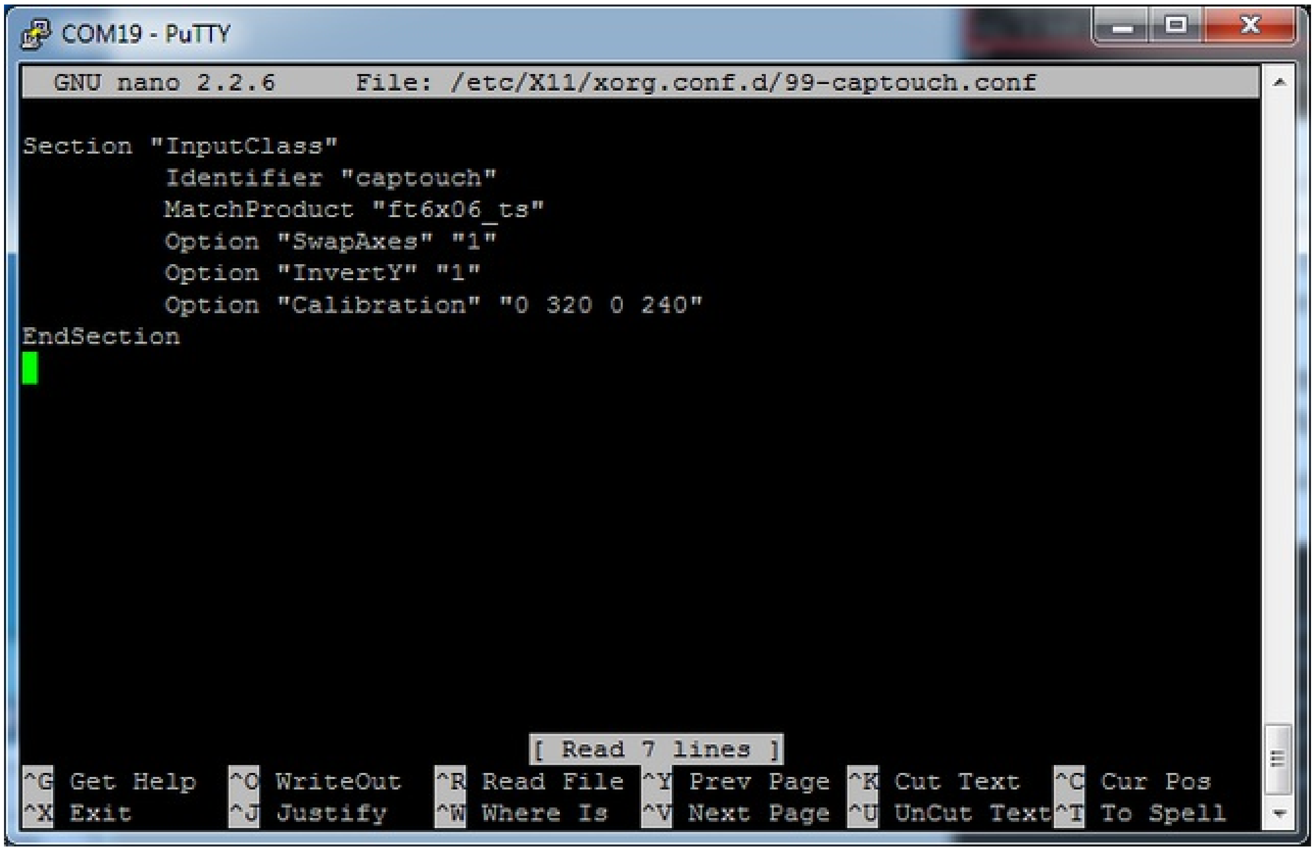
```
[    4.543920] fbtft_of_value: buswidth = 8
[    4.572892] fbtft_of_value: debug = 0
[    4.590365] fbtft_of_value: rotate = 90
[    4.597653] fbtft_of_value: fps = 20
[    4.867695] graphics fb1: fb_ili9340 frame buffer, 320x240, 150 KiB video mem
ory, 4 KiB DMA buffer memory, fps=20, spi0.0 at 32 MHz
[    4.883122] bcm2708_spi 3f204000.spi: SPI Controller at 0x3f204000 (irq 80)
[    4.891941] bcm2708_spi 3f204000.spi: SPI Controller running in dma mode
[    4.900748] bcm2708_i2c_init_pinmode(1,2)
[    4.906728] bcm2708_i2c_init_pinmode(1,3)
[    4.922822] bcm2708_i2c 3f804000.i2c: BSC1 Controller at 0x3f804000 (irq 79)
(baudrate 100000)
[    6.282948] EXT4-fs (mmcblk0p2): re-mounted. Opts: (null)
[    6.521625] EXT4-fs (mmcblk0p2): re-mounted. Opts: (null)
[    7.064131] input: ft6x06_ts as /devices/virtual/input/input0
[    7.229885] i2c-core: driver [ft6x06_ts] using legacy suspend method
[    7.238487] i2c-core: driver [ft6x06_ts] using legacy resume method
[    7.850318] random: nonblocking pool is initialized
[   11.612592] smsc95xx 1-1.1:1.0 eth0: hardware isn't capable of remote wakeup
[   13.256120] smsc95xx 1-1.1:1.0 eth0: link up, 100Mbps, full-duplex, lpa 0x45E
1
[   14.426639] Adding 102396k swap on /var/swap.  Priority:-1 extents:22 across:
2974180k SSFS
pi@raspberrypi:~$
```

We can set up the touchscreen for **rotate=90** configuration by doing the following (for more delicate calibration or for other rotate=XX values, see the next section)
Create the directory and new calibration configuration file:

```
sudo mkdir /etc/X11/xorg.conf.d
sudo nano /etc/X11/xorg.conf.d/99-captouch.conf
```

and enter in the following lines, then save.

```
Section "InputClass"
        Identifier "captouch"
        MatchProduct "ft6x06_ts"
        Option "SwapAxes" "1"
        Option "InvertY" "1"
        Option "Calibration" "0 320 0 240"
EndSection
```



COM19 - PuTTY

```
  GNU nano 2.2.6      File: /etc/X11/xorg.conf.d/99-captouch.conf

Section "InputClass"
         Identifier "captouch"
         MatchProduct "ft6x06_ts"
         Option "SwapAxes" "1"
         Option "InvertY" "1"
         Option "Calibration" "0 320 0 240"
EndSection

                              [ Read 7 lines ]
^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

You can now try to run X again with

```
FRAMEBUFFER=/dev/fb1 startx
```

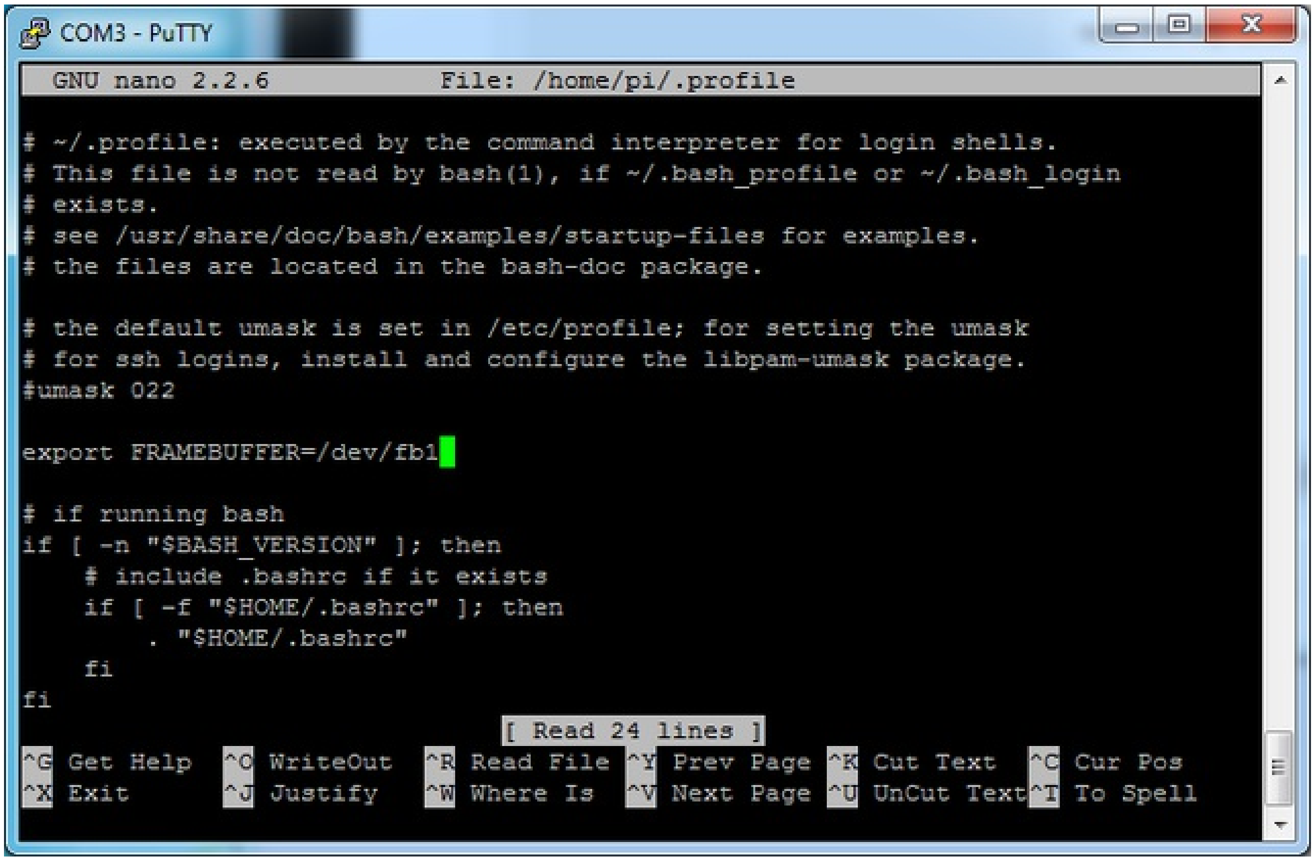
The touchscreen now works, you can try it out!

Type Control-C to quit **X**

If you don't ever want to have to type FRAMEBUFFER=/dev/fb1 before startx, you can make it a default state by editing your profile file: **sudo nano ~/.profile** and adding

```
export FRAMEBUFFER=/dev/fb1
```

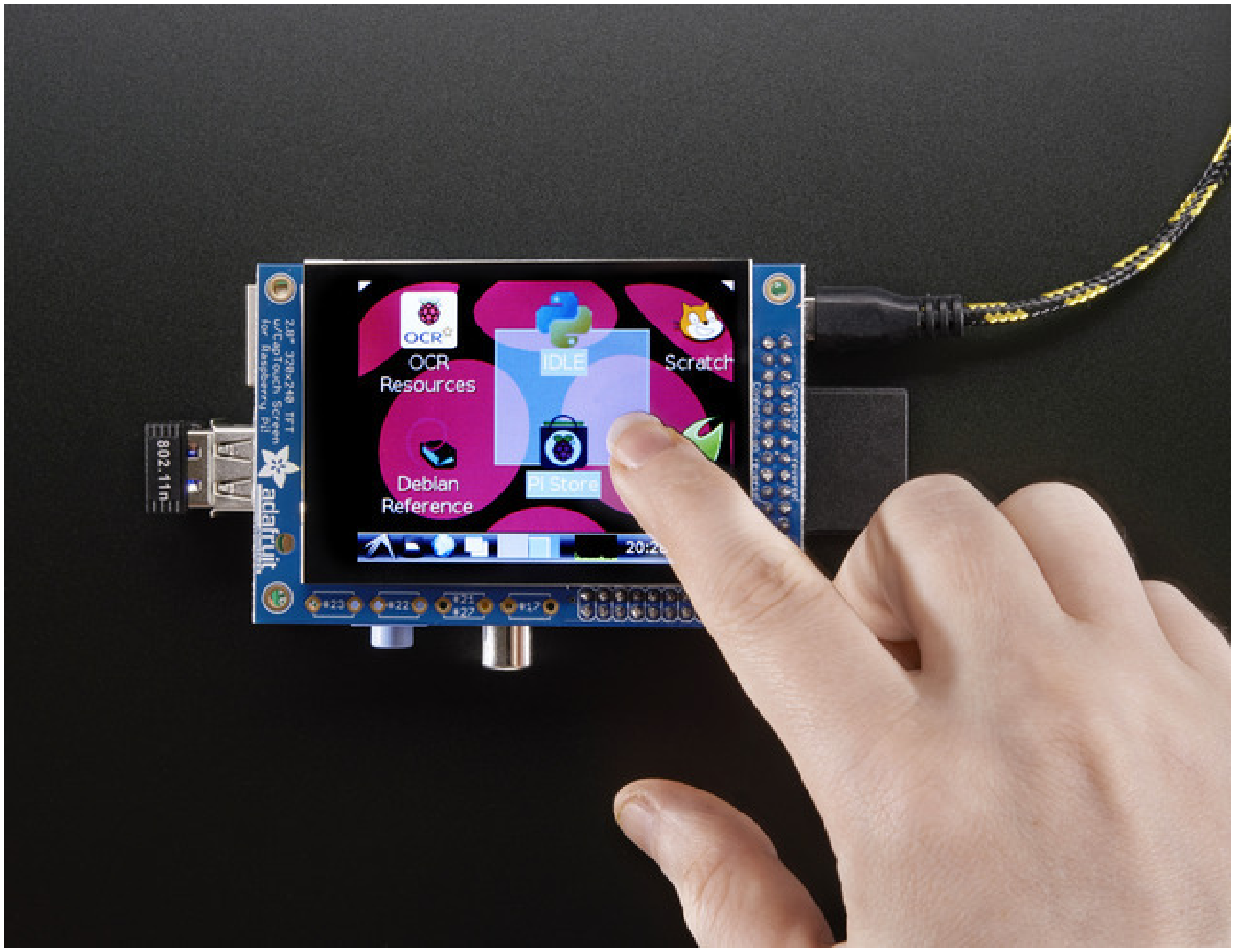
near the top and saving the file. Then reboot to reload the profile file. It will now always assume you want to use /dev/fb1



```
  GNU nano 2.2.6           File: /home/pi/.profile

# ~/.profile: executed by the command interpreter for login shells.
# This file is not read by bash(1), if ~/.bash_profile or ~/.bash_login
# exists.
# see /usr/share/doc/bash/examples/startup-files for examples.
# the files are located in the bash-doc package.

# the default umask is set in /etc/profile; for setting the umask
# for ssh logins, install and configure the libpam-umask package.
#umask 022

export FRAMEBUFFER=/dev/fb1

# if running bash
if [ -n "$BASH_VERSION" ]; then
    # include .bashrc if it exists
    if [ -f "$HOME/.bashrc" ]; then
        . "$HOME/.bashrc"
    fi
fi
                               [ Read 24 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Capacitive Touchscreen Configuration

If you've grabbed our Easy Install image, or use the script, this step is not required, it's already done! This is just for advanced users who are curious on how to configure and customize the touchscreen



The nifty thing about capacitive touch screens is that they **do not require calibration**! The calibration is done 'in chip' on the screen itself. However, we still do need to tell the Pi how to read the capacitive chip.

Before we start, we'll make a **udev** rule for the touchscreen. That's because the **eventX** name of the device will change a lot and its annoying to figure out what its called depending on whether you have a keyboard or other mouse installed.

First up figure out if you have the FT62X6 driver or FT6236 driver by running **dmesg | grep ft6**

If you are running FT6236 driver

Run

```
sudo nano /etc/udev/rules.d/95-ft6236.rules
```

to create a new **udev** file and copy & paste the following line in:

```
SUBSYSTEM=="input", ATTRS{name}=="ft6236", ENV{DEVNAME}=="*event*",
SYMLINK+="input/touchscreen"
```

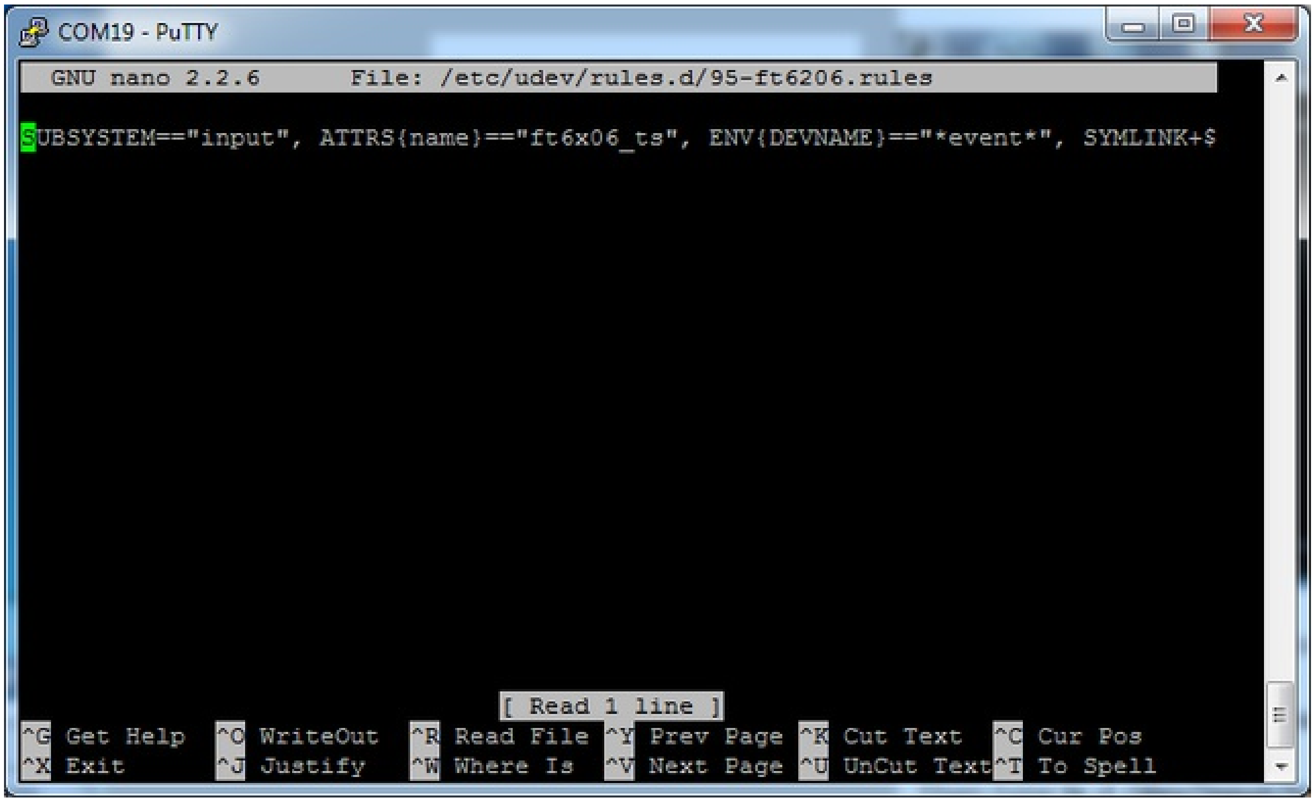
If you are running FT6X06 driver

Run

```
sudo nano /etc/udev/rules.d/95-ft6206.rules
```

to create a new **udev** file and copy & paste the following line in:

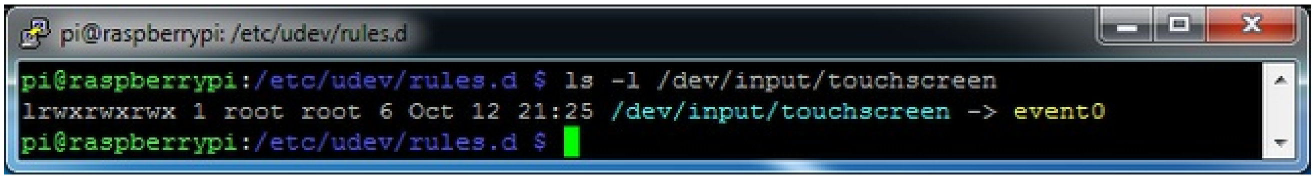
```
SUBSYSTEM=="input", ATTRS{name}=="ft6x06_ts", ENV{DEVNAME}=="*event*",
SYMLINK+="input/touchscreen"
```



```
  GNU nano 2.2.6       File: /etc/udev/rules.d/95-ft6206.rules

SUBSYSTEM=="input", ATTRS{name}=="ft6x06_ts", ENV{DEVNAME}=="*event*", SYMLINK+$

                            [ Read 1 line ]
^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

Reboot the Pi with **sudo reboot**

Then type ls -l /dev/input/touchscreen
It should point to **eventX** where X is some number, that number will be different on different setups since other keyboards/mice/USB devices will take up an event slot



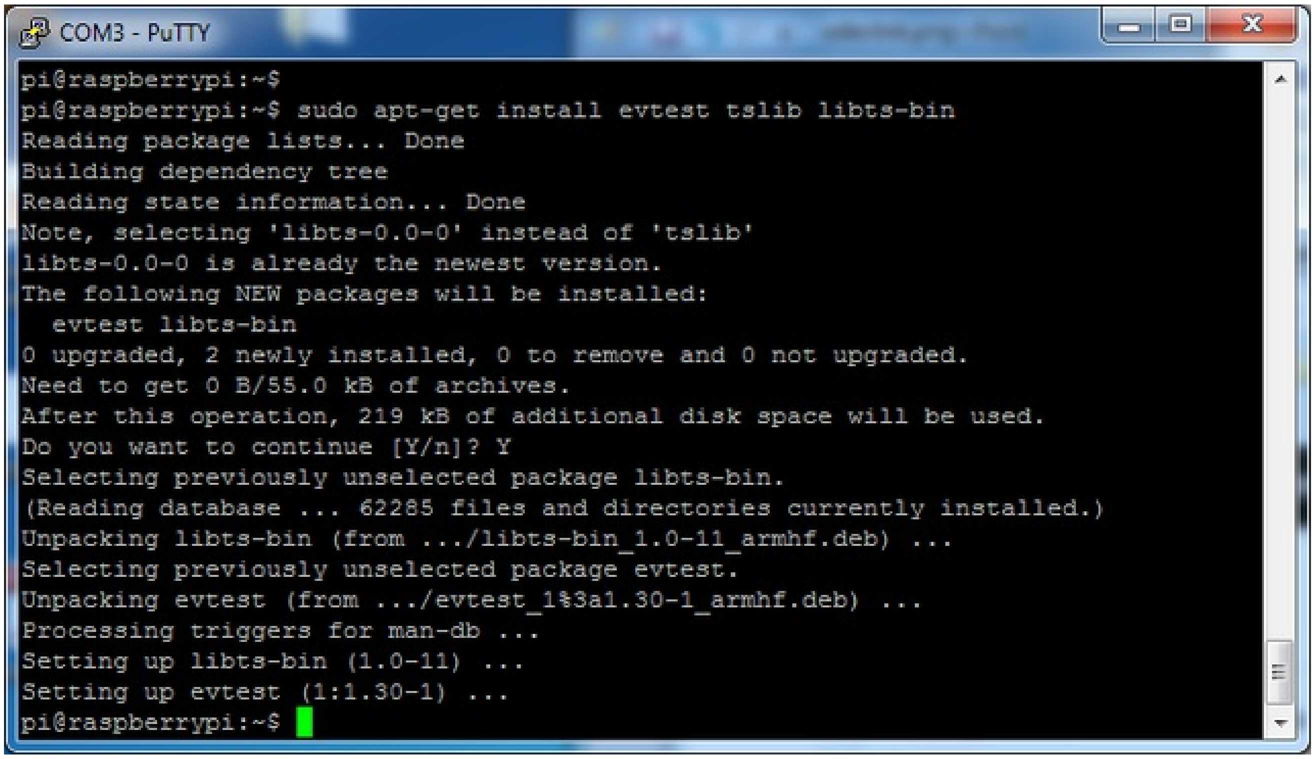
```
pi@raspberrypi:/etc/udev/rules.d $ ls -l /dev/input/touchscreen
lrwxrwxrwx 1 root root 6 Oct 12 21:25 /dev/input/touchscreen -> event0
pi@raspberrypi:/etc/udev/rules.d $
```

(http://adafru.it/dIX)

Event Testing

Even though capacitive touch screens don't require calibration, there are some useful tools we can use to debug the touchscreen. Install the "event test" and "touchscreen library" packages with

```
sudo apt-get install evtest tslib libts-bin
```

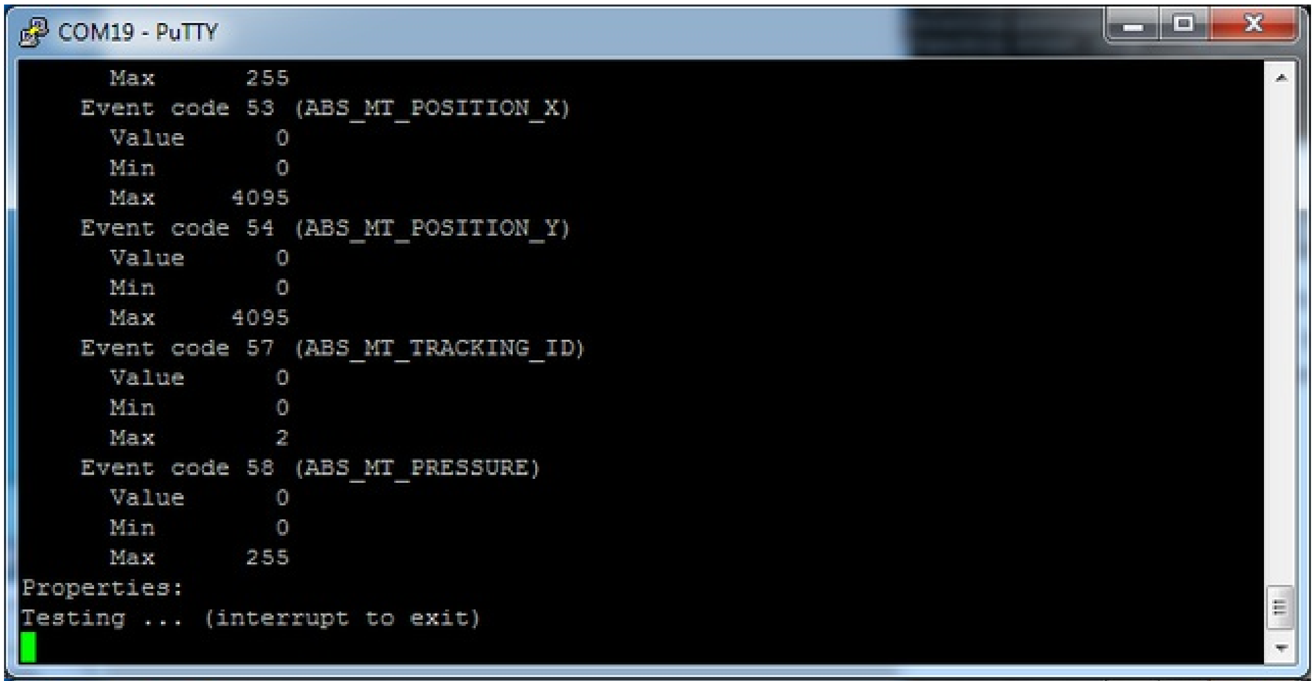


```
pi@raspberrypi:~$
pi@raspberrypi:~$ sudo apt-get install evtest tslib libts-bin
Reading package lists... Done
Building dependency tree
Reading state information... Done
Note, selecting 'libts-0.0-0' instead of 'tslib'
libts-0.0-0 is already the newest version.
The following NEW packages will be installed:
  evtest libts-bin
0 upgraded, 2 newly installed, 0 to remove and 0 not upgraded.
Need to get 0 B/55.0 kB of archives.
After this operation, 219 kB of additional disk space will be used.
Do you want to continue [Y/n]? Y
Selecting previously unselected package libts-bin.
(Reading database ... 62285 files and directories currently installed.)
Unpacking libts-bin (from .../libts-bin_1.0-11_armhf.deb) ...
Selecting previously unselected package evtest.
Unpacking evtest (from .../evtest_1%3a1.30-1_armhf.deb) ...
Processing triggers for man-db ...
Setting up libts-bin (1.0-11) ...
Setting up evtest (1:1.30-1) ...
pi@raspberrypi:~$
```

Now you can use some tools such as

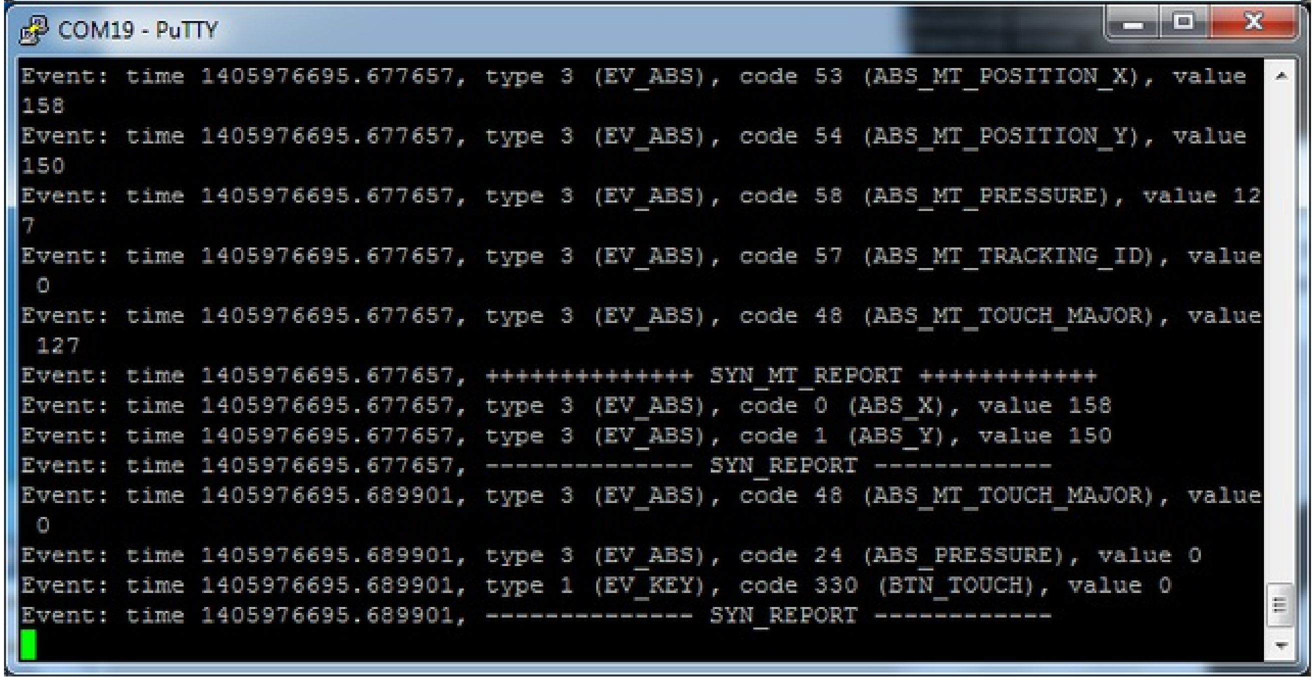
```
sudo evtest /dev/input/touchscreen
```

which will let you see touchscreen events in real time, press on the touchscreen to see the reports.



COM19 - PuTTY

```
      Max      255
    Event code 53 (ABS_MT_POSITION_X)
      Value      0
      Min        0
      Max     4095
    Event code 54 (ABS_MT_POSITION_Y)
      Value      0
      Min        0
      Max     4095
    Event code 57 (ABS_MT_TRACKING_ID)
      Value      0
      Min        0
      Max        2
    Event code 58 (ABS_MT_PRESSURE)
      Value      0
      Min        0
      Max      255
Properties:
Testing ... (interrupt to exit)
```



COM19 - PuTTY

```
Event: time 1405976695.677657, type 3 (EV_ABS), code 53 (ABS_MT_POSITION_X), value
158
Event: time 1405976695.677657, type 3 (EV_ABS), code 54 (ABS_MT_POSITION_Y), value
150
Event: time 1405976695.677657, type 3 (EV_ABS), code 58 (ABS_MT_PRESSURE), value 12
7
Event: time 1405976695.677657, type 3 (EV_ABS), code 57 (ABS_MT_TRACKING_ID), value
 0
Event: time 1405976695.677657, type 3 (EV_ABS), code 48 (ABS_MT_TOUCH_MAJOR), value
 127
Event: time 1405976695.677657, ++++++++++++++ SYN_MT_REPORT ++++++++++++
Event: time 1405976695.677657, type 3 (EV_ABS), code 0 (ABS_X), value 158
Event: time 1405976695.677657, type 3 (EV_ABS), code 1 (ABS_Y), value 150
Event: time 1405976695.677657, -------------- SYN_REPORT ------------
Event: time 1405976695.689901, type 3 (EV_ABS), code 48 (ABS_MT_TOUCH_MAJOR), value
 0
Event: time 1405976695.689901, type 3 (EV_ABS), code 24 (ABS_PRESSURE), value 0
Event: time 1405976695.689901, type 1 (EV_KEY), code 330 (BTN_TOUCH), value 0
Event: time 1405976695.689901, -------------- SYN_REPORT ------------
```

TSLIB calibration

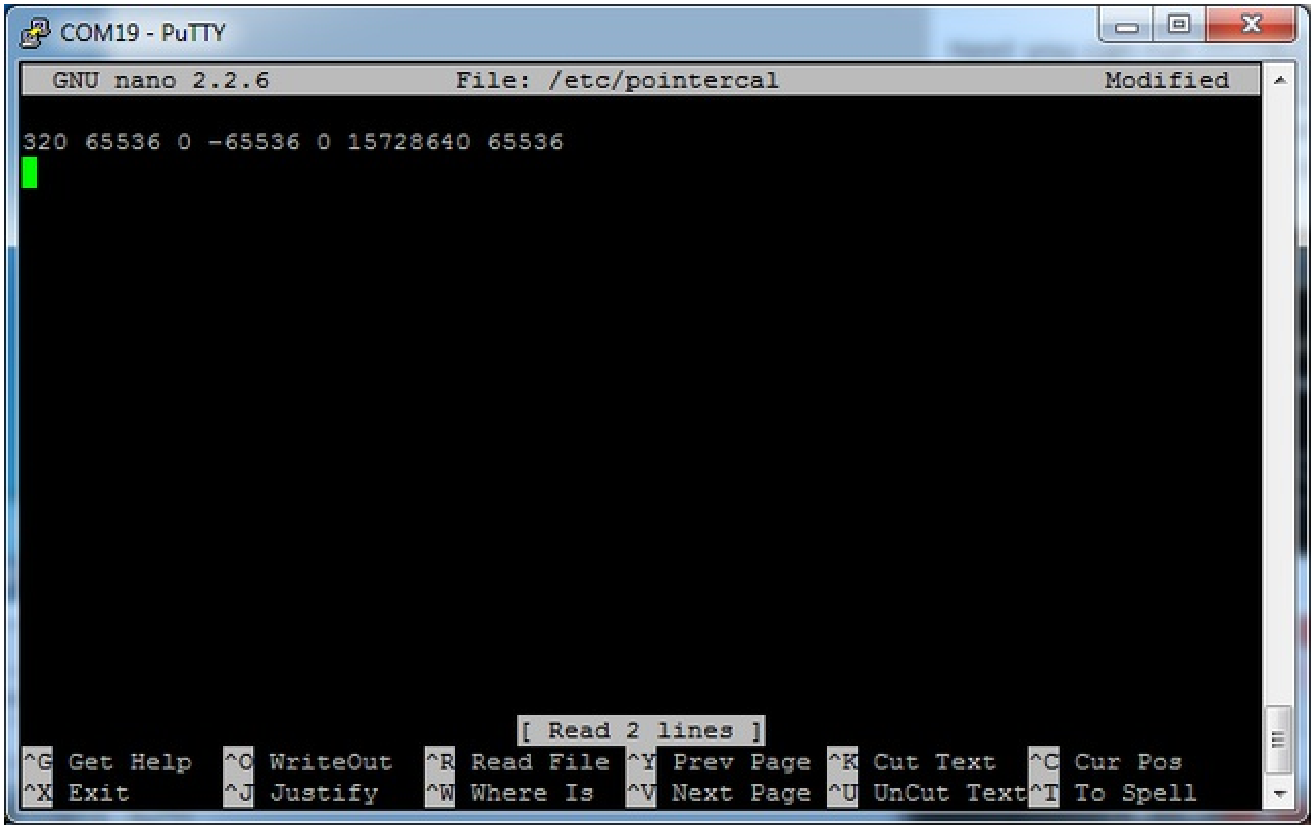
In order to use TSLIB - basically, the touchscreen without X11 - you'll need to set the calibration for TSLIB in /etc/pointercal

With a resistive touchscreen, you have to calibrate it. Since capacitive touchscreens don't require calibration you can just input the numbers directly. Run

```
sudo nano /etc/pointercal
```

And enter in the following values (there's a single space between each number) and hit return afterwards. Then save

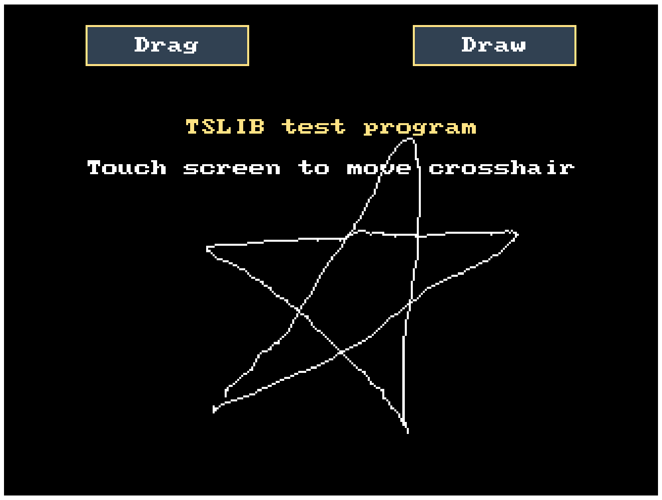
320 65536 0 -65536 0 15728640 65536



```
  GNU nano 2.2.6          File: /etc/pointercal                    Modified

320 65536 0 -65536 0 15728640 65536


                                [ Read 2 lines ]
^G Get Help  ^O WriteOut  ^R Read File ^Y Prev Page ^K Cut Text  ^C Cur Pos
^X Exit      ^J Justify   ^W Where Is  ^V Next Page ^U UnCut Text^T To Spell
```

Next you can run

```
sudo TSLIB_FBDEVICE=/dev/fb1 TSLIB_TSDEVICE=/dev/input/touchscreen ts_test
```



X11 Calibration

We can also manually enter in the touch calibration
edit the file with

```
sudo nano /etc/X11/xorg.conf.d/90-captouch.conf
```

and put the following in for rotation=90. For other rotations you may have to tweak SwapAxes and InvertY or InvertX

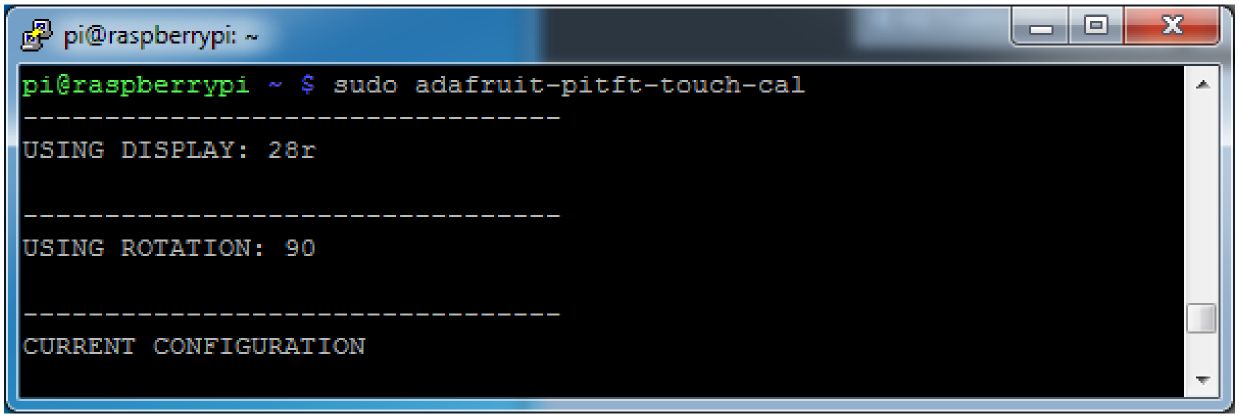
```
Section "InputClass"
        Identifier "captouch"
        MatchProduct "ft6x06_ts"
        Option "SwapAxes" "1"
        Option "InvertY" "1"
        Option "Calibration" "0 320 0 240"
EndSection
```

Calibration for other rotations

If you rotate the display you need to recalibrate the touchscreen to work with the new screen orientation. Since there's no calibration, you kinda just have to know the values for each rotation. To make it easy, you can run a small Python script which will automatically set a default touchscreen calibration based on the screen orientation.

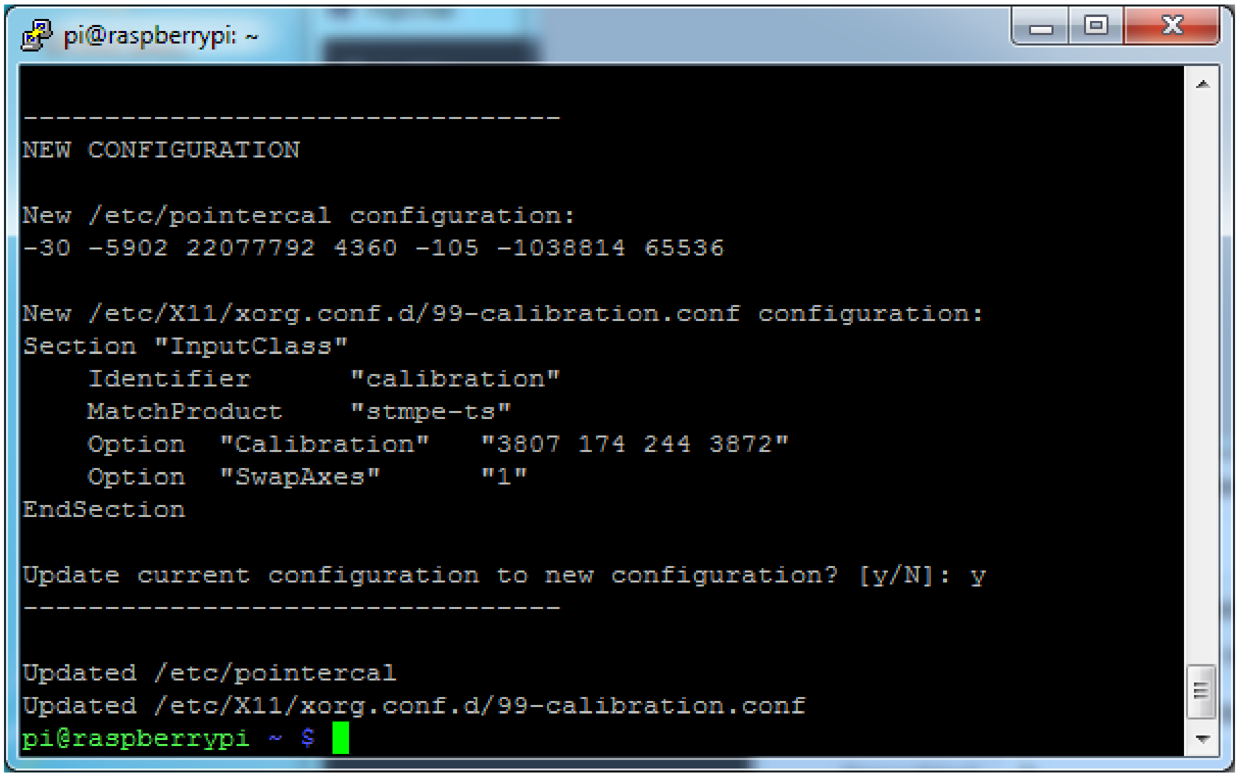
This helper is automatically installed for you but if you'd like you can check it out here on github (http://adafru.it/elu)

Run it at the command line with**sudo adafruit-pitft-touch-cal**
it will try to figure out what display you have installed and the rotation it's set up for



```
pi@raspberrypi ~ $ sudo adafruit-pitft-touch-cal
---------------------------------
USING DISPLAY: 28r

---------------------------------
USING ROTATION: 90

---------------------------------
CURRENT CONFIGURATION
```

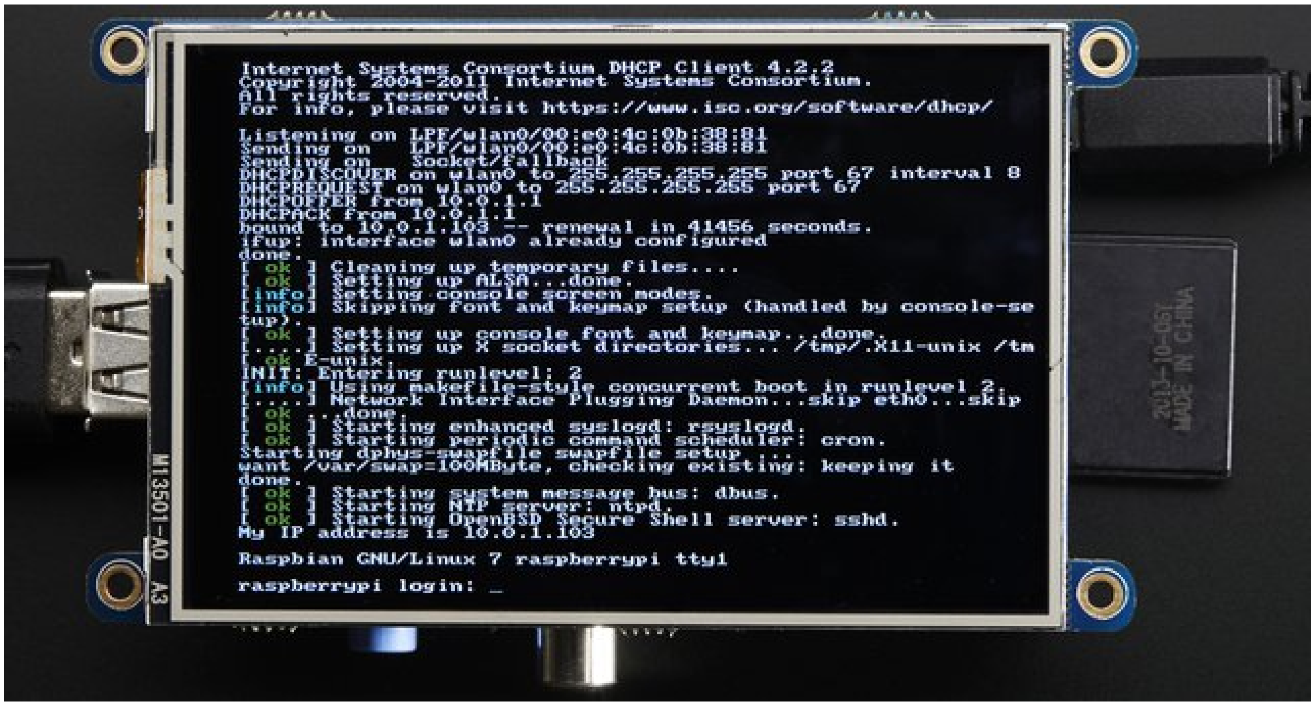
By default the script will attempt to read the screen orientation by examining the PiTFT module configuration with modprobe. If the script can read the orientation it will print out the current orientation, the current touchscreen calibration values, and the new touchscreen calibration values baesd on the current orientation. Before updating the calibration the script will ask you to confirm that you'd like to make the change. Press **y** and enter to confirm.



```

---------------------------------
NEW CONFIGURATION

New /etc/pointercal configuration:
-30 -5902 22077792 4360 -105 -1038814 65536

New /etc/X11/xorg.conf.d/99-calibration.conf configuration:
Section "InputClass"
    Identifier      "calibration"
    MatchProduct    "stmpe-ts"
    Option  "Calibration"   "3807 174 244 3872"
    Option  "SwapAxes"      "1"
EndSection

Update current configuration to new configuration? [y/N]: y
---------------------------------

Updated /etc/pointercal
Updated /etc/X11/xorg.conf.d/99-calibration.conf
pi@raspberrypi ~ $
```

Console Configuration

If you've grabbed our Easy Install image, or use the script, this step is not required, it's already done! This is just for advanced users who are curious on how to configure and customize the console



```
Internet Systems Consortium DHCP Client 4.2.2
Copyright 2004-2011 Internet Systems Consortium.
All rights reserved.
For info, please visit https://www.isc.org/software/dhcp/

Listening on LPF/wlan0/00:e0:4c:0b:38:81
Sending on   LPF/wlan0/00:e0:4c:0b:38:81
Sending on   Socket/fallback
DHCPDISCOVER on wlan0 to 255.255.255.255 port 67 interval 8
DHCPREQUEST on wlan0 to 255.255.255.255 port 67
DHCPOFFER from 10.0.1.1
DHCPACK from 10.0.1.1
bound to 10.0.1.103 -- renewal in 41456 seconds.
ifup: interface wlan0 already configured
done.
[ ok ] Cleaning up temporary files....
[ ok ] Setting up ALSA...done.
[info] Setting console screen modes.
[info] Skipping font and keymap setup (handled by console-se
tup).
[ ok ] Setting up console font and keymap...done.
[....] Setting up X socket directories... /tmp/.X11-unix /tm
[ ok E-unix.
INIT: Entering runlevel: 2
[info] Using makefile-style concurrent boot in runlevel 2.
[....] Network Interface Plugging Daemon...skip eth0...skip
[ ok ] ..done.
[ ok ] Starting enhanced syslogd: rsyslogd.
[ ok ] Starting periodic command scheduler: cron.
Starting dphys-swapfile swapfile setup ...
want /var/swap=100MByte, checking existing: keeping it
done.
[ ok ] Starting system message bus: dbus.
[ ok ] Starting NTP server: ntpd.
[ ok ] Starting OpenBSD Secure Shell server: sshd.
My IP address is 10.0.1.103

Raspbian GNU/Linux 7 raspberrypi tty1

raspberrypi login: _
```

One fun thing you can do with the display is have it as your main console instead of the HDMI/TV output. Even though it is small, with a good font you can get 20 x 40 of text. For more details, check out https://github.com/notro/fbtft/wiki/Boot-console (http://adafru.it/cXQ)

First up, we'll update the boot configuration file to use the TFT framebuffer/**dev/fb1** instead of the HDMI/TV framebuffer /dev/fb0

```
sudo nano /boot/cmdline.txt
```

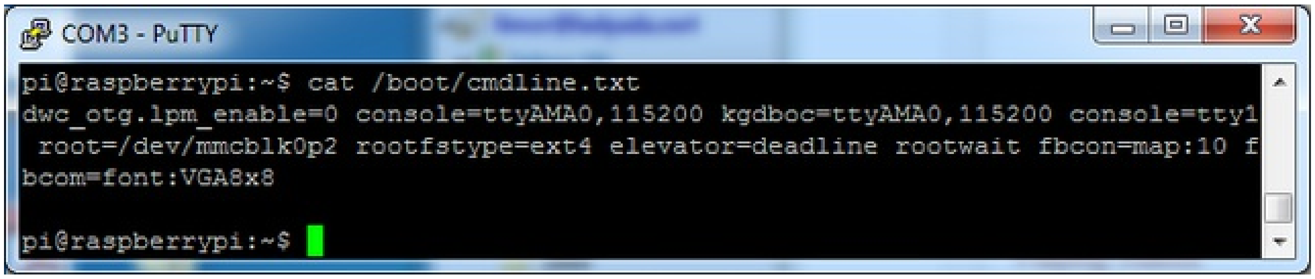
you can also edit it by putting the SD card into a computer and opening the same file.

At the end of the line, find the text that says `rootwait` and right after that, enter in: `fbcon=map:10 fbcon=font:VGA8x8` then save the file.

On the next boot, it will bring up the console.

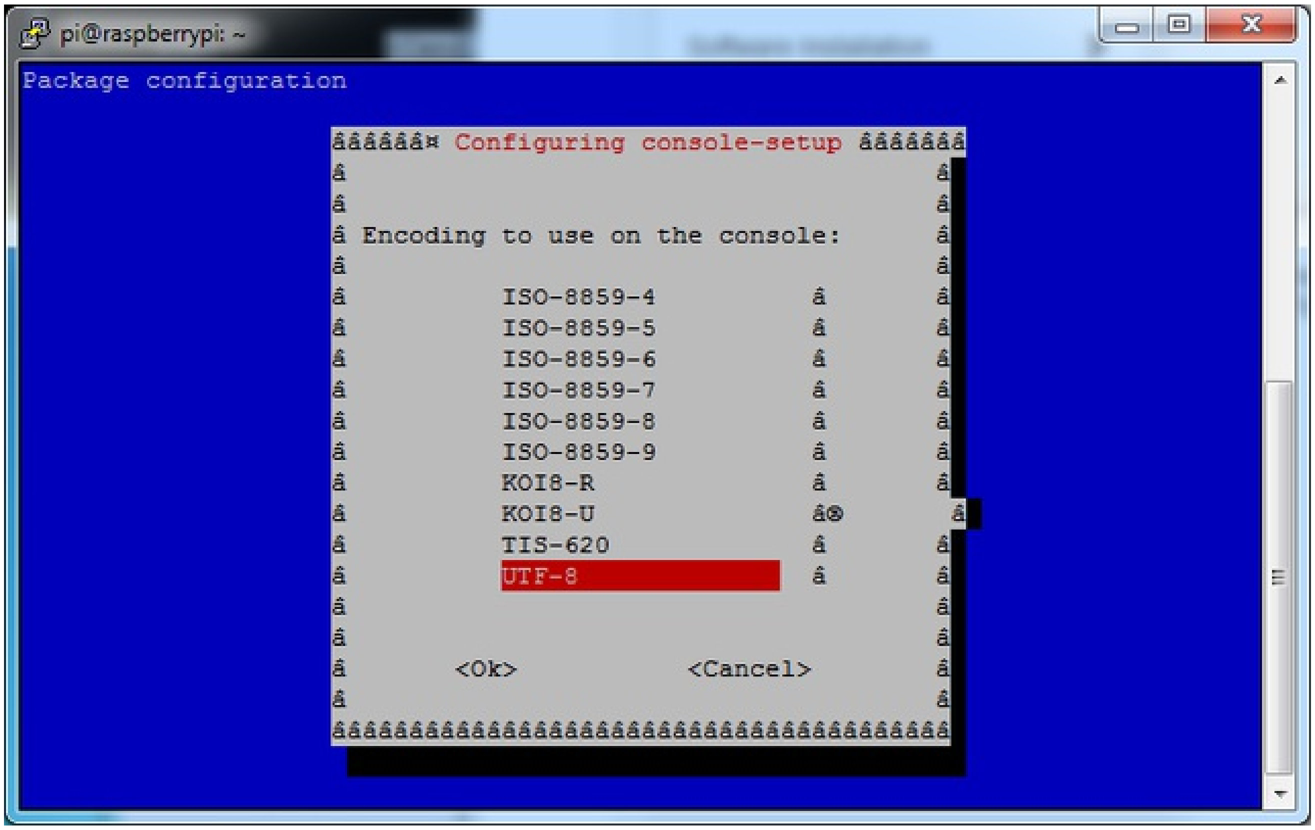
Note that the kernel has to load up the display driver module before it can display

anything on it so you won't get the rainbow screen, a NooBs prompt, or a big chunk of the kernel details since the module is loaded fairly late in the boot process.

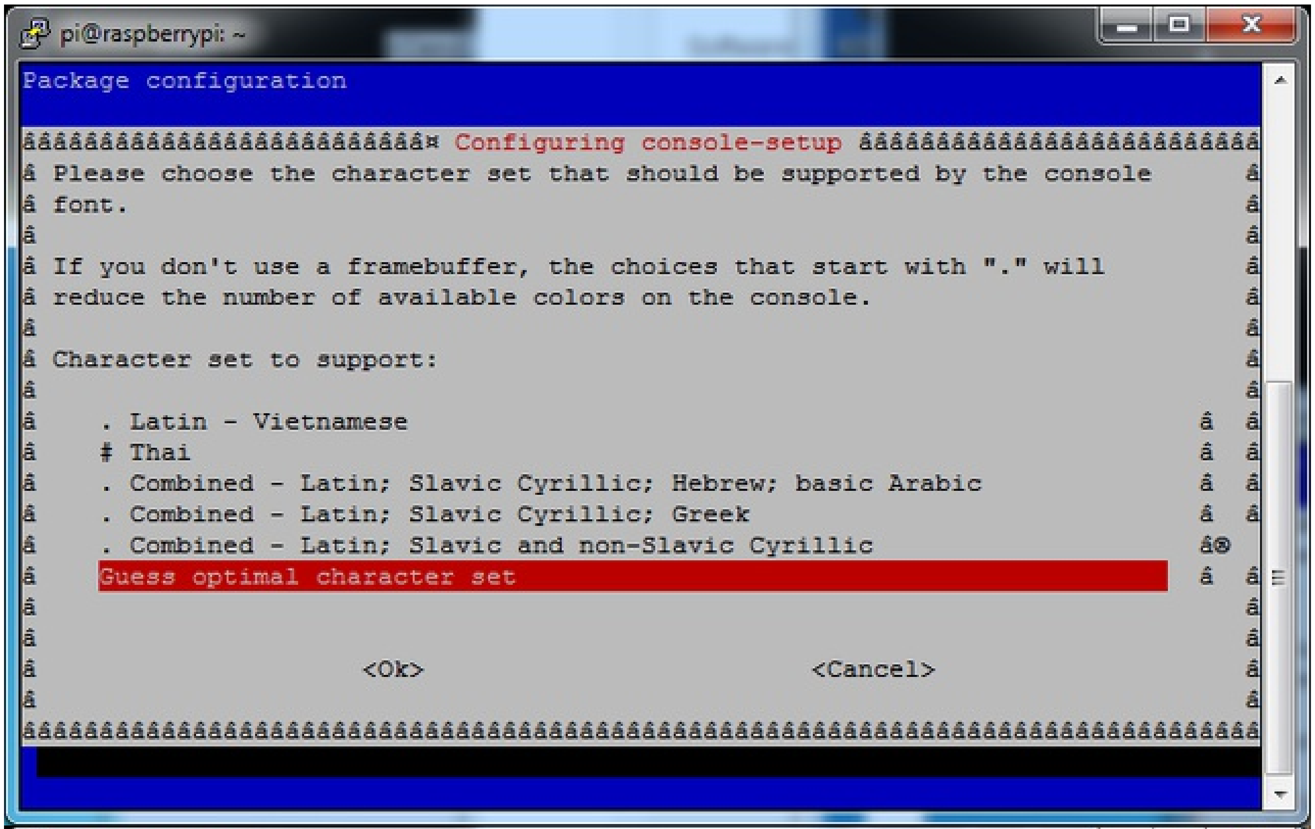


```
pi@raspberrypi:~$ cat /boot/cmdline.txt
dwc_otg.lpm_enable=0 console=ttyAMA0,115200 kgdboc=ttyAMA0,115200 console=tty1
 root=/dev/mmcblk0p2 rootfstype=ext4 elevator=deadline rootwait fbcon=map:10 f
bcom=font:VGA8x8

pi@raspberrypi:~$
```

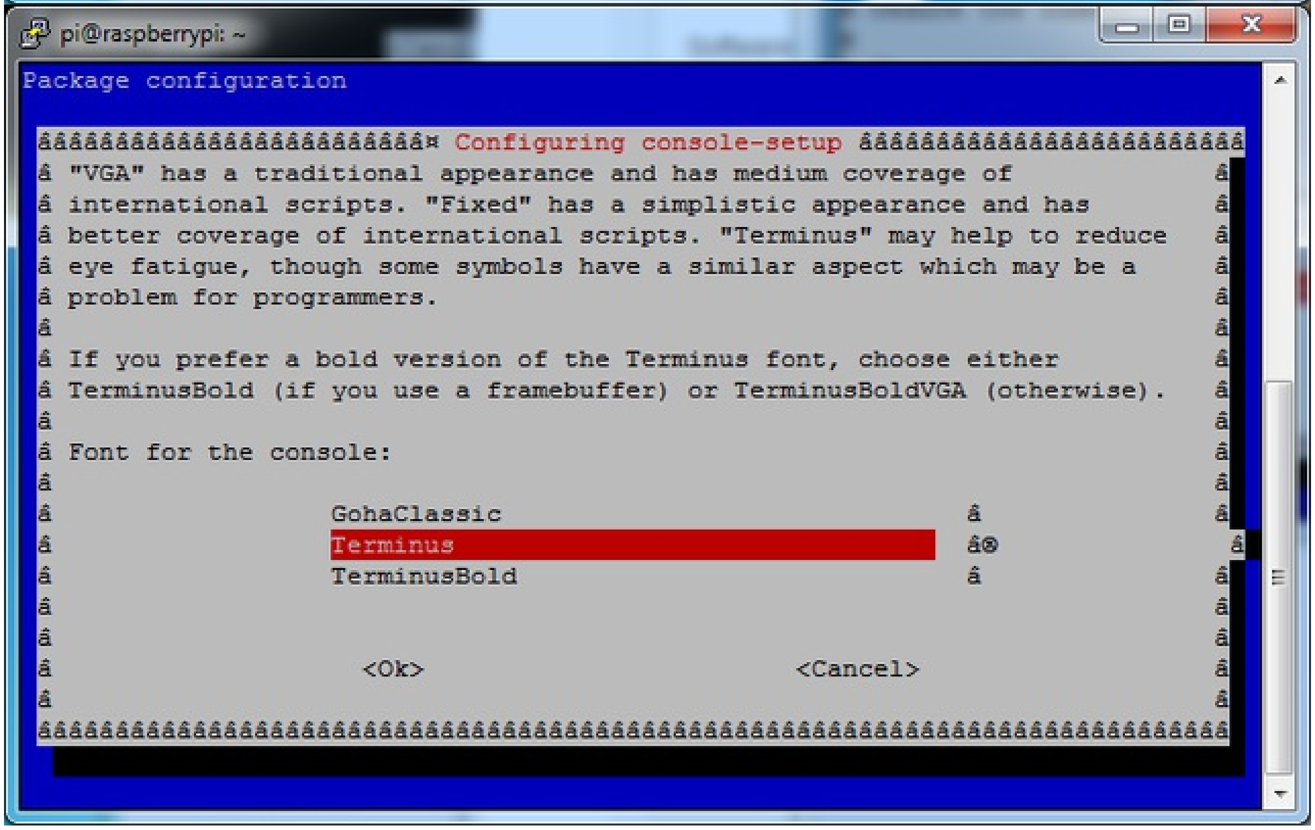
I think the VGA8x8 font is a bit chunky, you probably want 12x6 which is what is shown in the photo above. To change the font, run **sudo dpkg-reconfigure console-setup** and go thru to select Terminus 6x12



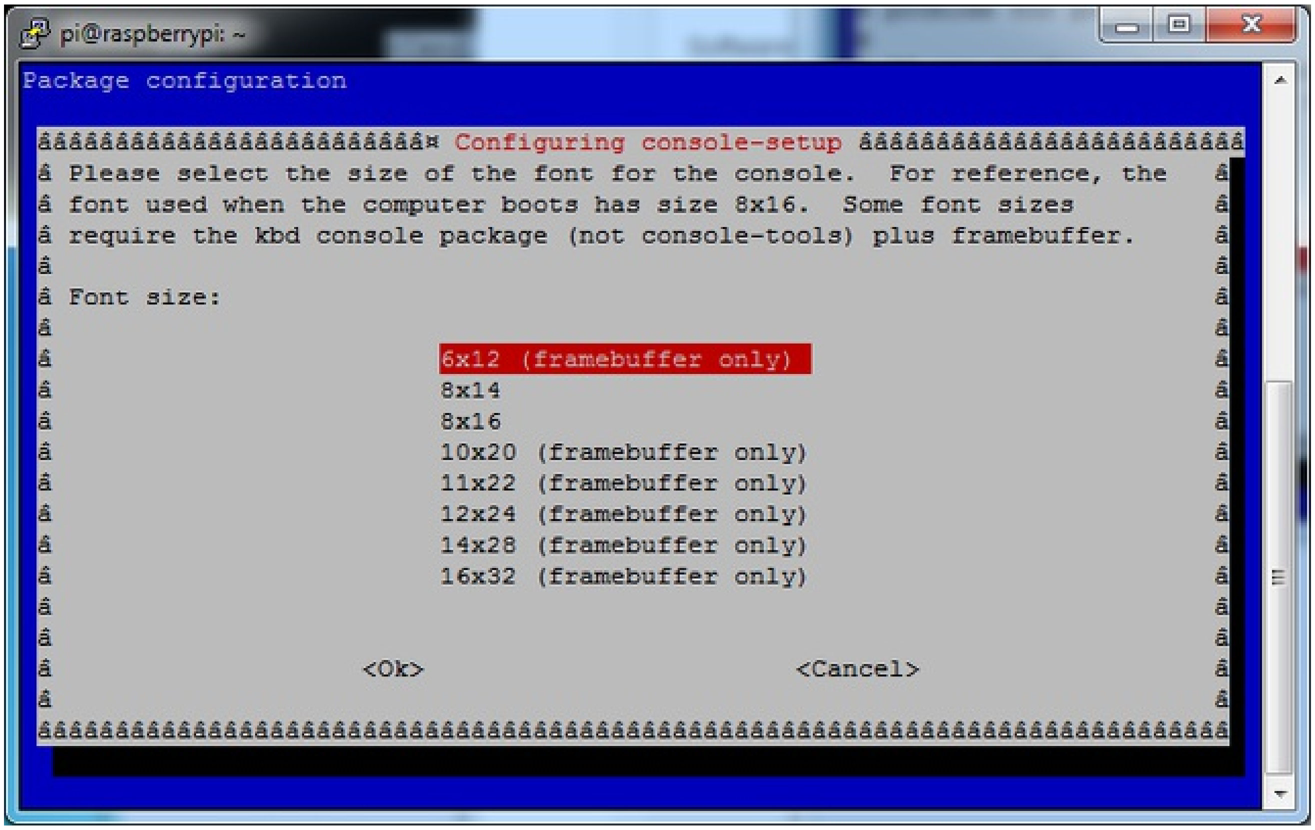
```
Package configuration

âââââââ¤ Configuring console-setup âââââââ
â                                       â
â                                       â
â Encoding to use on the console:       â
â                                       â
â          ISO-8859-4          â        â
â          ISO-8859-5          â        â
â          ISO-8859-6          â        â
â          ISO-8859-7          â        â
â          ISO-8859-8          â        â
â          ISO-8859-9          â        â
â          KOI8-R              â        â
â          KOI8-U              â®        â
â          TIS-620             â        â
â          UTF-8               â        â
â                                       â
â                                       â
â       <Ok>           <Cancel>         â
â                                       â
âââââââââââââââââââââââââââââââââââââââââ
```



```
pi@raspberrypi: ~
Package configuration

ââââââââââââââââââââââââââ¤ Configuring console-setup âââââââââââââââââââââââââ
â Please choose the character set that should be supported by the console     â
â font.                                                                        â
â                                                                              â
â If you don't use a framebuffer, the choices that start with "." will         â
â reduce the number of available colors on the console.                        â
â                                                                              â
â Character set to support:                                                    â
â                                                                              â
â     . Latin - Vietnamese                                                  â  â
â     # Thai                                                                â  â
â     . Combined - Latin; Slavic Cyrillic; Hebrew; basic Arabic             â  â
â     . Combined - Latin; Slavic Cyrillic; Greek                            â  â
â     . Combined - Latin; Slavic and non-Slavic Cyrillic                    â®
â     Guess optimal character set                                           â  â
â                                                                              â
â                                                                              â
â                      <Ok>                         <Cancel>                   â
â                                                                              â
ââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââ
```



```
pi@raspberrypi: ~
Package configuration

 ââââââââââââââââââââââââââ¤ Configuring console-setup ââââââââââââââââââââââââââ
 â "VGA" has a traditional appearance and has medium coverage of             â
 â international scripts. "Fixed" has a simplistic appearance and has       â
 â better coverage of international scripts. "Terminus" may help to reduce   â
 â eye fatigue, though some symbols have a similar aspect which may be a     â
 â problem for programmers.                                                  â
 â                                                                           â
 â If you prefer a bold version of the Terminus font, choose either          â
 â TerminusBold (if you use a framebuffer) or TerminusBoldVGA (otherwise).   â
 â                                                                           â
 â Font for the console:                                                     â
 â                                                                           â
 â                  GohaClassic                       â                      â
 â                  Terminus                          â®                       â
 â                  TerminusBold                      â                      â
 â                                                                           â
 â                                                                           â
 â                    <Ok>                          <Cancel>                 â
 â                                                                           â
 ââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââ
```



```
Package configuration

ââââââââââââââââââââââââââ¤ Configuring console-setup âââââââââââââââââââââââââ
â Please select the size of the font for the console.  For reference, the   â
â font used when the computer boots has size 8x16.  Some font sizes          â
â require the kbd console package (not console-tools) plus framebuffer.      â
â                                                                            â
â Font size:                                                                 â
â                                                                            â
â                          6x12 (framebuffer only)                           â
â                          8x14                                              â
â                          8x16                                              â
â                          10x20 (framebuffer only)                          â
â                          11x22 (framebuffer only)                          â
â                          12x24 (framebuffer only)                          â
â                          14x28 (framebuffer only)                          â
â                          16x32 (framebuffer only)                          â
â                                                                            â
â                                                                            â
â                     <Ok>                         <Cancel>                  â
â                                                                            â
ââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââââ
```

Turn off Console Blanking

You may notice the console goes black after 30 minutes, this is a sort of 'power saving' or 'screensaver' feature.

Raspbian Jessie

Add the following line to /etc/rc.local

```
sudo sh -c "TERM=linux setterm -blank 0 >/dev/tty0"
```

on the line before the final `exit 0`

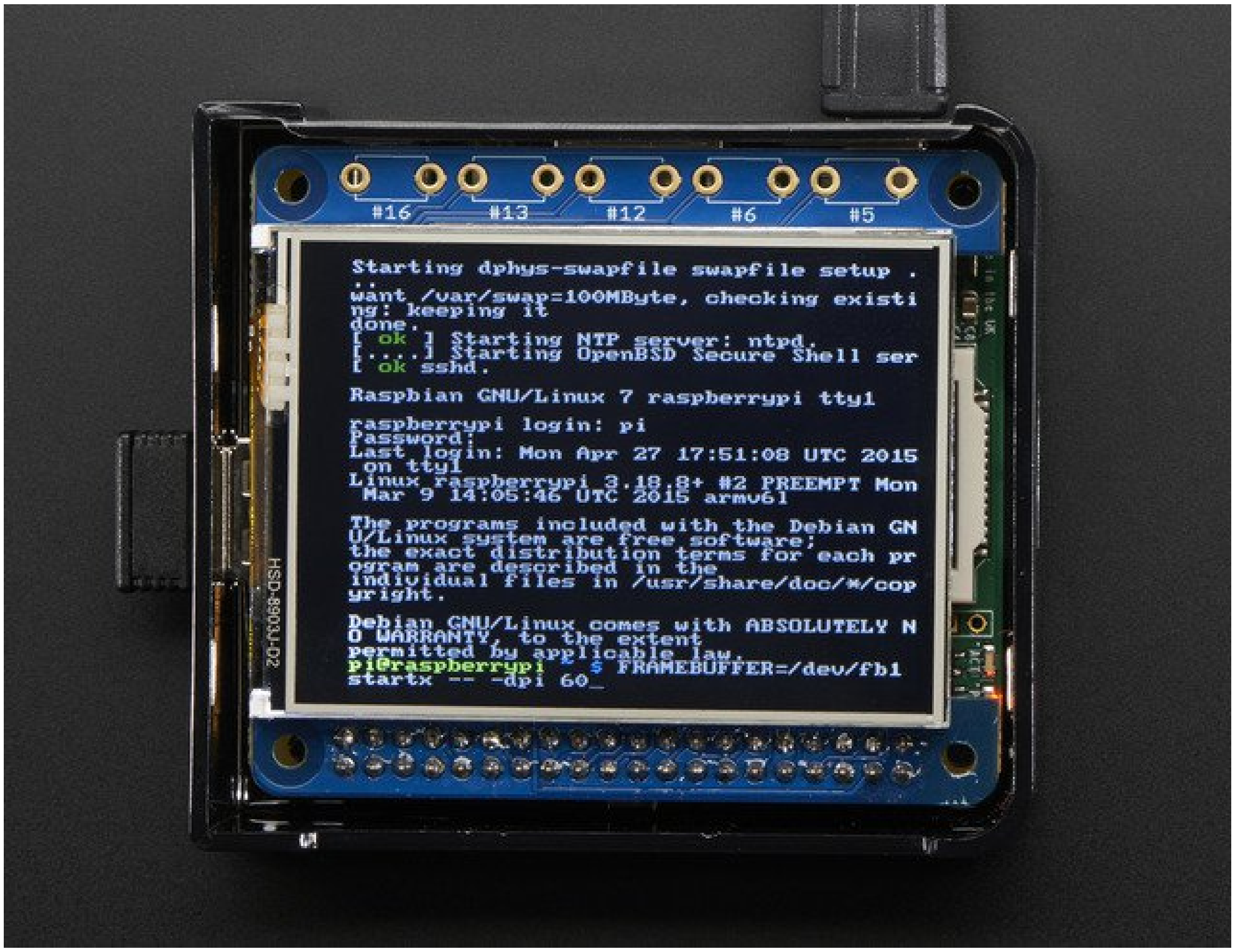
Raspbian Wheezy

You can disable this by editing **/etc/kbd/config** and looking for

```
BLANK_TIME=30
```

and setting the blank time to 0 (which turns it off)

```
BLANK_TIME=0
```



```
Starting dphys-swapfile swapfile setup .
..
want /var/swap=100MByte, checking existi
ng: keeping it
done.
[ ok ] Starting NTP server: ntpd.
[....] Starting OpenBSD Secure Shell ser
[ ok sshd.

Raspbian GNU/Linux 7 raspberrypi tty1

raspberrypi login: pi
Password:
Last login: Mon Apr 27 17:51:08 UTC 2015
 on tty1
Linux raspberrypi 3.18.8+ #2 PREEMPT Mon
 Mar 9 14:05:46 UTC 2015 armv6l

The programs included with the Debian GN
U/Linux system are free software;
the exact distribution terms for each pr
ogram are described in the
individual files in /usr/share/doc/*/cop
yright.

Debian GNU/Linux comes with ABSOLUTELY N
O WARRANTY, to the extent
permitted by applicable law.
pi@raspberrypi ~ $ FRAMEBUFFER=/dev/fb1
startx -- -dpi 60_
```

Playing Videos

- Try a different -vo driver (-vo help for a list)
or try -framedrop!
A: 17.9 V: 17.3 A-V: 0.658 ct: 1.694 0/ 0 65%
2.8" 320x240 TFT w/Touch Screen for Raspberry Pi!
adafruit
Connector on reverse!
Connector on reverse!

How To Play Videos

You can play many types of videos on the screen, using mplayer you don't even need to run X and you can script the movies to play using Python. We'll show you how to just play one video for now.
To demo, we'll use an mp4 of Big Buck Bunny for 320 pixel wide screens. Below we show you how to create/resize videos, but to make it easy, just download our version with:

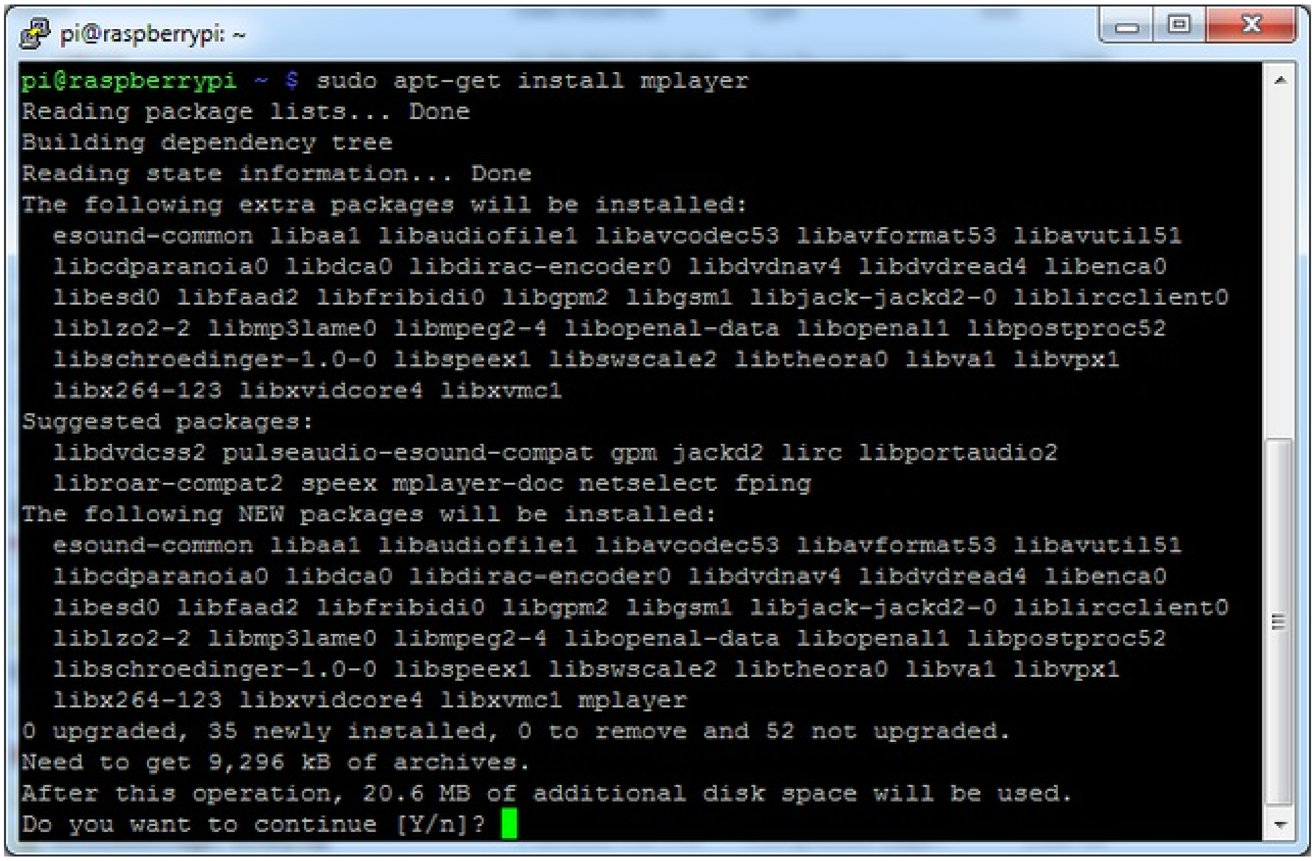
wget http://adafruit-download.s3.amazonaws.com/bigbuckbunny320p.mp4 (http://adafru.it/cXR)

The video is 30MB which is a lot if you haven't expanded your SD card yet. Before you do this, run sudo raspi-config to expand the SD card so you don't run out of space!

If you don't have **mplayer** yet, run

```
sudo apt-get update

sudo apt-get install mplayer
```

to install it. It may take a few minutes to complete

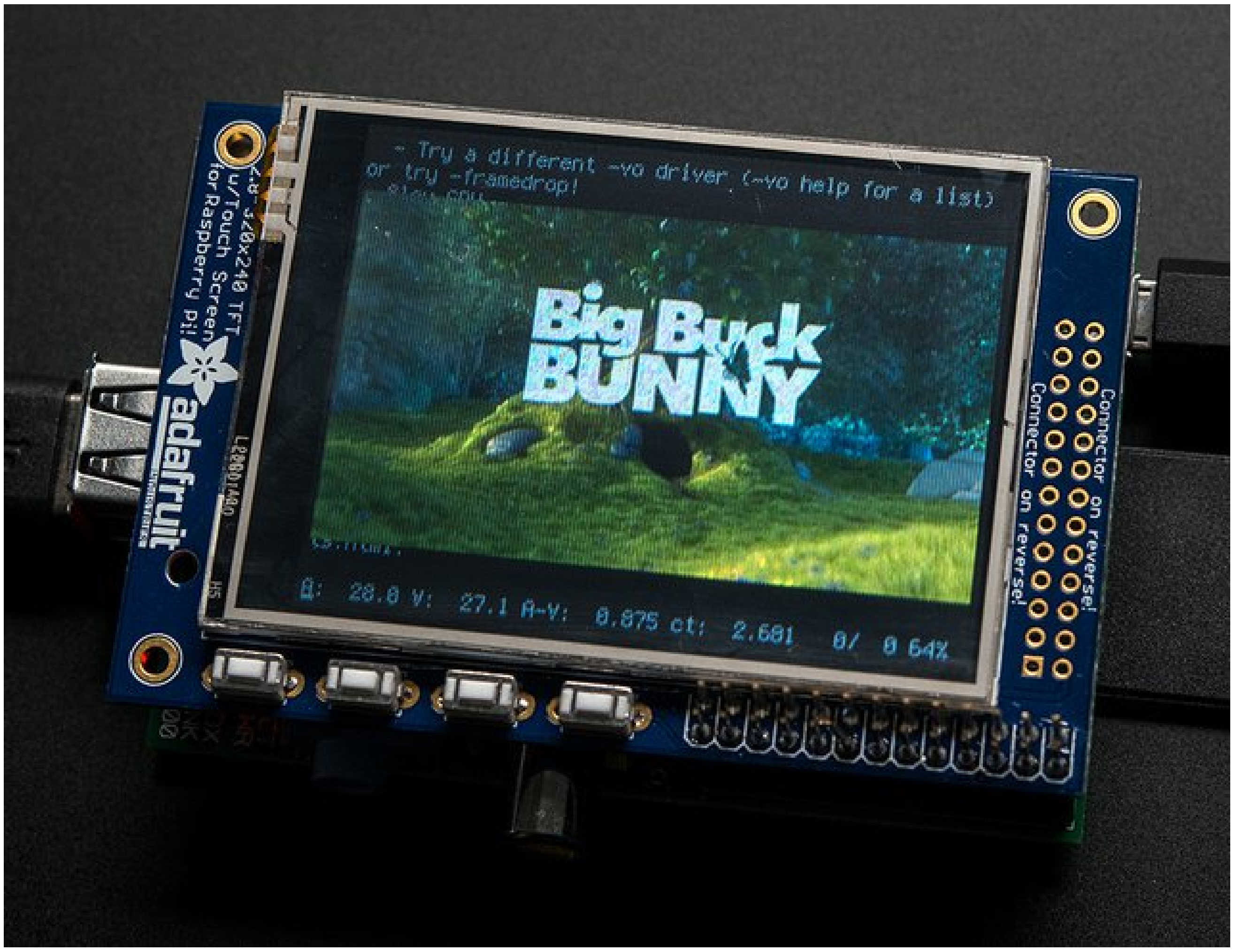


```
pi@raspberrypi ~ $ sudo apt-get install mplayer
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  esound-common libaa1 libaudiofile1 libavcodec53 libavformat53 libavutil51
  libcdparanoia0 libdca0 libdirac-encoder0 libdvdnav4 libdvdread4 libenca0
  libesd0 libfaad2 libfribidi0 libgpm2 libgsm1 libjack-jackd2-0 liblircclient0
  liblzo2-2 libmp3lame0 libmpeg2-4 libopenal-data libopenal1 libpostproc52
  libschroedinger-1.0-0 libspeex1 libswscale2 libtheora0 libva1 libvpx1
  libx264-123 libxvidcore4 libxvmc1
Suggested packages:
  libdvdcss2 pulseaudio-esound-compat gpm jackd2 lirc libportaudio2
  libroar-compat2 speex mplayer-doc netselect fping
The following NEW packages will be installed:
  esound-common libaa1 libaudiofile1 libavcodec53 libavformat53 libavutil51
  libcdparanoia0 libdca0 libdirac-encoder0 libdvdnav4 libdvdread4 libenca0
  libesd0 libfaad2 libfribidi0 libgpm2 libgsm1 libjack-jackd2-0 liblircclient0
  liblzo2-2 libmp3lame0 libmpeg2-4 libopenal-data libopenal1 libpostproc52
  libschroedinger-1.0-0 libspeex1 libswscale2 libtheora0 libva1 libvpx1
  libx264-123 libxvidcore4 libxvmc1 mplayer
0 upgraded, 35 newly installed, 0 to remove and 52 not upgraded.
Need to get 9,296 kB of archives.
After this operation, 20.6 MB of additional disk space will be used.
Do you want to continue [Y/n]?
```

OK now you just have to run:

```
sudo SDL_VIDEODRIVER=fbcon SDL_FBDEV=/dev/fb1 mplayer -vo sdl -framedrop bigbuckbunny320p.mp4
```

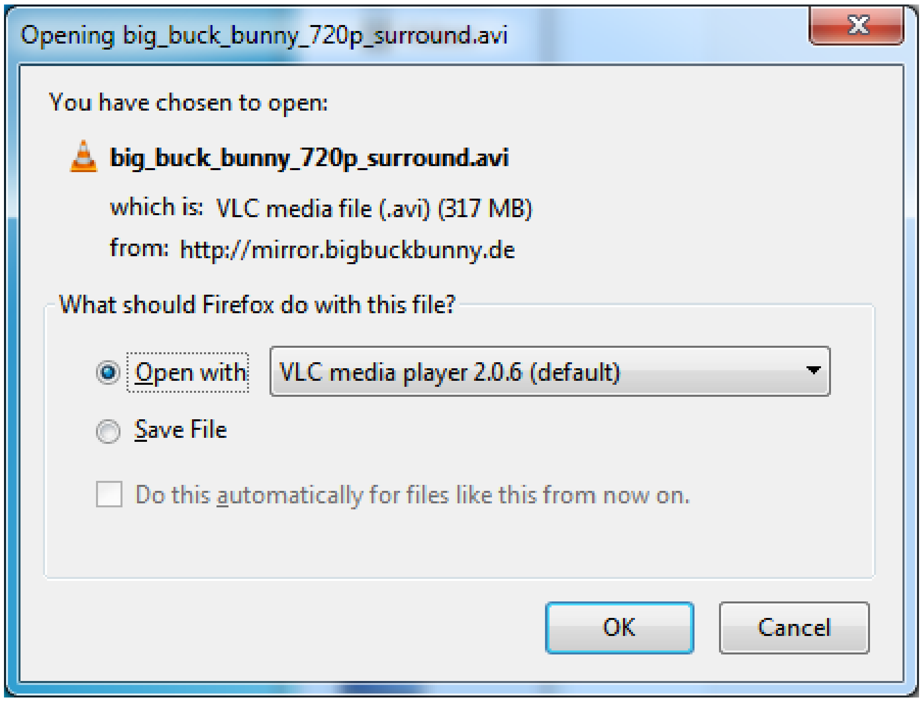
If your video is not sized for 320 wide, you may need to add a `-zoom` after `-framedrop` so that it will resize - note that this is quite taxing for the Pi, so it may result in a choppy or mis-synced video!



Converting/Resizing Videos

It's possible to play full length videos on the TFT plate, but since the screen is small and the Pi cant use hardware accelleration to play the videos its best to scale them down to 320x240 pixels. This will be easier for the Pi to play and also save you tons of storage space. For this demo, we'll be using the famous Big Buck Bunny (http://adafru.it/cXS) video, which is creative commons and also very funny!

You can download it from the link above, we'll be using the 720p AVI version.


Opening big_buck_bunny_720p_surround.avi

You have chosen to open:

big_buck_bunny_720p_surround.avi

which is: VLC media file (.avi) (317 MB)

from: http://mirror.bigbuckbunny.de

What should Firefox do with this file?

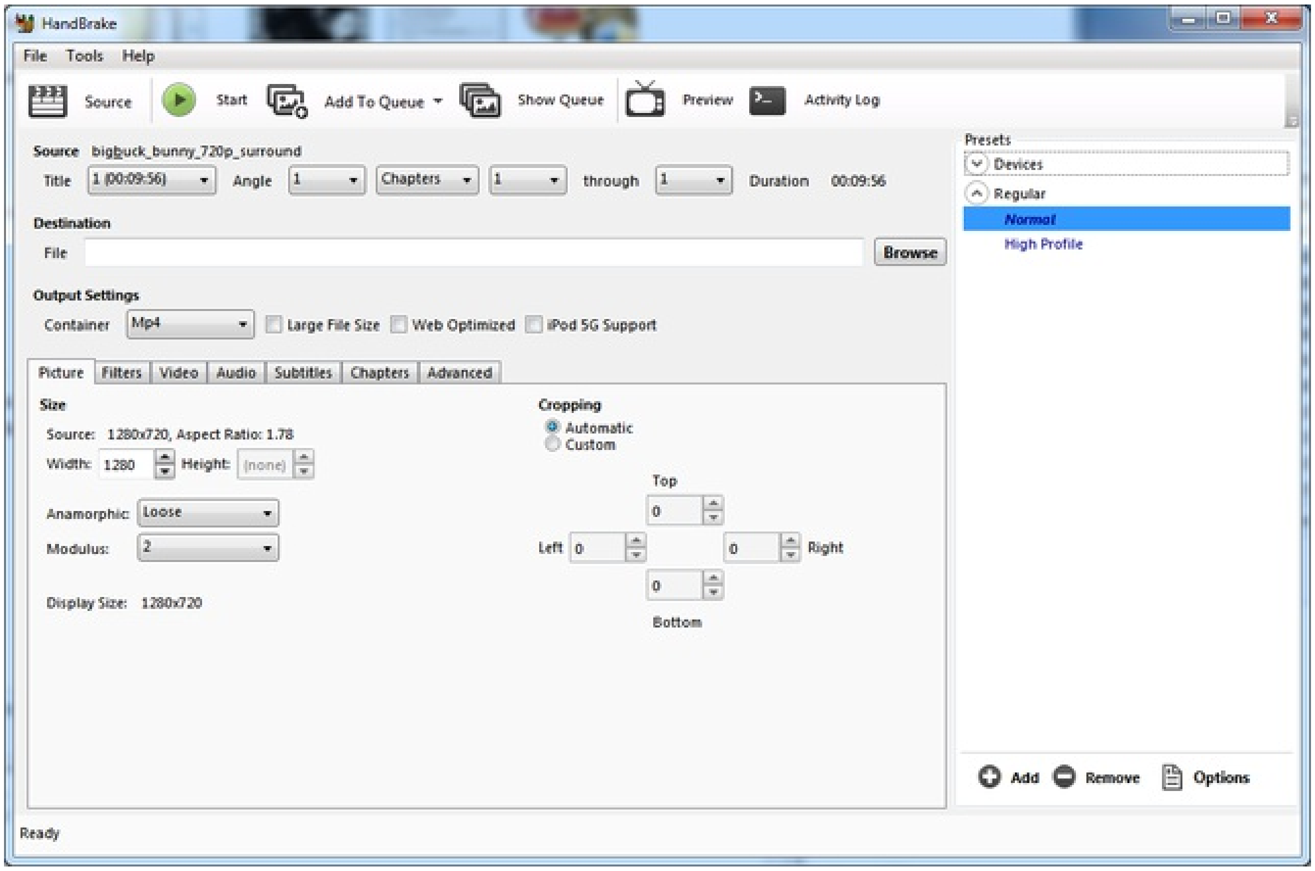
[✓ Open with] VLC media player 2.0.6 (default)

Save File

Do this automatically for files like this from now on.

OK Cancel

To do the conversion itself, we suggest HandBrake (http://adafru.it/cXT) which works great and is open source so it runs on all operating systems! Download and install from the link. Then run the installed application and open up the AVI file from before. The app will pre-fill a bunch of information about it.



Source bigbuck_bunny_720p_surround

Title 1 (00:09:56) Angle 1 Chapters 1 through 1 Duration 00:09:56

Destination

File Browse

Output Settings

Container Mp4 Large File Size Web Optimized iPod 5G Support

Picture Filters Video Audio Subtitles Chapters Advanced

Size

Source: 1280x720, Aspect Ratio: 1.78

Width: 1280 Height: (none)

Anamorphic: Loose

Modulus: 2

Display Size: 1280x720

Cropping

[✓ Automatic] Custom

Top 0

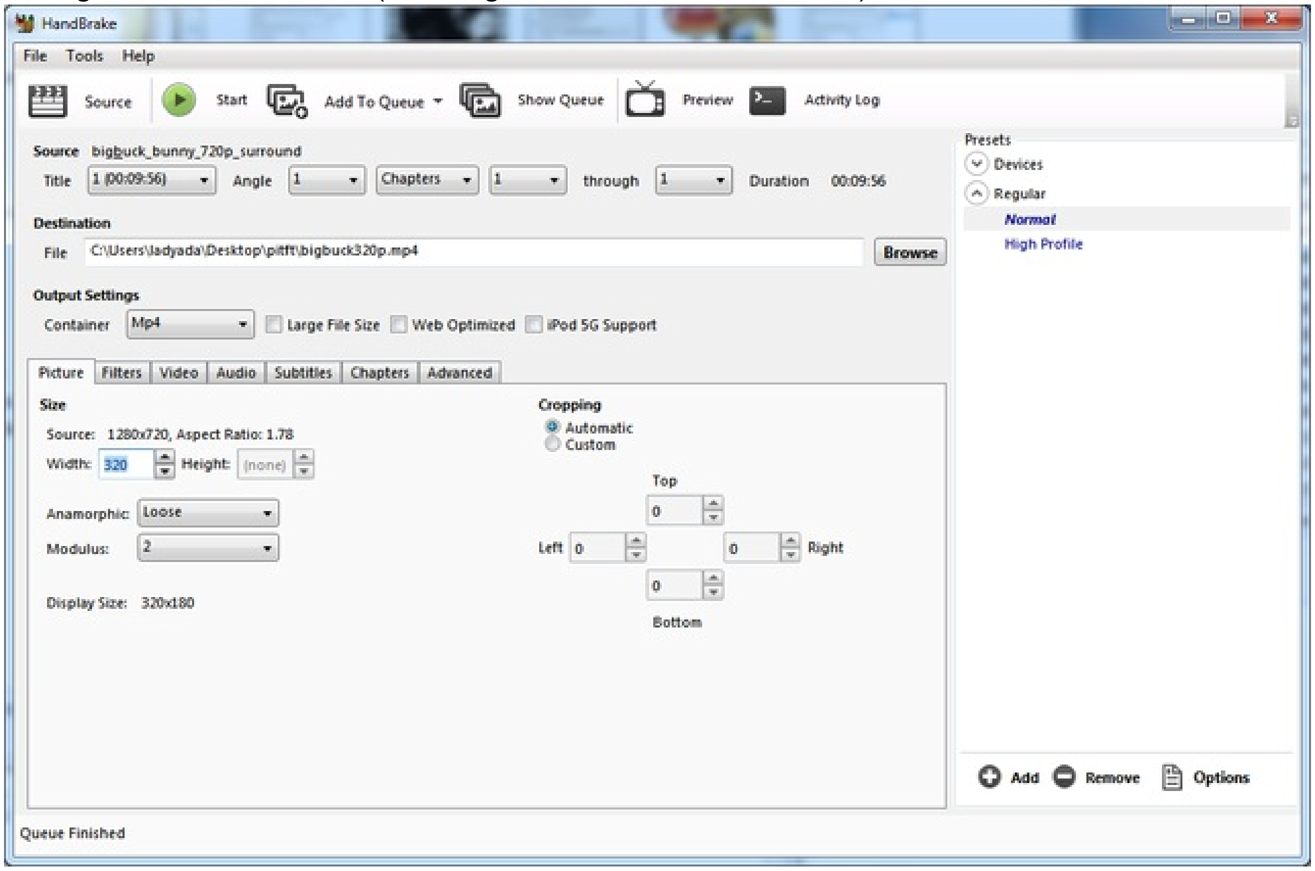
Left 0 0 Right

0

Bottom

Ready

Under **Destination** click **Browse...** to select a new MP4 file to save. Then under**Picture** change the **Width** to 320 (the height will be auto-calculated)



Source bigbuck_bunny_720p_surround
Title 1 (00:09:56) Angle 1 Chapters 1 through 1 Duration 00:09:56

Destination
File C:\Users\ladyada\Desktop\pitft\bigbuck320p.mp4

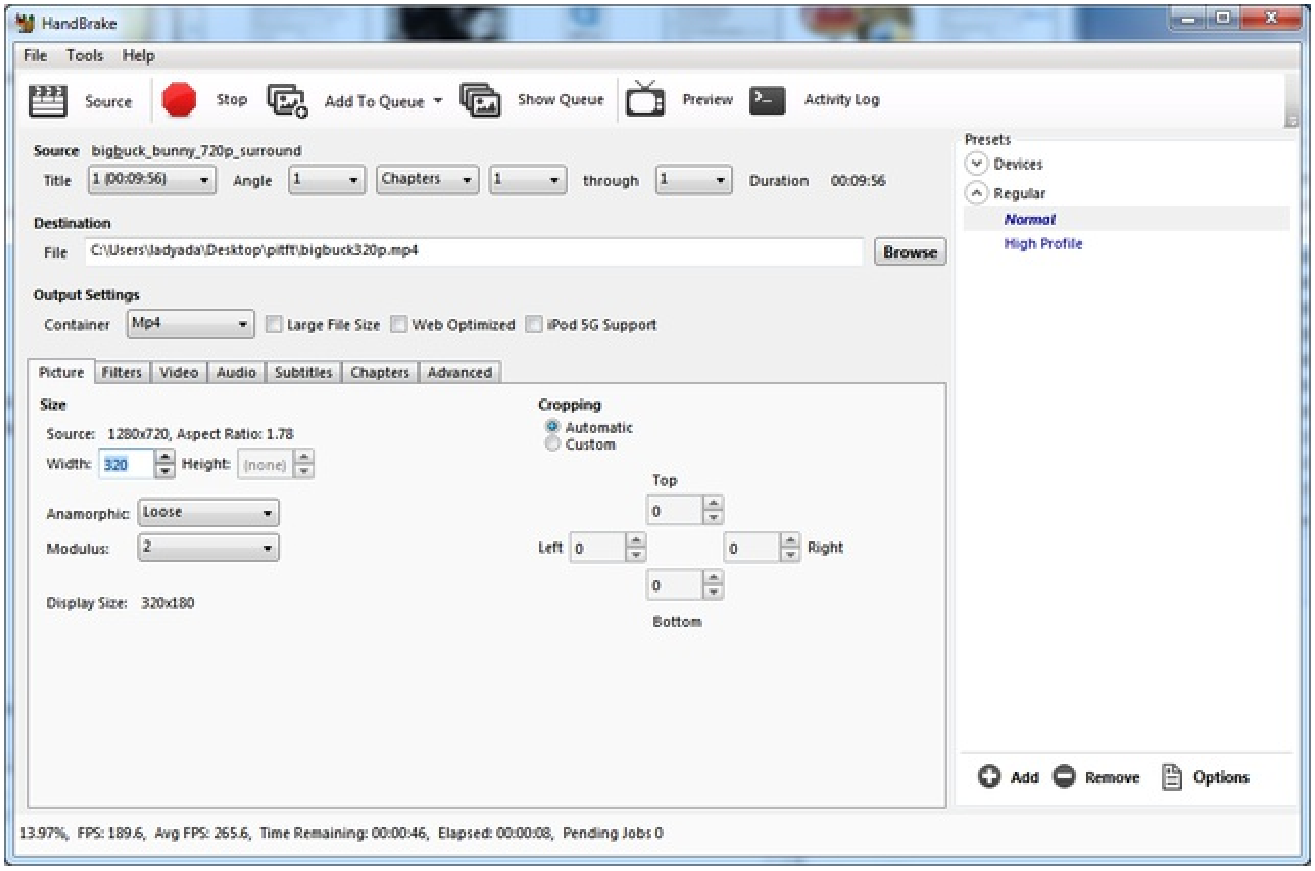
Output Settings
Container Mp4 Large File Size Web Optimized iPod 5G Support

Size
Source: 1280x720, Aspect Ratio: 1.78
Width: 320 Height: (none)
Anamorphic: Loose
Modulus: 2
Display Size: 320x180

Cropping
Automatic
Custom
Top 0
Left 0
Right 0
Bottom 0

Queue Finished

Click **START** to begin the conversion, it will take a minute or two.



Source bigbuck_bunny_720p_surround
Title 1 (00:09:56) Angle 1 Chapters 1 through 1 Duration 00:09:56

Destination
File C:\Users\ladyada\Desktop\pitft\bigbuck320p.mp4

Output Settings
Container Mp4 [ ] Large File Size [ ] Web Optimized [ ] iPod 5G Support

Size
Source: 1280x720, Aspect Ratio: 1.78
Width: 320 Height: (none)
Anamorphic: Loose
Modulus: 2
Display Size: 320x180

Cropping
[✓ Automatic]
Custom
Top 0
Left 0 Right 0
Bottom 0

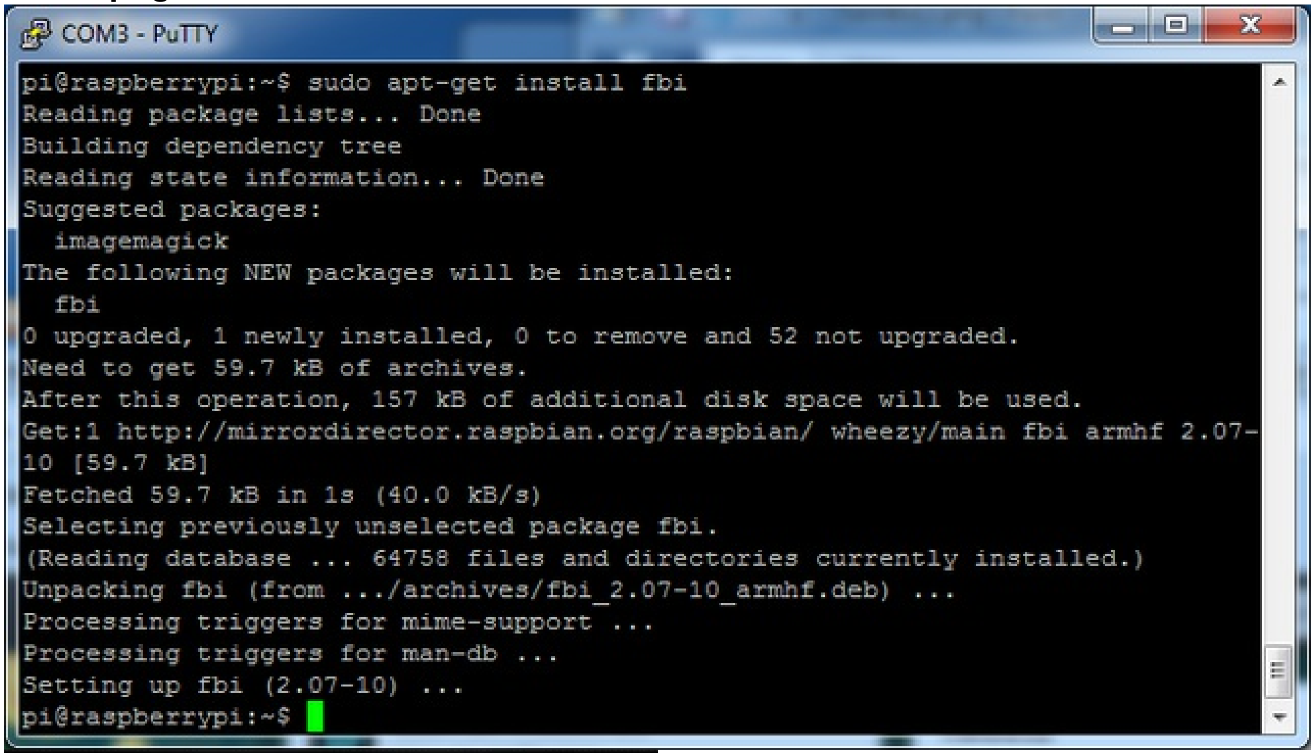
13.97%, FPS: 189.6, Avg FPS: 265.6, Time Remaining: 00:00:46, Elapsed: 00:00:08, Pending Jobs 0

That's it! You now have a smaller file. Don't forget to play it on your computer to make sure it plays right before copying it to your Pi

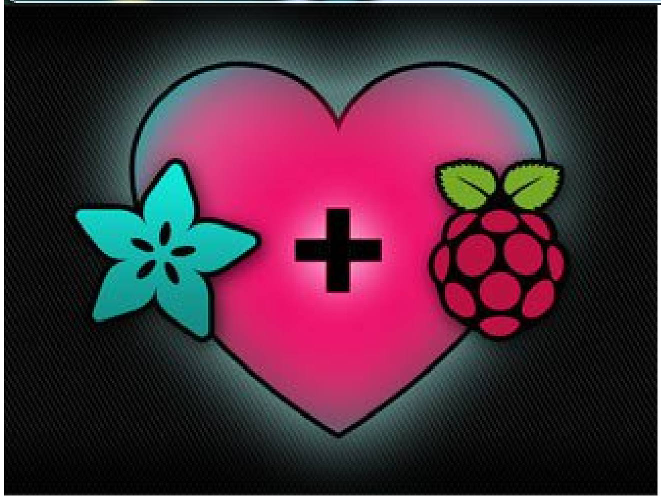
Displaying Images

You can display every day images such as GIFs, JPGs, BMPs, etc on the screen. To do this we'll install **fbi** which is the **frame buffer image** viewer (not to be confused with the FBI agency!)

sudo apt-get install fbi will install it



```
pi@raspberrypi:~$ sudo apt-get install fbi
Reading package lists... Done
Building dependency tree
Reading state information... Done
Suggested packages:
  imagemagick
The following NEW packages will be installed:
  fbi
0 upgraded, 1 newly installed, 0 to remove and 52 not upgraded.
Need to get 59.7 kB of archives.
After this operation, 157 kB of additional disk space will be used.
Get:1 http://mirrordirector.raspbian.org/raspbian/ wheezy/main fbi armhf 2.07-
10 [59.7 kB]
Fetched 59.7 kB in 1s (40.0 kB/s)
Selecting previously unselected package fbi.
(Reading database ... 64758 files and directories currently installed.)
Unpacking fbi (from .../archives/fbi_2.07-10_armhf.deb) ...
Processing triggers for mime-support ...
Processing triggers for man-db ...
Setting up fbi (2.07-10) ...
pi@raspberrypi:~$
```



Grab our lovely wallpapers with

```
wget http://adafruit-download.s3.amazonaws.com/adapiluv320x240.jpg
wget http://adafruit-download.s3.amazonaws.com/adapiluv480x320.png (http://adafru.it/cXU)
```

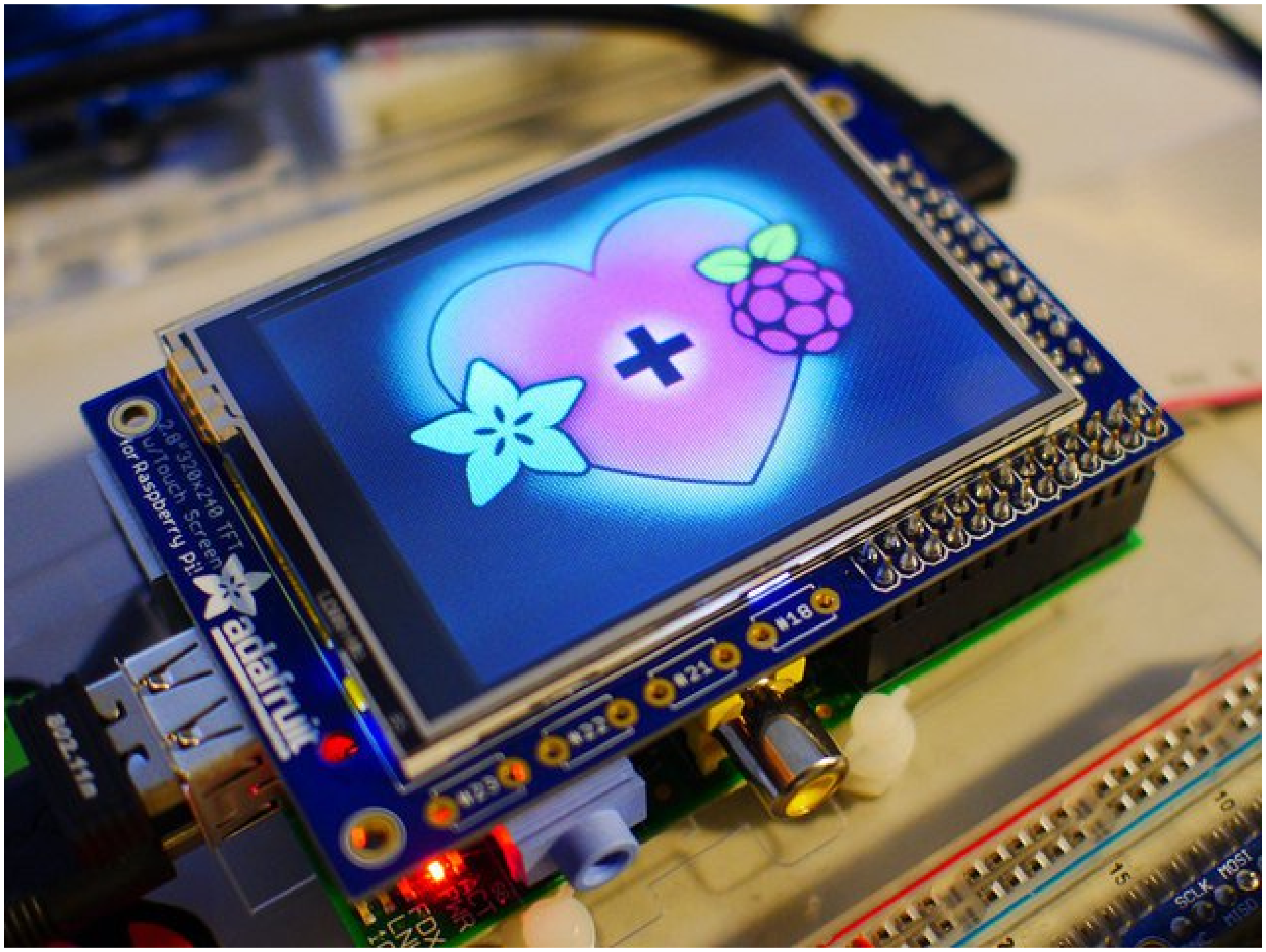
For 320x240 PiTFTs (2.2", 2.4", 2.8" or 3.2") view it with

```
sudo fbi -T 2 -d /dev/fb1 -noverbose -a adapiluv320x240.jpg
```

or for 3.5" PiTFTs:

```
sudo fbi -T 2 -d /dev/fb1 -noverbose -a adapiluv480x320 (http://adafru.it/cXU).jpg
```

That's it!



Using FBCP

The Ideal: Adafruit's PiTFT displays are razor sharp. Whereas small composite screens on the Raspberry Pi usually require some video scaling (resulting in blurriness), PiTFT uses the GPIO header, digitally controlled pixel-by-pixel for a rock steady image. Though not a *lot* of pixels, it works great for retro gaming (and the display neatly stacks above the board, no side protuberances for video cables).

The Downside: this GPIO link entirely bypasses the Pi's video hardware, including the graphics accelerator. Many games and emulators *depend* on the GPU for performance gains. So the PiTFT has traditionally been limited to just a subset of specially-compiled emulators that can work and run well enough without the GPU.

The Solution: our latest PiTFT drivers, along with a tool called *fbcp* (framebuffer copy), careful system configuration, and (optionally) the more potent Raspberry Pi 2 board open the doors to many more gaming options. Existing emulator packages (such as RetroPie, with *dozens* of high-performance emulators and ports) — previously off-limits to the PiTFT — can run quite effectively now!

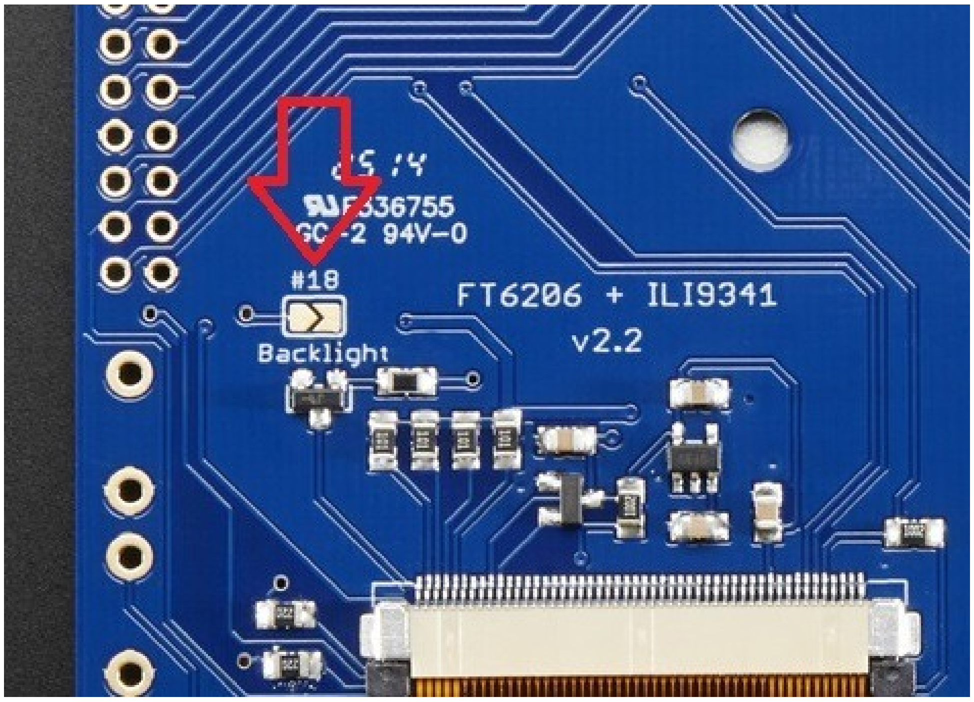
Click here to go to our FBCP tutorial!
http://adafru.it/fbe

Backlight Control

Unlike the resistive PiTFT, the capacitive version does not have a resistive touch controller chip that we can take advantage of as an extra backlight control pin. Instead, you can set up GPIO #18 as an on/off or PWM control.

Note that if you are playing audio out the headphone jack, you can't use the PWM capabilities of GPIO #18 at the same time, the PWM function is reassigned to do audio. However, you can use it as a simple on/off pin

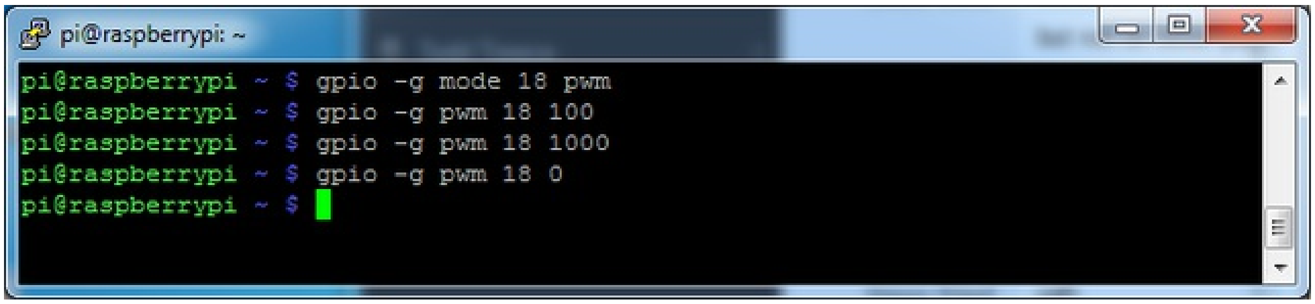
To enable using GPIO #18 as a backlight, solder closed the #18 backlight jumper on the PiTFT capacitive PCB!



OK now you can use the PWM output on GPIO 18. There's python code available for controlling the PWM pin but you can also just use the WiringPi shell commands.

With these basic shell commands, you can set the GPIO #18 pin to PWM mode, set the output to 100 (out of 1023, so dim!), set the output to 1000 (out of 1023, nearly all the way on) and 0 (off)

```
gpio -g mode 18 pwm
gpio -g pwm 18 100
gpio -g pwm 18 1000
gpio -g pwm 18 0
```



```
pi@raspberrypi ~ $ gpio -g mode 18 pwm
pi@raspberrypi ~ $ gpio -g pwm 18 100
pi@raspberrypi ~ $ gpio -g pwm 18 1000
pi@raspberrypi ~ $ gpio -g pwm 18 0
pi@raspberrypi ~ $
```

Try other numbers, from 0 (off) to 1023 (all the way on)!

Extras!

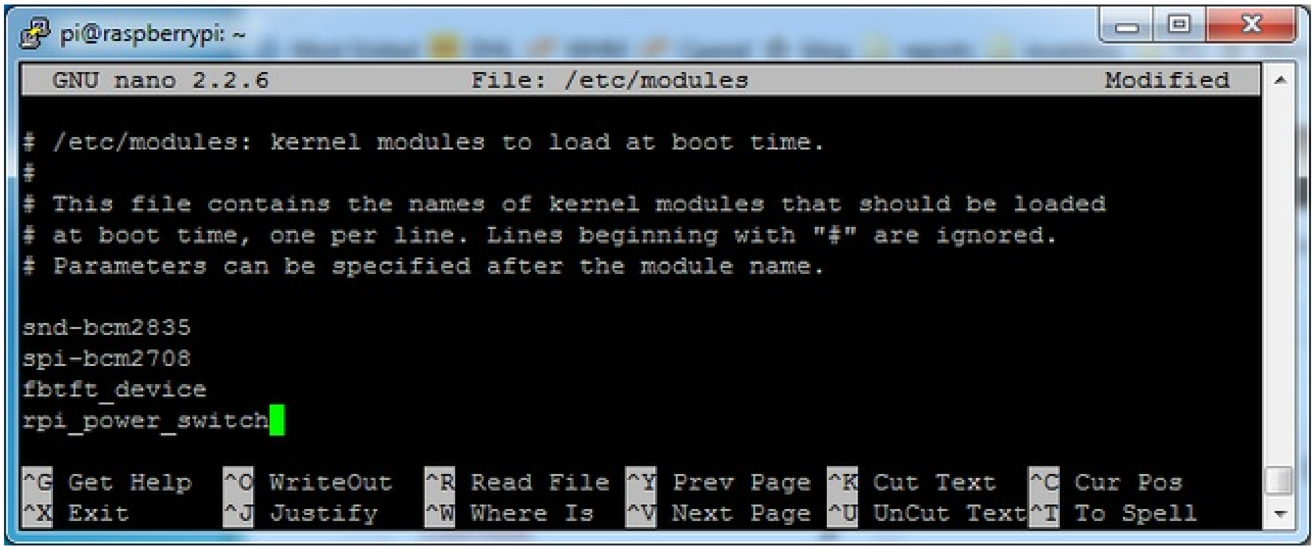
Tactile switch as power button

Its a good idea to safely turn off your Pi with a good**sudo shutdown -h now** but that often means pulling out a keyboard or connecting to the console. With our kernel we added a cool module that will let you turn any GPIO into a power button. Since there's a couple of tactile switches right there on the front, lets turn one into a power button. Press once to properly turn off the pi, press again to start it up. Isn't that nice?

We'll be using GPIO #23, the left-most button on a PiTFT 2.8", on the 2.4" HAT, #16 is a good choice since its on a tactile button. But, you can use any GPIO you want, really!

You will have to grab a pack of slim tactile switches(http://adafru.it/1489) or otherwise solder in a button

Add **rpi_power_switch** to **/etc/modules** and save



```
pi@raspberrypi: ~
  GNU nano 2.2.6              File: /etc/modules                    Modified

# /etc/modules: kernel modules to load at boot time.
#
# This file contains the names of kernel modules that should be loaded
# at boot time, one per line. Lines beginning with "#" are ignored.
# Parameters can be specified after the module name.

snd-bcm2835
spi-bcm2708
fbtft_device
rpi_power_switch

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

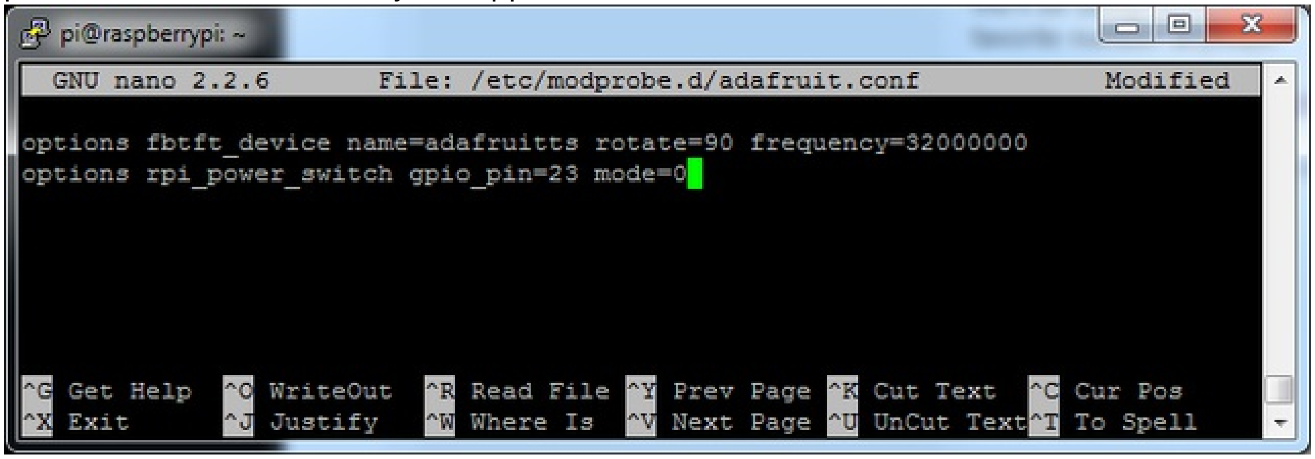
Now create a new conf file or edit our existing one with

sudo nano /etc/modprobe.d/adafruit.conf

and enter in the line

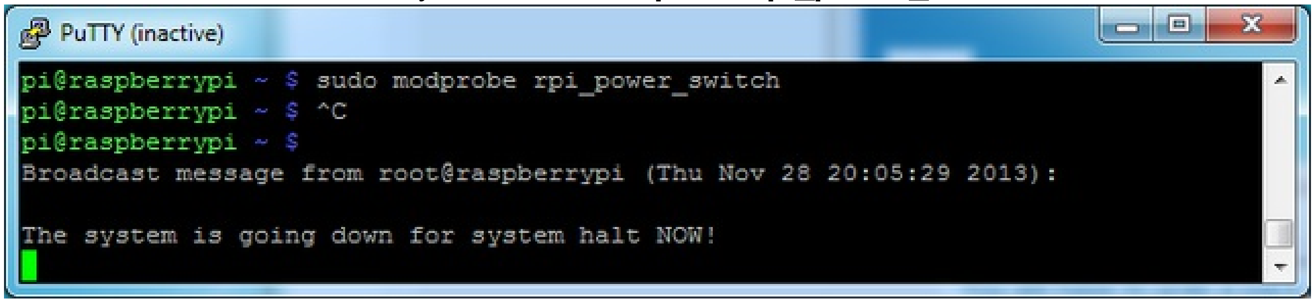
options rpi_power_switch gpio_pin=23 mode=0

Of course, change the **gpio_pin** setting to some other # if you wish. **mode=0** means its a pushbutton *not* a switch. If you happen to install an on/off switch, use **mode=1**



```
pi@raspberrypi: ~
  GNU nano 2.2.6        File: /etc/modprobe.d/adafruit.conf              Modified

options fbtft_device name=adafruitts rotate=90 frequency=32000000
options rpi_power_switch gpio_pin=23 mode=0

^G Get Help   ^O WriteOut   ^R Read File  ^Y Prev Page  ^K Cut Text   ^C Cur Pos
^X Exit       ^J Justify    ^W Where Is   ^V Next Page  ^U UnCut Text ^T To Spell
```

To make it active immediately run **sudo modprobe rpi_power_switch**

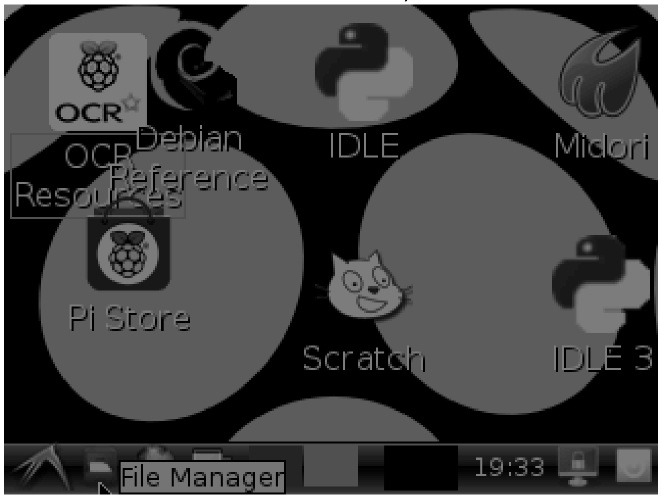


```
PuTTY (inactive)
pi@raspberrypi ~ $ sudo modprobe rpi_power_switch
pi@raspberrypi ~ $ ^C
pi@raspberrypi ~ $
Broadcast message from root@raspberrypi (Thu Nov 28 20:05:29 2013):

The system is going down for system halt NOW!
```

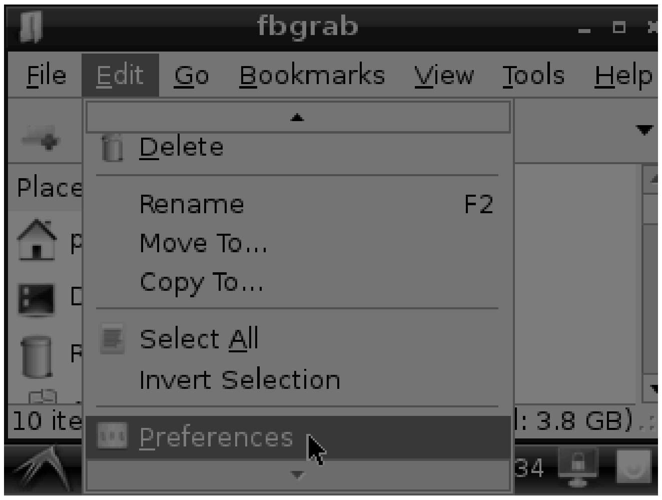
Making it easier to click icons in X

If you want to double-click on icons to launch something in X you may find it annoying to get it to work right. In LXDE you can simply set it up so that you only need to single click instead of double.

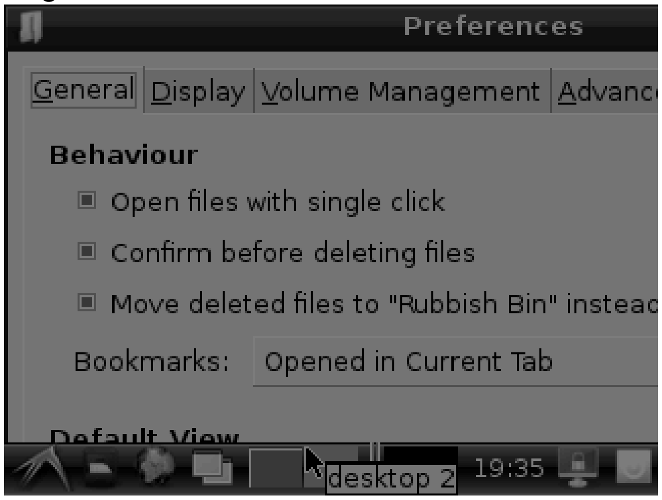
From LXDE launch the file manager (sorry these pix are grayscale, still figuring out how to screenshot the framebuffer!)



Then under the **Edit** menu, select **Preferences**



Then select **Open files with single click** and close the window (you'll need to drag it over to get to the X button



Open files with single click
Confirm before deleting files
Move deleted files to "Rubbish Bin" instead
Bookmarks: Opened in Current Tab

Boot to X Windows on PiTFT

To enable booting straight to X windows on the PiTFT follow the steps below. First make sure a display configuration which would conflict is **not** present by executing in a terminal on the Pi:

```
sudo mv /usr/share/X11/xorg.conf.d/99-fbturbo.conf ~
```

Don't worry if the command fails with an error that the file doesn't exist. This failure is normal and should happen on a good PiTFT install. You can ignore it and move on.

Next run the command below to open the nano text editor as root and create the file **/usr/share/X11/xorg.conf.d/99-pitft.conf**:

```
sudo nano /usr/share/X11/xorg.conf.d/99-pitft.conf
```

When the editor loads to a blank file, copy in the text below:

```
Section "Device"
  Identifier "Adafruit PiTFT"
  Driver "fbdev"
  Option "fbdev" "/dev/fb1"
EndSection
```

Then save the file by pressing **Ctrl-O** and then **enter**, and finally exit by pressing **Ctrl-X**.

The step above will create a configuration file which tells X windows that it should use the PiTFT framebuffer (located at /dev/fb1) by default when it runs.

At this point you can use the raspi-config tool to enable booting to desktop just like normal on the Pi. Run the following command:

```
sudo raspi-config
```

Then pick the **Enable Boot to Desktop/Scratch** option and choose if you want to boot to the console, desktop, or scratch environment. After exiting the tool and rebooting you should see the Pi load X windows on the PiTFT after (be patient it can take around 30 seconds to load).

If you want to disable booting to X, just use the raspi-config command again to choose the console boot option.

Right-click on a touchscreen

Obviously if you have a touchscreen, it cannot tell what finger you are pressing with. This means that all 'clicks' are left clicks. But if you want a right-click, you *can* do it.

Just add the following lines into your InputClass of **/etc/X11/xorg.conf.d/99-calibration.conf** after the calibration section

```
Option "EmulateThirdButton" "1"
Option "EmulateThirdButtonTimeout" "750"
Option "EmulateThirdButtonMoveThreshold" "30"
```

So for example your file will look like:

```
Section "InputClass"
   Identifier      "calibration"
   MatchProduct    "stmpe-ts"
   Option  "Calibration"   "3800 120 200 3900"
   Option  "SwapAxes"      "1"
   Option "EmulateThirdButton" "1"
```

```
  Option "EmulateThirdButtonTimeout" "750"
  Option "EmulateThirdButtonMoveThreshold" "30"
EndSection
```

This makes a right mouse click emulated when holding down the stylus for 750 ms.

(Thx adamaddin! (http://adafru.it/fH3))

Gesture Input

With the PiTFT touchscreen and xstroke (http://adafru.it/dD0) you can enter text in applications by drawing simple character gestures on the screen! Check out the video below for a short demonstration and overview of gesture input with xstroke:

Installation

Unfortunately xstroke hasn't been actively maintained for a few years so there isn't a binary package you can directly install. However compiling the tool is straightforward and easy with the steps below. Credit for these installation steps goes to mwilliams03 at ozzmaker.com (http://adafru.it/dD1).

First install a few dependencies by opening a command window on the Pi and executing:

```
sudo apt-get -y install build-essential libxft-dev libxpm-dev libxtst-dev
```

Now download, compile, and install xstroke by executing:

```
cd ~
wget http://mirror.egtvedt.no/avr32linux.org/twiki/pub/Main/XStroke/xstroke-0.6.tar.gz
tar xfv xstroke-0.6.tar.gz
cd xstroke-0.6
./configure
sed -i '/^X_LIBS = / s/$/ -lXrender -lX11 -lXext -ldl/' Makefile
make
sudo make install
```

If the commands above execute successfully xstroke should be installed. If you see an error message, carefully check the dependencies above were installed and try again.

Once xstroke is installed you will want to add a couple menu shortcuts to start and stop xstroke. Execute the following commands to install these shortcuts:

```
wget https://github.com/adafruit/PiTFT_Extras/raw/master/xstroke.desktop
wget https://github.com/adafruit/PiTFT_Extras/raw/master/xstrokekill.desktop
sudo cp xstroke*.desktop /usr/share/applications/
```

Usage

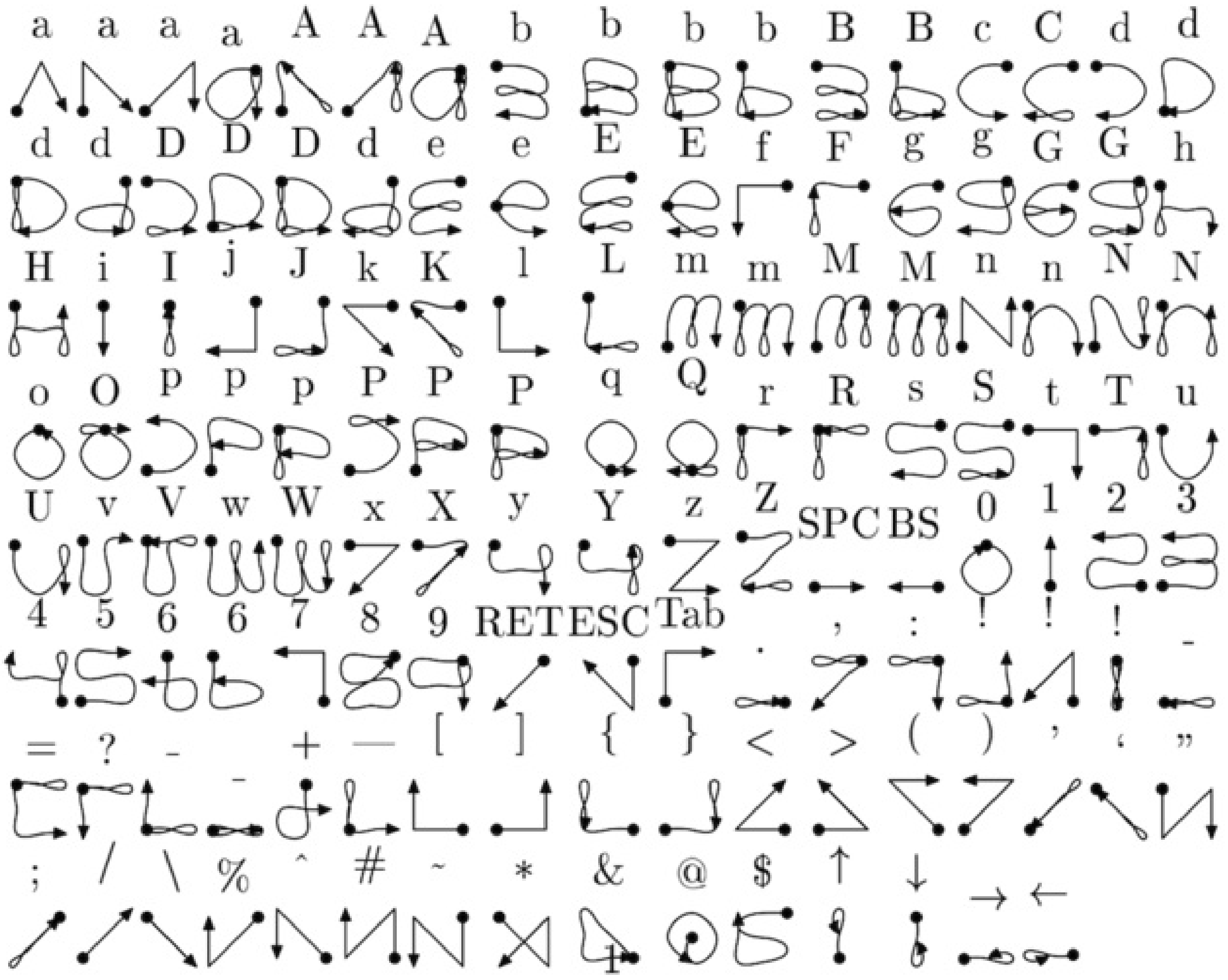
To use xstroke I highly recommend using a plastic stylus instead of your finger. Also calibrate the touchscreen for X-Windows (http://adafru.it/dD2) so you have the best control

over the cursor possible.
Don't use a ballpoint pen or sharp metal stylus as it could scratch or damage the touchscreen!
Start X-Windows on the PiTFT and open the LXDE menu by clicking the icon in the lower left corner. Scroll up to the **Accessories** menu at the top and notice the new**XStroke** and **XStroke Kill** commands.

Click the **XStroke** menu option to start xstroke. You should see a small pencil icon appear on the bottom right side of the screen. The pencil icon means xstroke is running, however by default it's not yet looking for gesture input.

Open an application that takes text input, such as LXTerminal. To enable gesture input click the xstroke pencil icon. You should see the pencil turn green and the text 'abc' written over top of the icon. You might need to click the icon a few times to get the click to register in the right spot.

When xstroke is looking for gesture input you can drag the mouse cursor in a gesture anywhere on the screen to send specific key strokes. Here's a picture of the possible gestures you can send:


(credit to Carl Worth for the image above)

To draw a gesture from the above image, press anywhere on the screen, start from the circle in the gesture, and follow the gesture pattern towards the arrow. As you draw a gesture you should see a blue line displayed that shows what you've drawn. Lift up the stylus when you get to the end of the gesture at the arrow. If xstroke recognizes the gesture it will send the appropriate key press to the active window. Try drawing a few characters from the image above to get the hang of writing gestures.

A few very useful gestures are backspace (which deletes a character), return/enter, and space. To draw a backspace gesture just draw a line going from the right side of the screen to the left side. The gesture for return/enter is a diagonal line from the top right to bottom left. Finally a space is a straight line from the left to the right.

Note that when xstroke is looking for gestures you might not be able to click or control the cursor as you normally would expect. To stop xstroke's gesture recognition carefully press the xstroke pencil icon again until the 'abc' text disappears. I've found this process can be a little finicky as the icon is very small and any movement will be interpreted as a gesture. Use a light touch and try a few times to click the icon.

If you get stuck completely and can't disable xstroke by clicking the icon, connect to the Raspberry Pi in a terminal/SSH connection and run 'killall xstroke' (without quotes) to force xstroke to quit. The normal way to stop xstroke is to navigate to the **Accessories** -> **XStroke Kill** command, but you might not be able to do that if xstroke is listening for gesture input.

Have fun using xstroke to control your Pi by writing gestures on the PiTFT screen!

PiTFT PyGame Tips

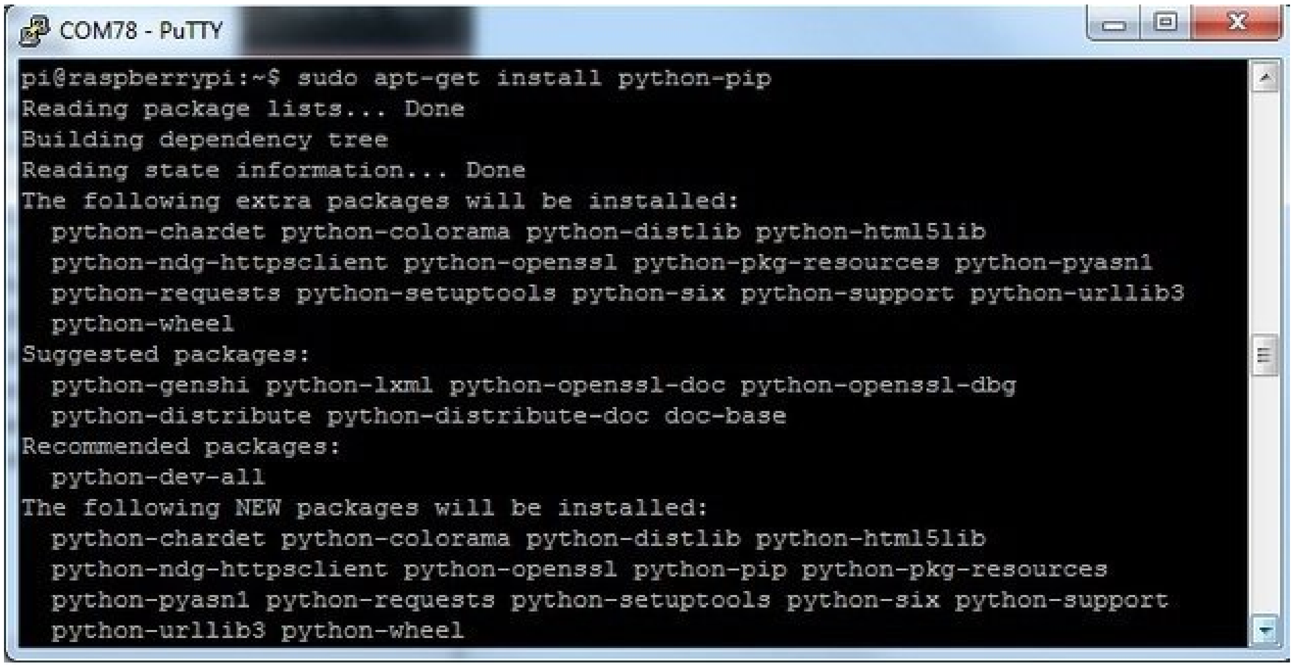
Since the PiTFT screen is fairly small, you may need to write custom UI programs. Pygame is the easiest way by far to do this.

Jeremy Blythe has an excellent tutorial here on getting started.(http://adafru.it/saw)

However, *before* you follow that link you'll want to set up pygame for the best compatibility:

Install pip & pygame

Install Pip: **sudo apt-get install python-pip**

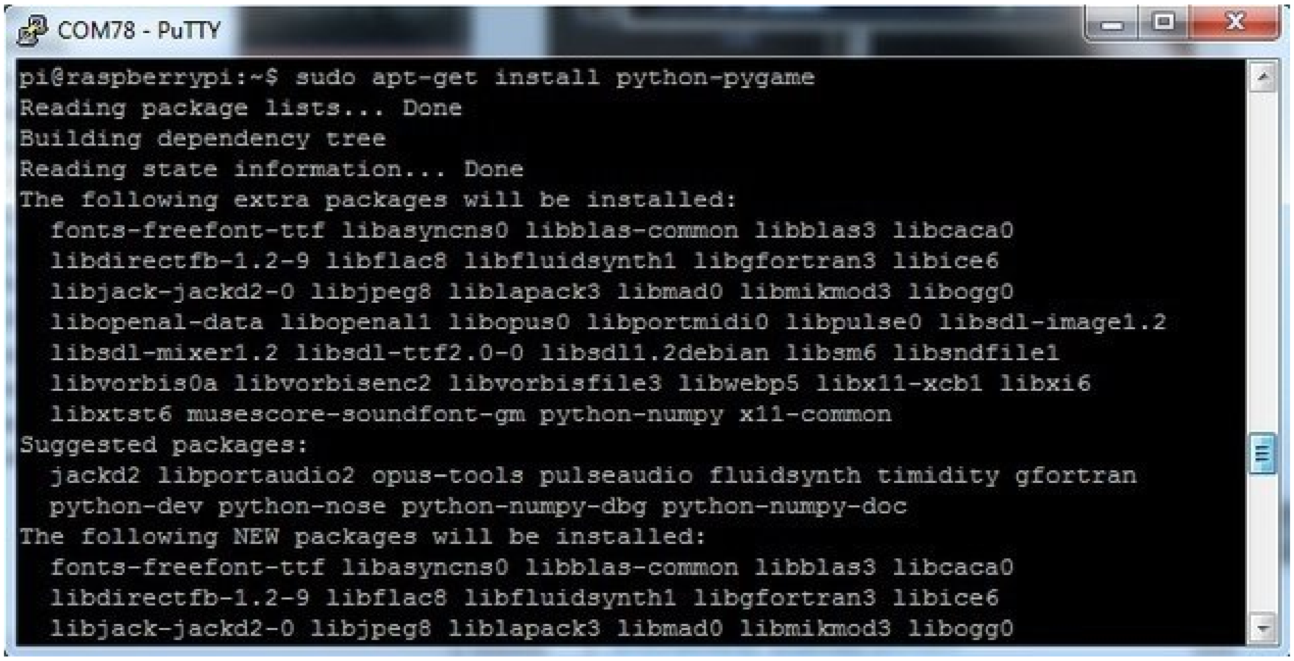


COM78 - PuTTY [redacted]

```
pi@raspberrypi:~$ sudo apt-get install python-pip
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  python-chardet python-colorama python-distlib python-html5lib
  python-ndg-httpsclient python-openssl python-pkg-resources python-pyasn1
  python-requests python-setuptools python-six python-support python-urllib3
  python-wheel
Suggested packages:
  python-genshi python-lxml python-openssl-doc python-openssl-dbg
  python-distribute python-distribute-doc doc-base
Recommended packages:
  python-dev-all
The following NEW packages will be installed:
  python-chardet python-colorama python-distlib python-html5lib
  python-ndg-httpsclient python-openssl python-pip python-pkg-resources
  python-pyasn1 python-requests python-setuptools python-six python-support
  python-urllib3 python-wheel
```

Install Pygame: **sudo apt-get install python-pygame**

(this will take a while)



```
pi@raspberrypi:~$ sudo apt-get install python-pygame
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  fonts-freefont-ttf libasyncns0 libblas-common libblas3 libcaca0
  libdirectfb-1.2-9 libflac8 libfluidsynth1 libgfortran3 libice6
  libjack-jackd2-0 libjpeg8 liblapack3 libmad0 libmikmod3 libogg0
  libopenal-data libopenal1 libopus0 libportmidi0 libpulse0 libsdl-image1.2
  libsdl-mixer1.2 libsdl-ttf2.0-0 libsdl1.2debian libsm6 libsndfile1
  libvorbis0a libvorbisenc2 libvorbisfile3 libwebp5 libx11-xcb1 libxi6
  libxtst6 musescore-soundfont-gm python-numpy x11-common
Suggested packages:
  jackd2 libportaudio2 opus-tools pulseaudio fluidsynth timidity gfortran
  python-dev python-nose python-numpy-dbg python-numpy-doc
The following NEW packages will be installed:
  fonts-freefont-ttf libasyncns0 libblas-common libblas3 libcaca0
  libdirectfb-1.2-9 libflac8 libfluidsynth1 libgfortran3 libice6
  libjack-jackd2-0 libjpeg8 liblapack3 libmad0 libmikmod3 libogg0
```

Ensure you are running SDL 1.2

SDL 2.x and SDL 1.2.15-10 have some serious incompatibilities with touchscreen. You can force SDL 1.2 by running a script. (Thanks to heine in the forums!(http://adafru.it/sax))

Edit a new file with **sudo nano installsdl.sh**
and paste in the following text:

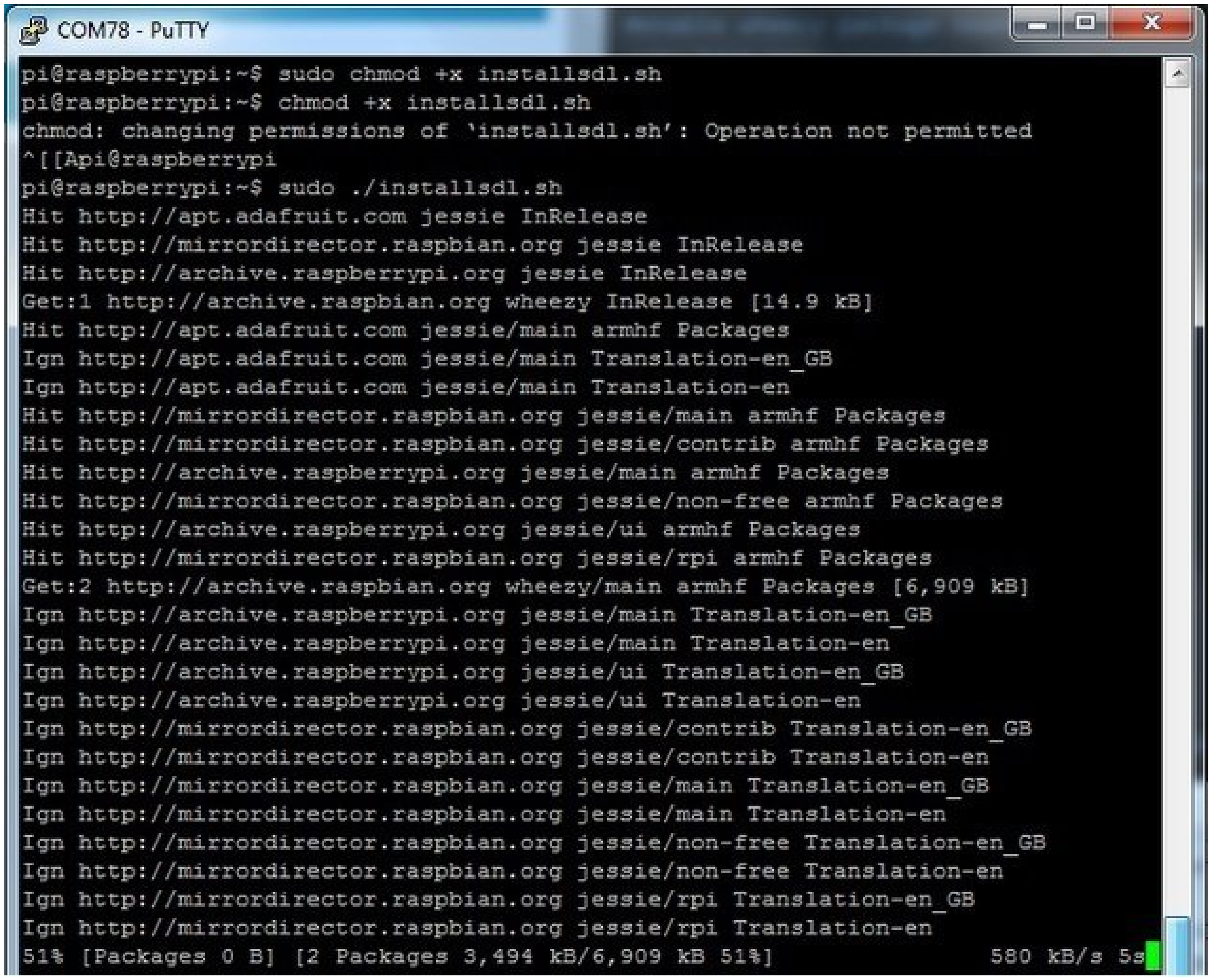
```
#!/bin/bash

#enable wheezy package sources
echo "deb http://archive.raspbian.org/raspbian wheezy main
" > /etc/apt/sources.list.d/wheezy.list

#set stable as default package source (currently jessie)
echo "APT::Default-release \"stable\";
" > /etc/apt/apt.conf.d/10defaultRelease

#set the priority for libsdl from wheezy higher then the jessie package
echo "Package: libsdl1.2debian
Pin: release n=jessie
Pin-Priority: -10
Package: libsdl1.2debian
Pin: release n=wheezy
Pin-Priority: 900
" > /etc/apt/preferences.d/libsdl

#install
apt-get update
```

apt-get -y --force-yes install libsdl1.2debian/wheezy

run

sudo chmod +x installsdl.sh

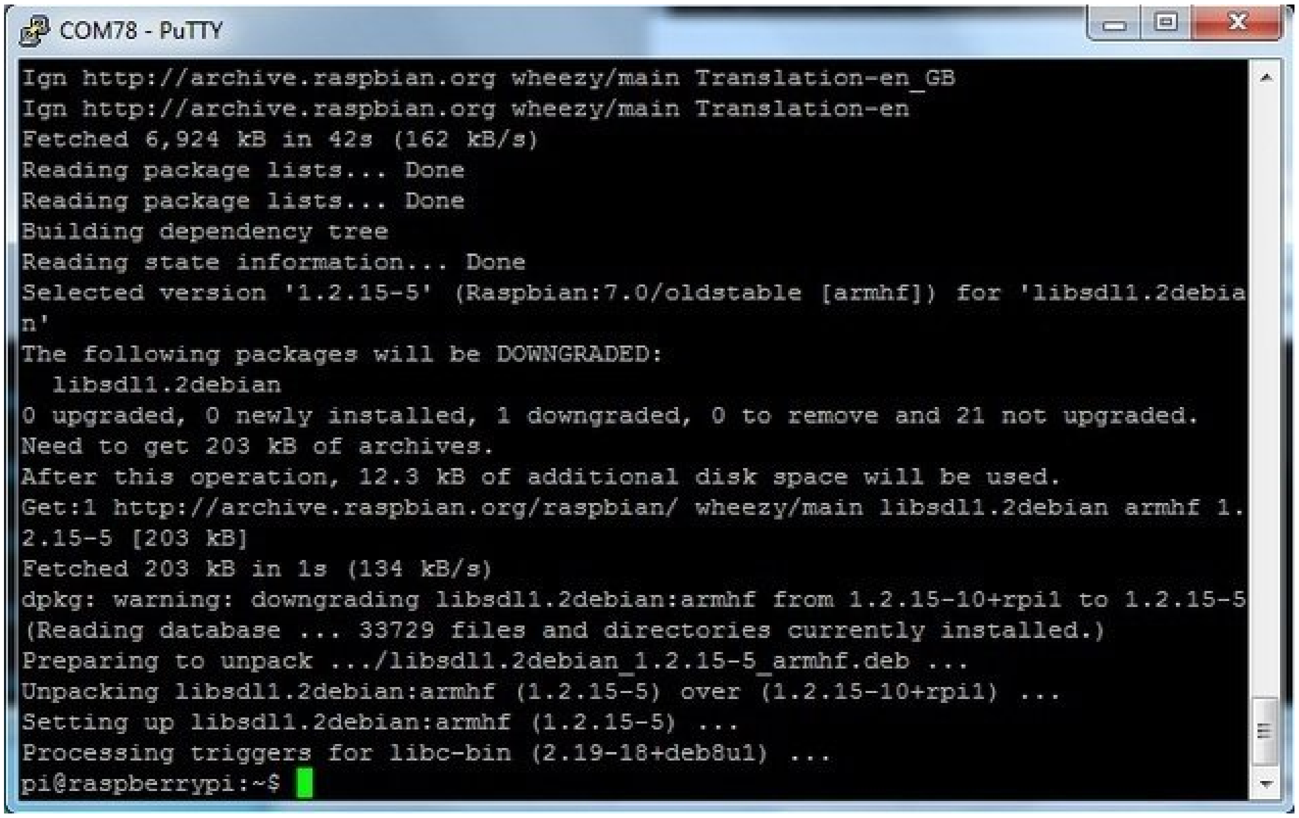
sudo ./installsdl.sh



COM78 - PuTTY

```
pi@raspberrypi:~$ sudo chmod +x installsdl.sh
pi@raspberrypi:~$ chmod +x installsdl.sh
chmod: changing permissions of 'installsdl.sh': Operation not permitted
^[[Api@raspberrypi
pi@raspberrypi:~$ sudo ./installsdl.sh
Hit http://apt.adafruit.com jessie InRelease
Hit http://mirrordirector.raspbian.org jessie InRelease
Hit http://archive.raspberrypi.org jessie InRelease
Get:1 http://archive.raspbian.org wheezy InRelease [14.9 kB]
Hit http://apt.adafruit.com jessie/main armhf Packages
Ign http://apt.adafruit.com jessie/main Translation-en_GB
Ign http://apt.adafruit.com jessie/main Translation-en
Hit http://mirrordirector.raspbian.org jessie/main armhf Packages
Hit http://mirrordirector.raspbian.org jessie/contrib armhf Packages
Hit http://archive.raspberrypi.org jessie/main armhf Packages
Hit http://mirrordirector.raspbian.org jessie/non-free armhf Packages
Hit http://archive.raspberrypi.org jessie/ui armhf Packages
Hit http://mirrordirector.raspbian.org jessie/rpi armhf Packages
Get:2 http://archive.raspbian.org wheezy/main armhf Packages [6,909 kB]
Ign http://archive.raspberrypi.org jessie/main Translation-en_GB
Ign http://archive.raspberrypi.org jessie/main Translation-en
Ign http://archive.raspberrypi.org jessie/ui Translation-en_GB
Ign http://archive.raspberrypi.org jessie/ui Translation-en
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/contrib Translation-en
Ign http://mirrordirector.raspbian.org jessie/main Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/main Translation-en
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/non-free Translation-en
Ign http://mirrordirector.raspbian.org jessie/rpi Translation-en_GB
Ign http://mirrordirector.raspbian.org jessie/rpi Translation-en
51% [Packages 0 B] [2 Packages 3,494 kB/6,909 kB 51%]            580 kB/s 5s
```

it will force install SDL 1.2


COM78 - PuTTY

```
Ign http://archive.raspbian.org wheezy/main Translation-en_GB
Ign http://archive.raspbian.org wheezy/main Translation-en
Fetched 6,924 kB in 42s (162 kB/s)
Reading package lists... Done
Reading package lists... Done
Building dependency tree
Reading state information... Done
Selected version '1.2.15-5' (Raspbian:7.0/oldstable [armhf]) for 'libsdl1.2debia
n'
The following packages will be DOWNGRADED:
  libsdl1.2debian
0 upgraded, 0 newly installed, 1 downgraded, 0 to remove and 21 not upgraded.
Need to get 203 kB of archives.
After this operation, 12.3 kB of additional disk space will be used.
Get:1 http://archive.raspbian.org/raspbian/ wheezy/main libsdl1.2debian armhf 1.
2.15-5 [203 kB]
Fetched 203 kB in 1s (134 kB/s)
dpkg: warning: downgrading libsdl1.2debian:armhf from 1.2.15-10+rpi1 to 1.2.15-5
(Reading database ... 33729 files and directories currently installed.)
Preparing to unpack .../libsdl1.2debian_1.2.15-5_armhf.deb ...
Unpacking libsdl1.2debian:armhf (1.2.15-5) over (1.2.15-10+rpi1) ...
Setting up libsdl1.2debian:armhf (1.2.15-5) ...
Processing triggers for libc-bin (2.19-18+deb8u1) ...
pi@raspberrypi:~$ 
```

OK **now** you can continue with pygame

F.A.Q.

The display works, but the capacitive touch part doesnt

Check that you installed the right image, there's one for resistive and one for capacitive PiTFT's

If that doesn't help, you can verify your RasPi model number with the command**cat /proc/cpuinfo**, if it's revision #**0002** or **0003** that means it's a rev 1 Model B, (http://adafru.it/dXg) and will not work due to the I2C pins changing.

Does this screen do multi-touch?
Nope! This capacitive touch screen is single-touch only.
Hey...I was looking at the FT6206 datasheet and it looks like it can support multitouch (two points)!
The chip does in fact support multitouch, but the screen layout itself is single-touch.
We'll keep looking for a low cost multitouch screen, but we found that at the small size of this screen, single-touch is pretty good! Also, very few linux programs support MT.
How do I automatically boot to X windows on the PiTFT?

Check out the 2.8" resistive PiTFT FAQ (http://adafru.it/dJ2) for an answer to this common question.

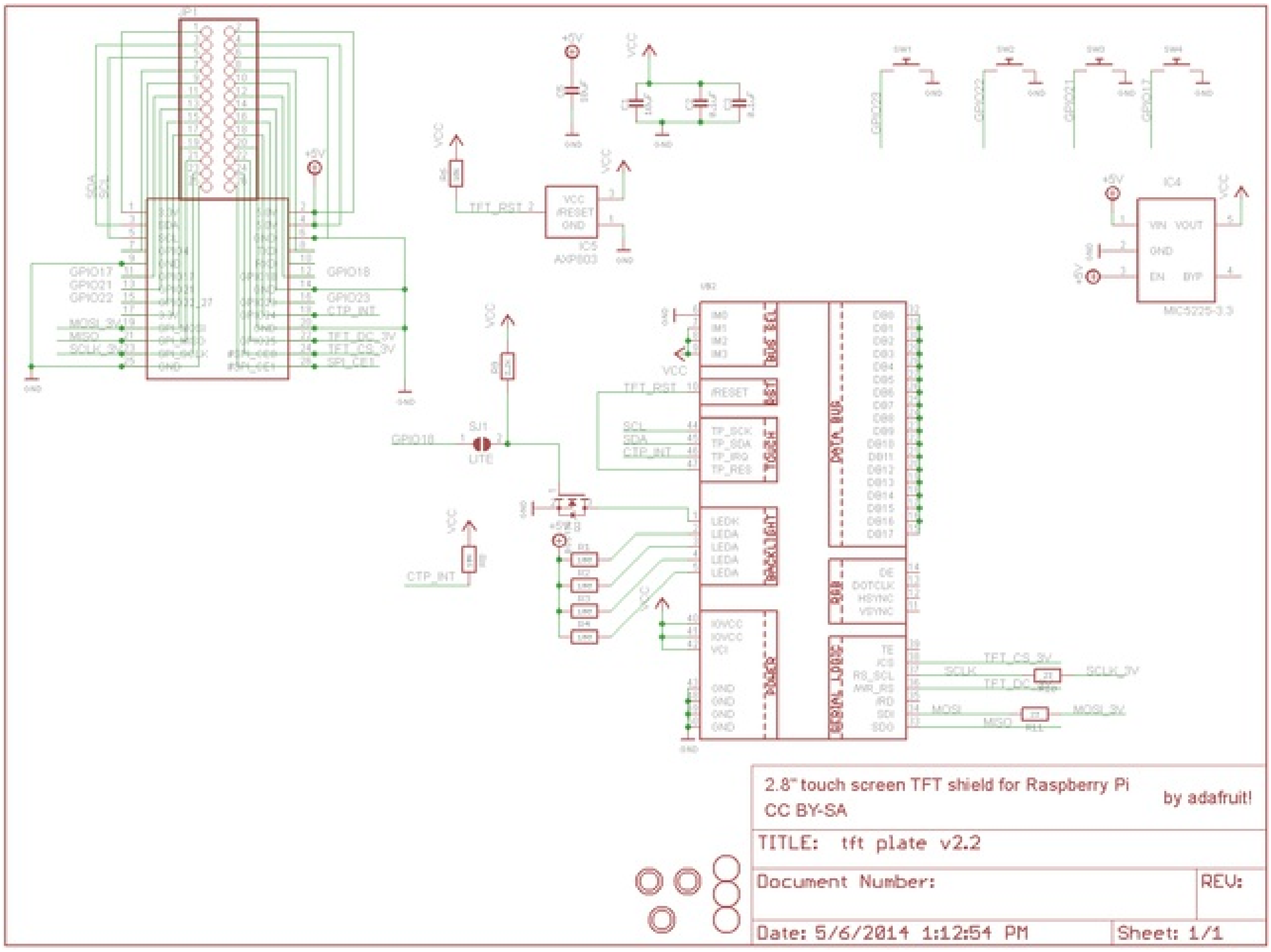
I have some more questions!
Check out the 2.8" Resistive PiTFT FAQ page for some other questions you may want answered
Visit the 2.8" Resistive PiTFT FAQ page
http://adafru.it/dJ2

Downloads

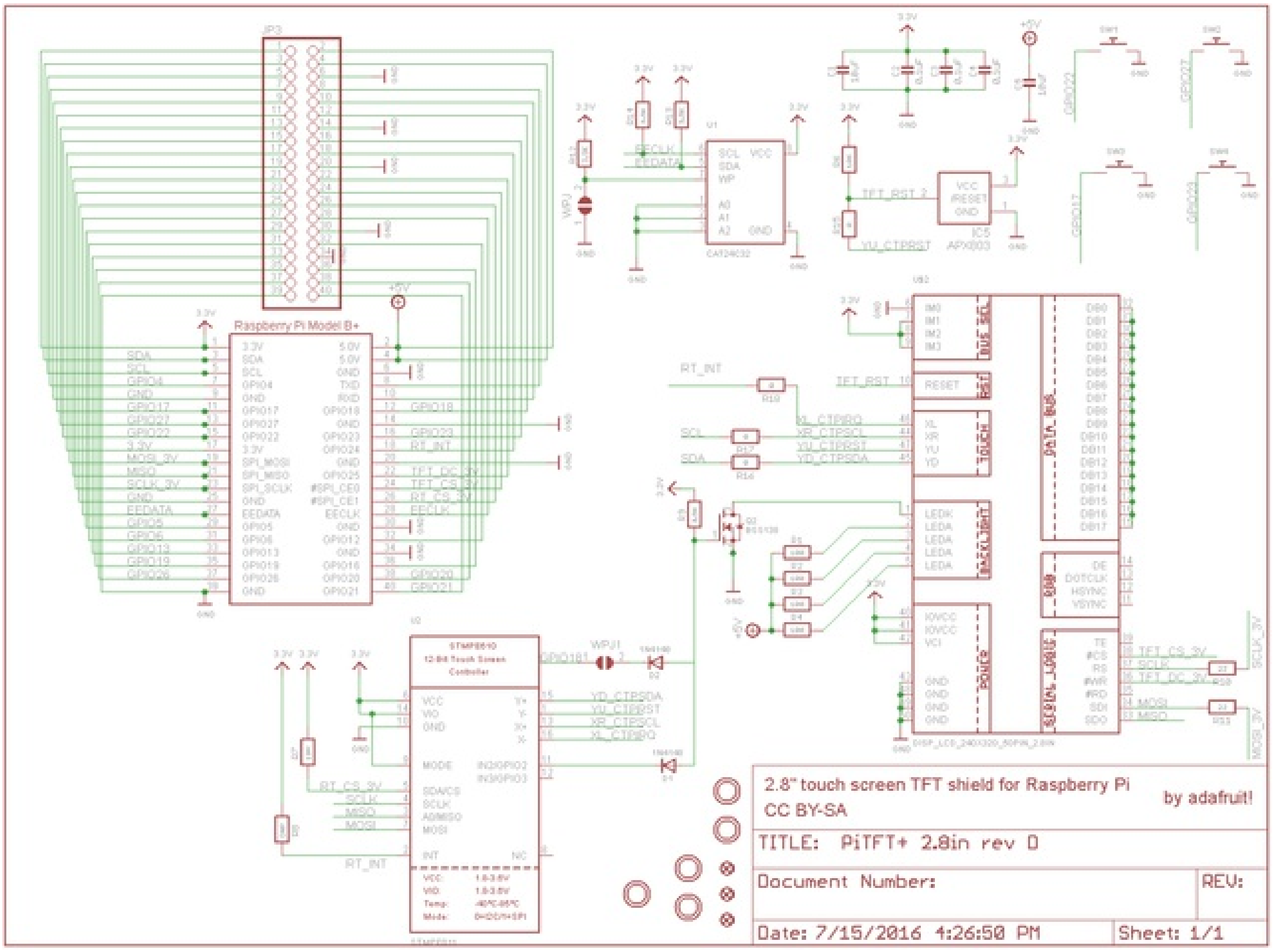
Files

- The latest kernel fork that adds all the TFT, touchscreen, and other addons is here on github (http://adafru.it/aPa)
- Datasheet for the 'raw' 2.8" TFT display (http://adafru.it/sEt)
- FT6206 Datasheet (http://adafru.it/sEu) & App note (http://adafru.it/dRn) (capacitive chip)
- EagleCAD PCB files on GitHub (http://adafru.it/oYC)

Schematic for Pi 1 Version

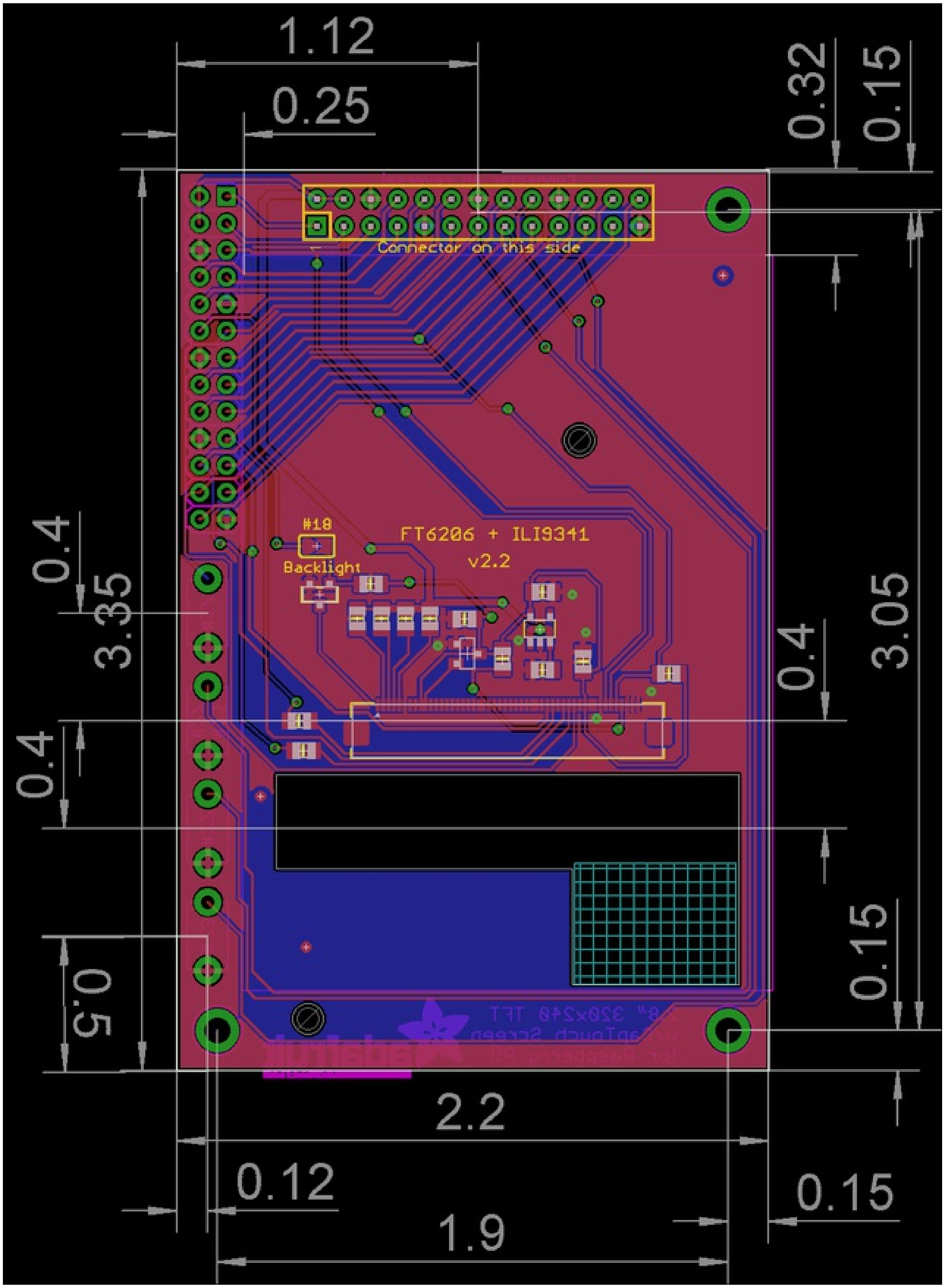


Schematic for PiTFT Plus (B+/Pi 2 shape)



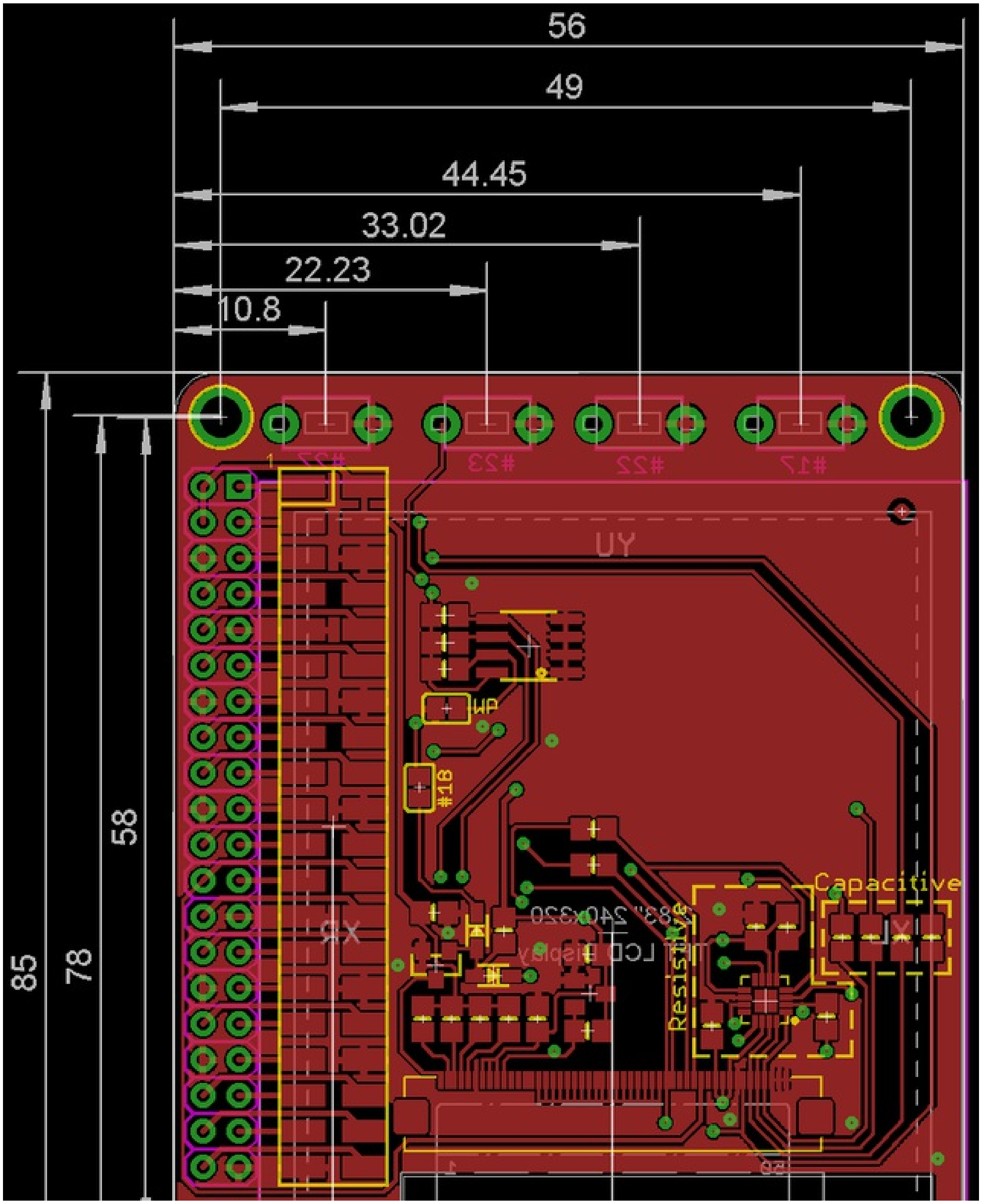
Fabrication Print (Pi 1 Version)

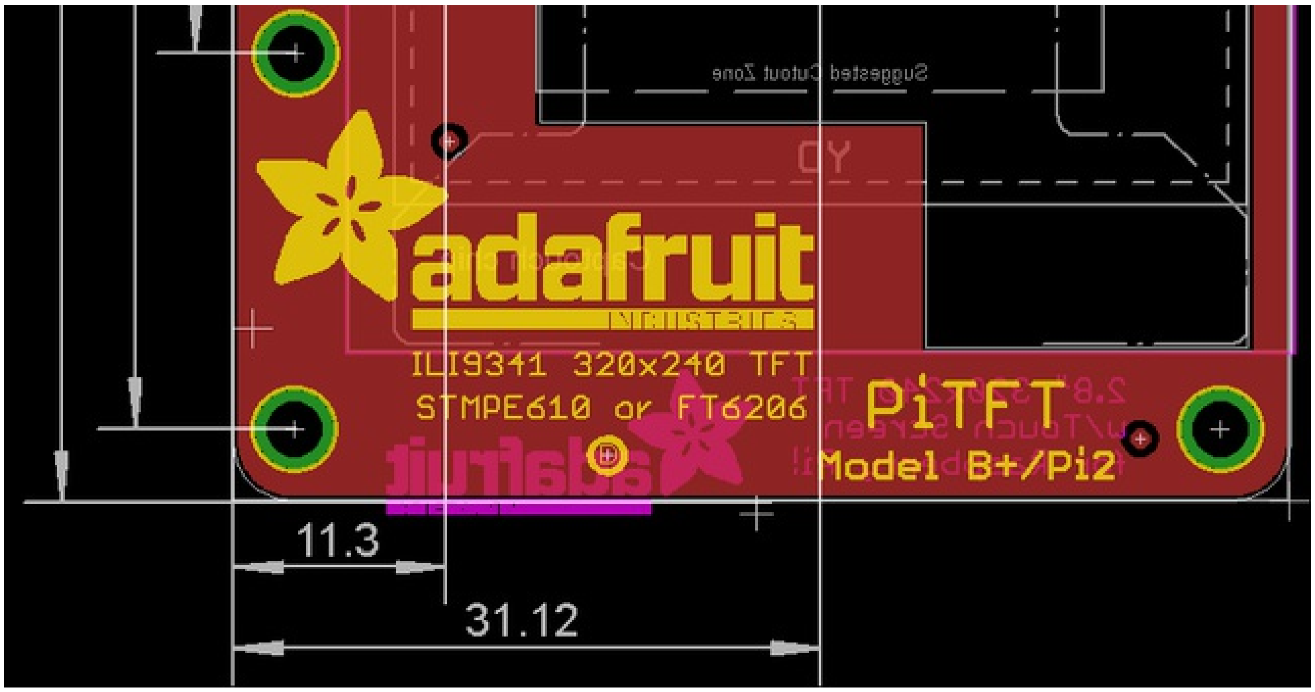
Dimensions in Inches


1.12
0.25
0.32
0.15
Connector on this side
#18
Backlight
FT6206 + ILI9341
v2.2
0.4
3.35
0.4
0.5
3.05
0.4
0.15
2.2
0.12
1.9
0.15

Fabrication Print (B+/Pi 2 Version)

Dimensions in mm




Suggested Cutout Zone
adafruit
INDUSTRIES
ILI9341 320x240 TFT
STMPE610 or FT6206
PiTFT
Model B+/Pi2
11.3
31.12