

□

Adafruit 2.8" PiTFT - Capacitive Touch

Created by lady ada



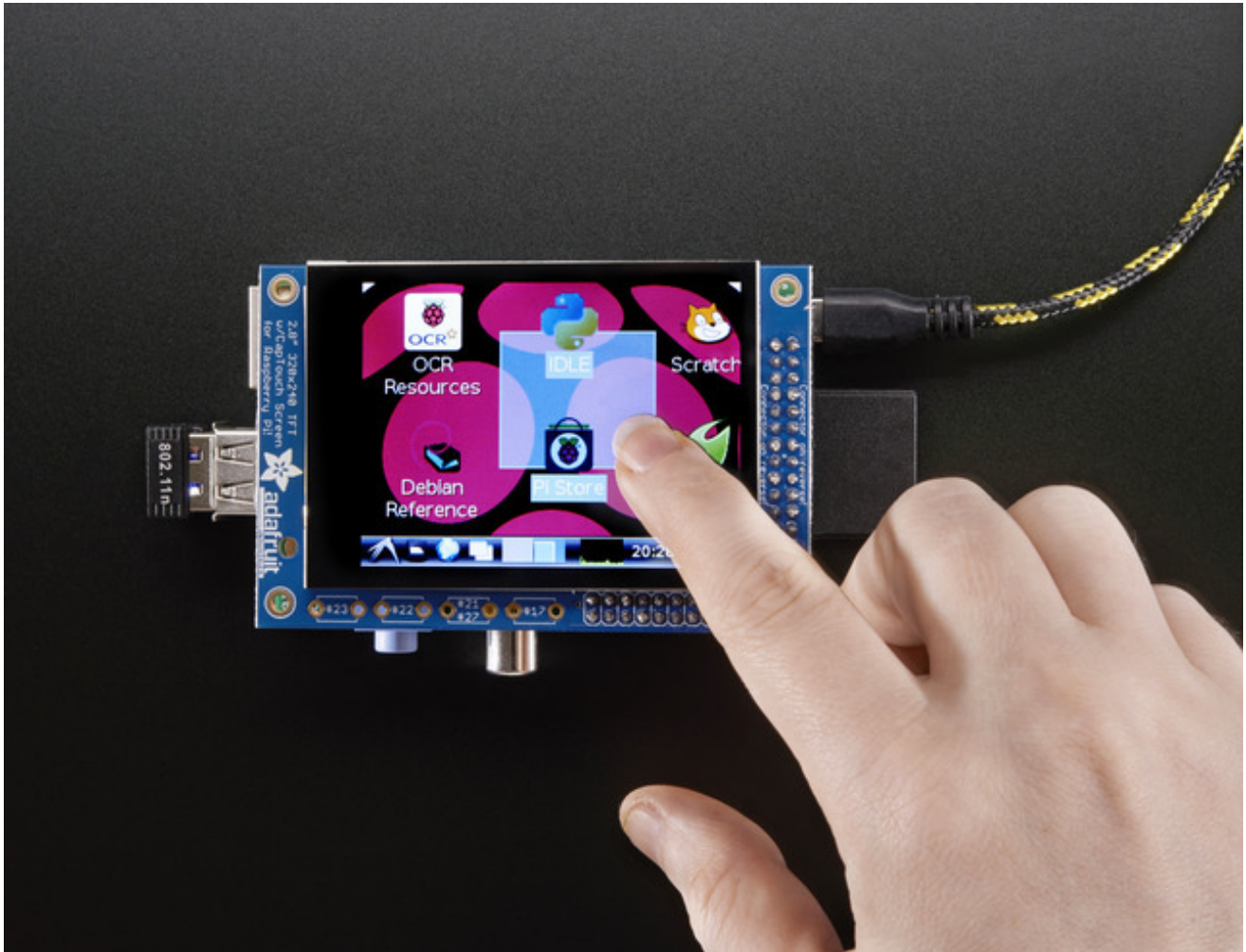
Last updated on 2016-10-01 06:53:52 PM UTC

Guide Contents

Guide Contents	2
Overview	4
Assembly	9
Easy Install	10
Ready to go image	10
DIY Installer script	10
Detailed Installation	15
Before you start	15
Download & Install Kernel	16
Capacitive Touchscreen	25
Event Testing	26
TSLIB calibration	28
X11 Calibration	29
Calibration for other rotations	30
Using the Console	32
Backlight Control	33
Playing Videos	35
Displaying Images	36
Using FBCP	37
Extras!	39
Tactile switch as power button	39
Making it easier to click icons in X	40
Right-click on a touchscreen	42
Gesture Input	43
PiTFT Pygame Tips	44
Install pip & pygame	44
Ensure you are running SDL 1.2	45
F.A.Q.	48
Downloads	49
Files	49

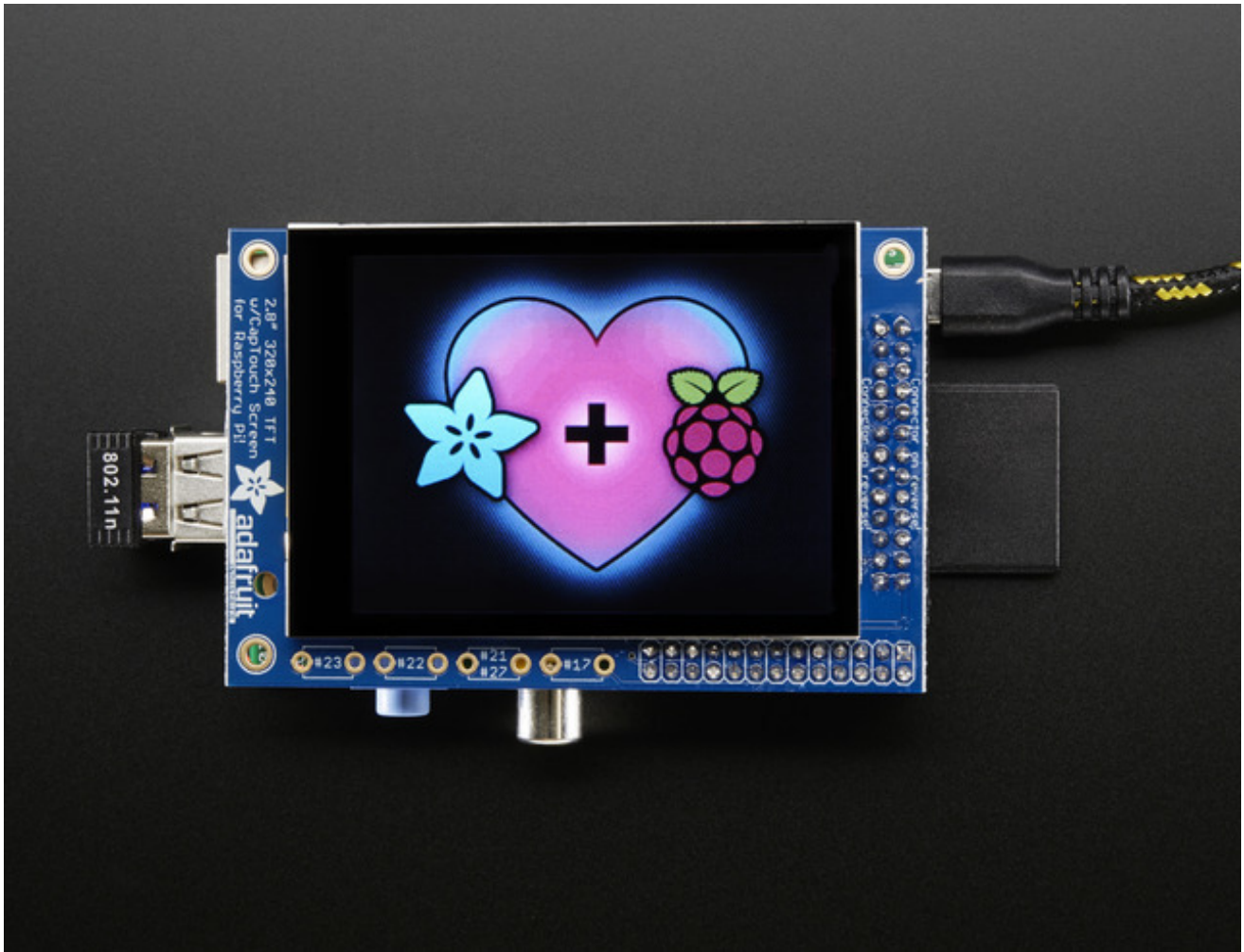
Schematic for Pi 1 Version	49
Schematic for PiTFT Plus (B+/Pi 2 shape)	50
Fabrication Print (Pi 1 Version)	50
Fabrication Print (B+/Pi 2 Version)	52

Overview

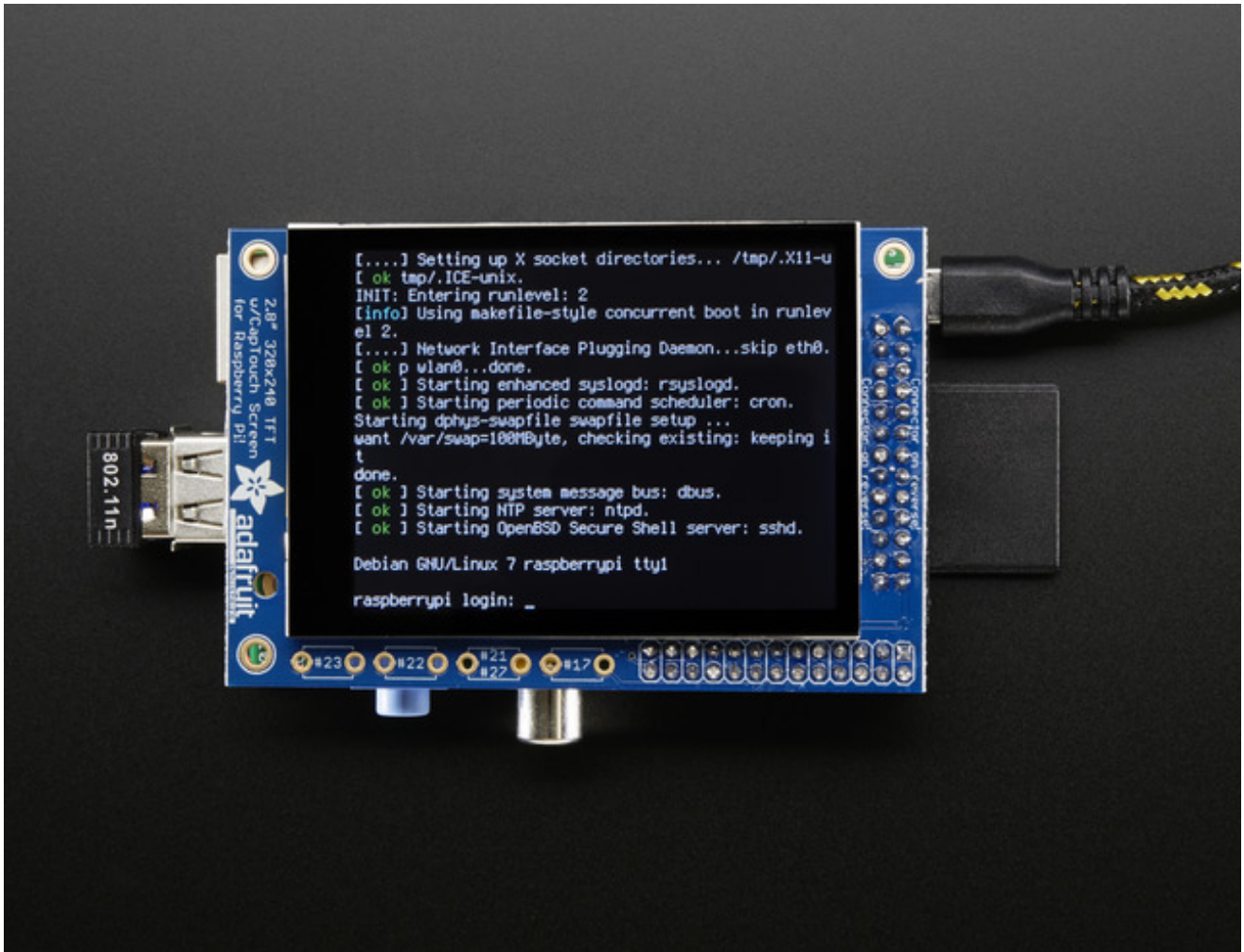


Our best-selling PiTFT just got a fancy upgrade, now we have a version with a **capacitive touchscreen!** That's right, instead of a resistive touchscreen, which requires a fingernail or stylus, you can now use a fingerpad. The screen looks much nicer, with a black bezel and glass overlay.

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.

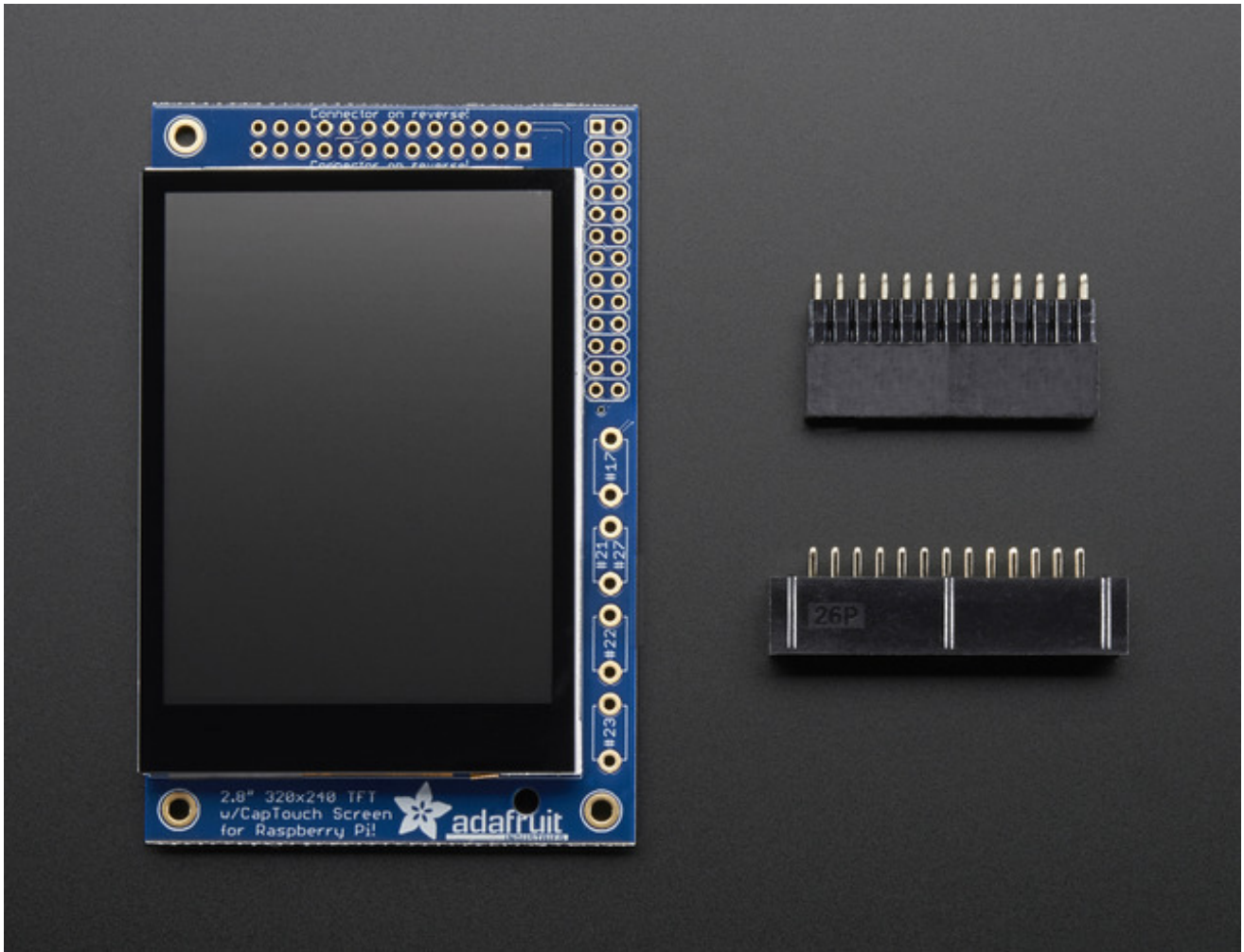


The screen is the same size as the resistive type so you can use this with the PiTFT PiBow or any other enclosure you may already have. We also use the same SDL device and signals so PyGame and X11 based programs can be swapped in with no changes in code.



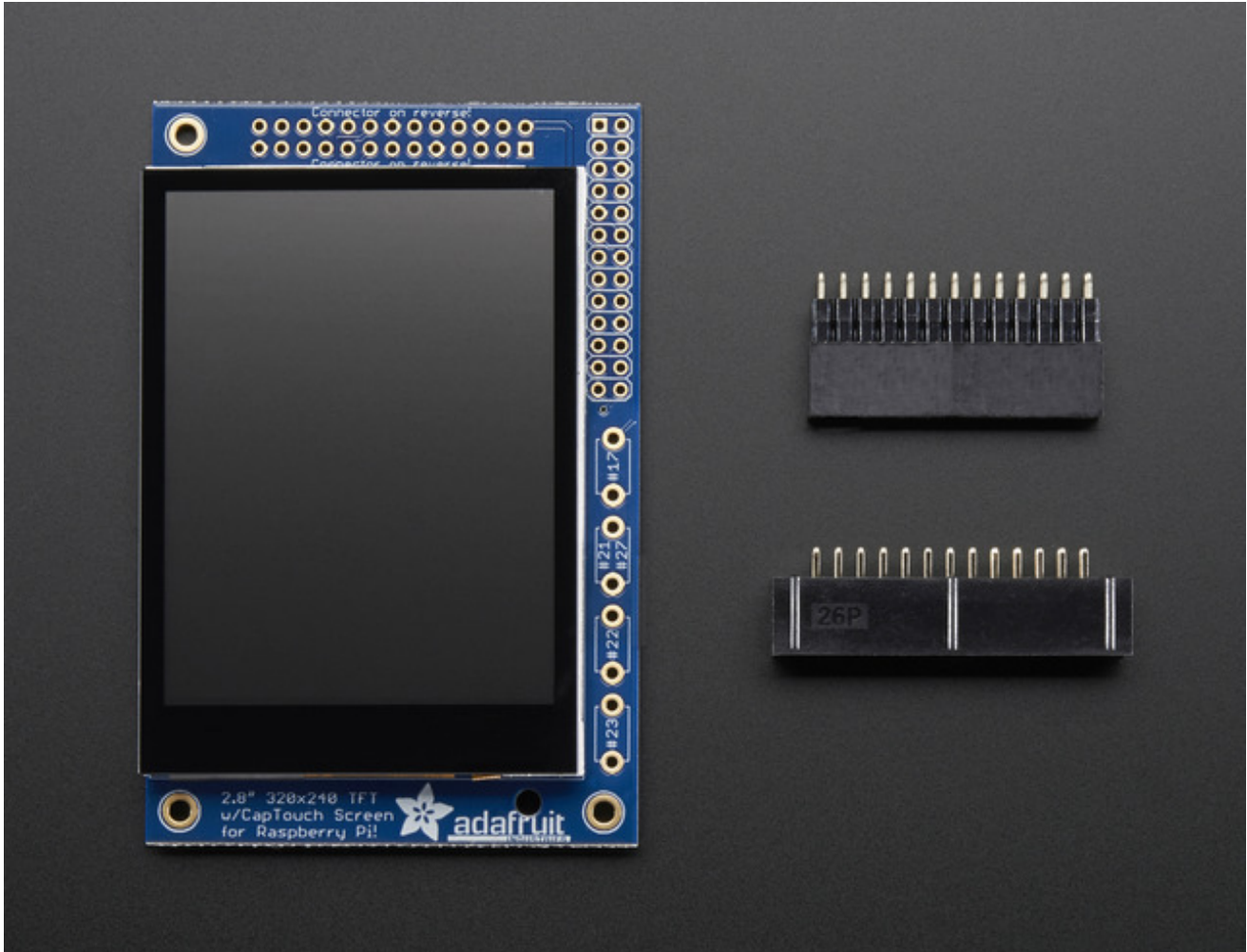
It's designed to fit nicely onto the Pi Model A or B rev 2 but also works perfectly fine with the Model B+ as long as you don't mind the PCB overhangs the USB ports by 5mm, see the photos above. Model B rev 1 have an older layout for the I2C pins and won't be able to use the touch screen

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. \(http://adafru.it/1489\)](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. \(http://adafru.it/1112\)](http://adafru.it/1112)



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)

<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 \(http://adafru.it/aMW\)](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 \(March 25, 2015\)](http://adafru.it/mAc)

<http://adafru.it/mAc>

[Download Jessie Lite-based PiTFT 2.8" Capacitive Image for Pi 1, Pi 2 and Pi 3 \(March 25, 2015\)](http://adafru.it/mAd)

<http://adafru.it/mAd>

Older images:

- [Raspbian Jessie 2015/09/24-based image \(http://adafru.it/iDy\)](http://adafru.it/iDy)
- [Raspbian Wheezy 2015/09/24-based image \(http://adafru.it/idz\)](http://adafru.it/idz)
- [Raspbian 2014/09/18-based image \(http://adafru.it/e11\)](http://adafru.it/e11)
- [Raspbian 2014/06/20-based image \(http://adafru.it/dSO\)](http://adafru.it/dSO)
- [Raspbian image from 2015/03/03 \(http://adafru.it/eUI\)](http://adafru.it/eUI)

The DIY Installer isn't working right now, please try the All-In-One image above - no ETA on why its not working, something to do with the latest Raspbian has changed. Thanks!

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 \(http://adafru.it/eIn\)](http://adafru.it/eIn) if you are interested in how it works

To download and run it, simply run the following commands:

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

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 two do 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
```

```
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
```

```
pi@raspberrypi ~/Adafruit-Occidentalis $ sudo apt-get install adafruit-pitft-helper
Reading package lists... Done
Building dependency tree... 50%
```

It's normal for the Pi to pause at this step for up to 20 minutes, there's a lot of kernel software to replace

OK now the kernel is 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: ~
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: ~
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

The DIY Installer isn't working right now, please try the All-In-One image above - no ETA on why its not working, something to do with the latest Raspbian has changed. Thanks!
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

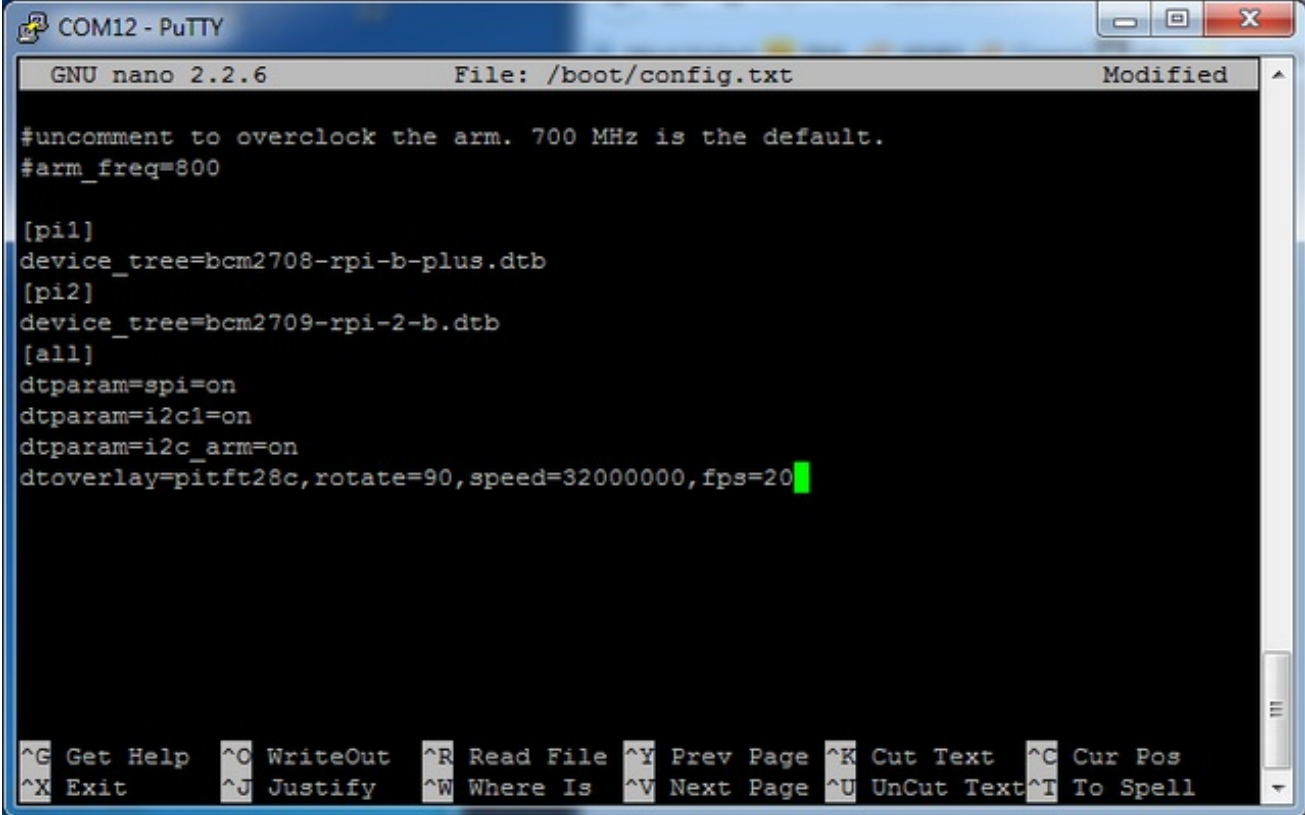
pi@raspberrypi ~/Adafruit-Occidentalis $ sudo apt-get install raspberrypi-bootloader
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following extra packages will be installed:
  libraspberrypi-bin libraspberrypi-dev libraspberrypi-doc libraspberrypi0
The 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
```



```
COM12 - PuTTY
GNU nano 2.2.6 File: /boot/config.txt Modified
#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
^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
```

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 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

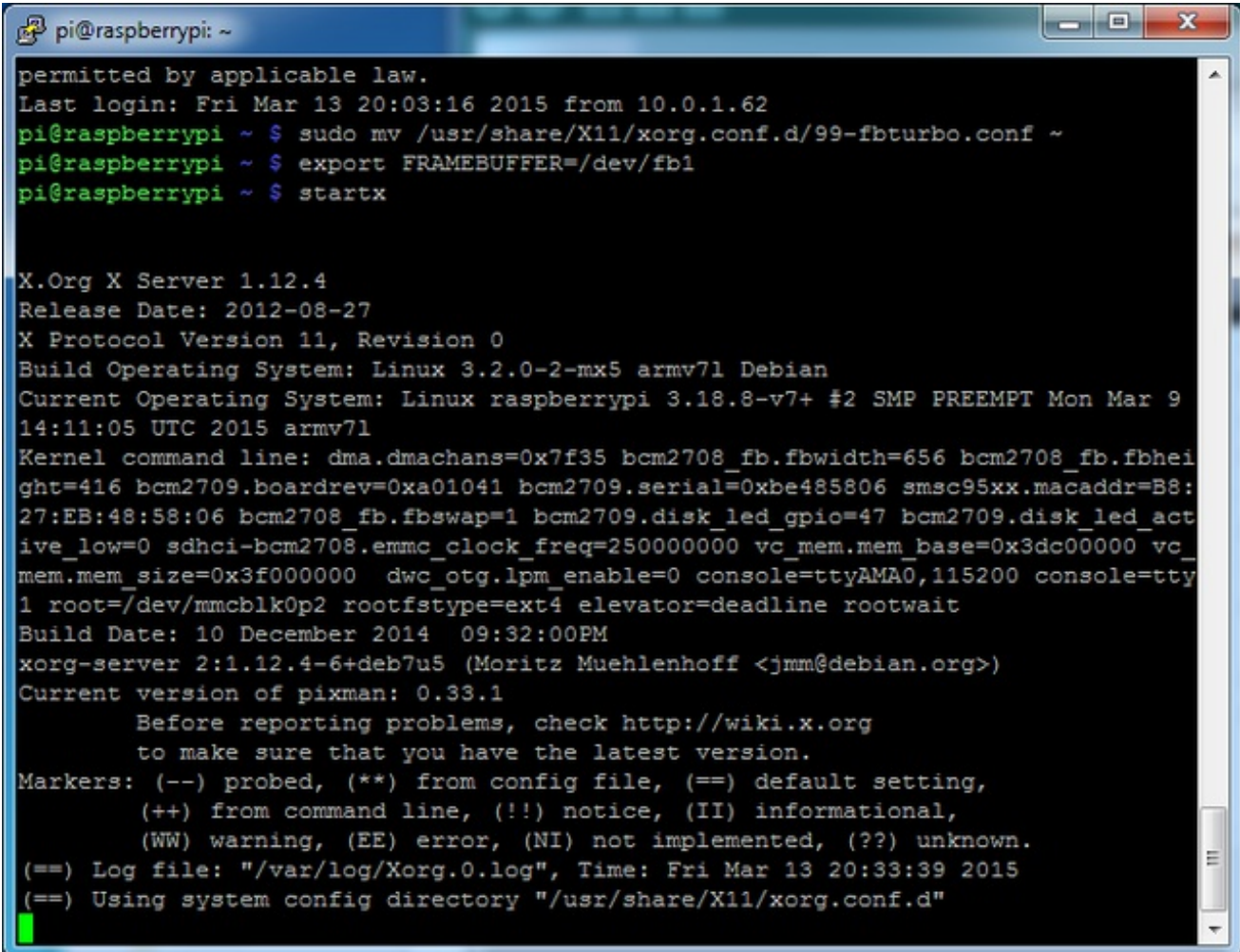
sudo reboot (if you have the TFT plate installed already)

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
```



```
pi@raspberrypi: ~
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"
```

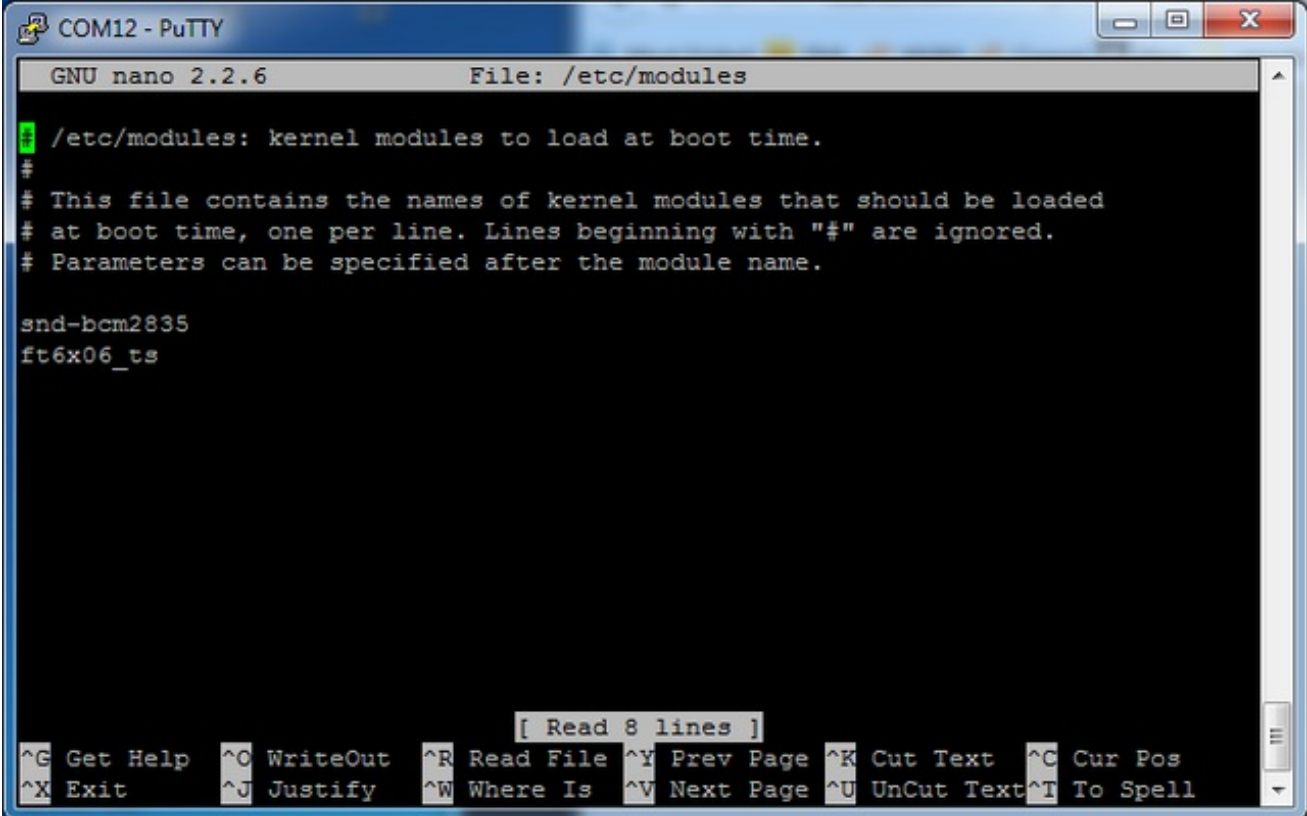
You should see the Pi desktop show up on the TFT! Congrats, you've completed the first test perfectly.

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

```
COM12 - PuTTY
[ 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`


```
COM3 - PuTTY
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

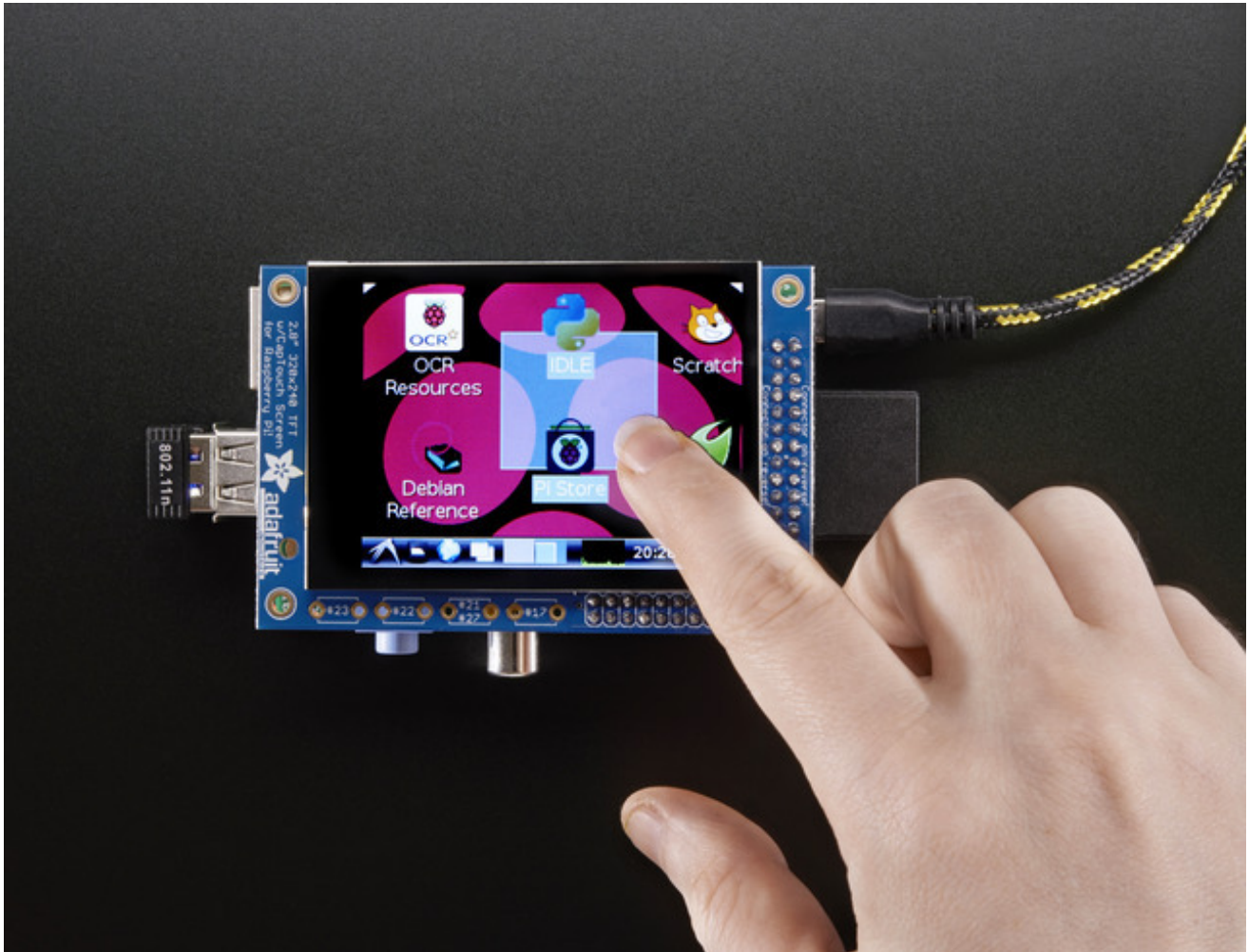
export FRAMEBUFFER=/dev/fb1

# 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

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.

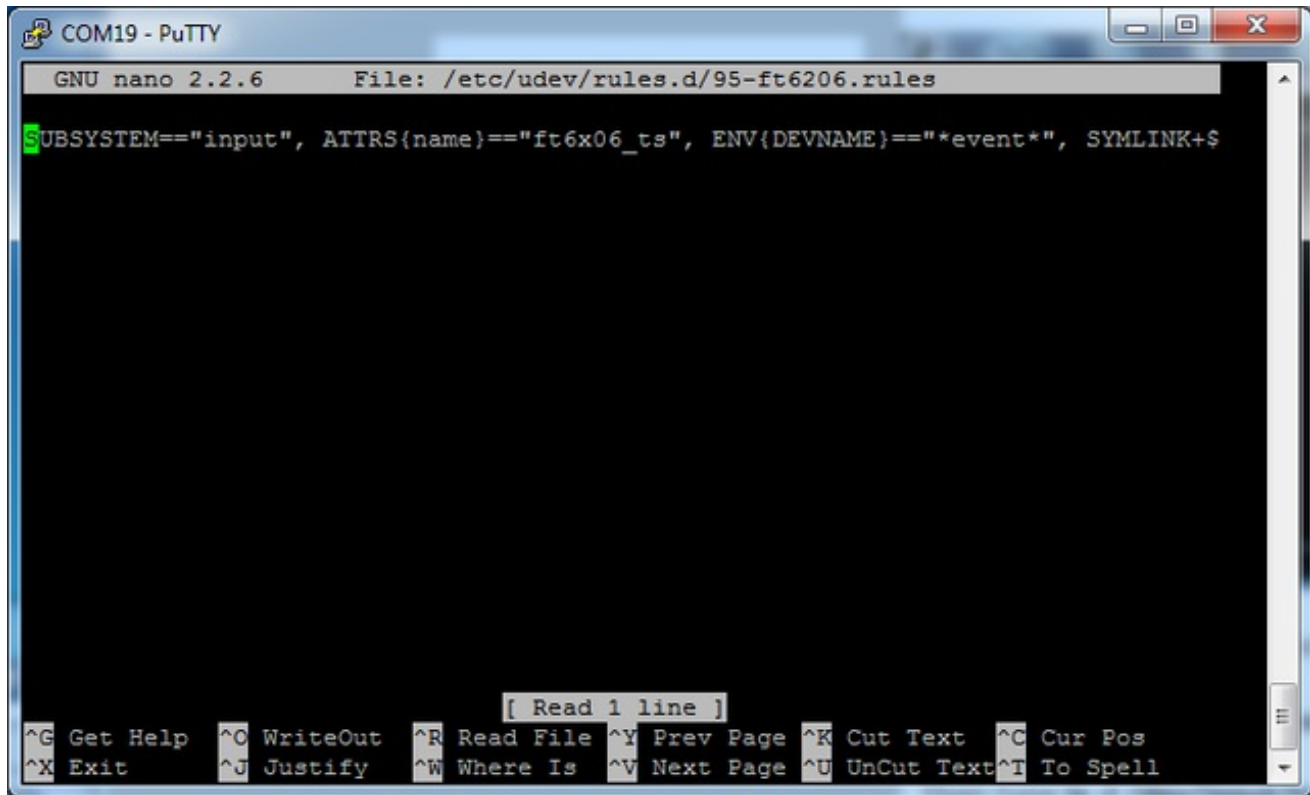
Before we start, we'll make a **udev** rule for the touchscreen. That's because the **eventX** name of the device will change a lot and its annoying to figure out what its called depending on whether you have a keyboard or other mouse installed.

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"
```



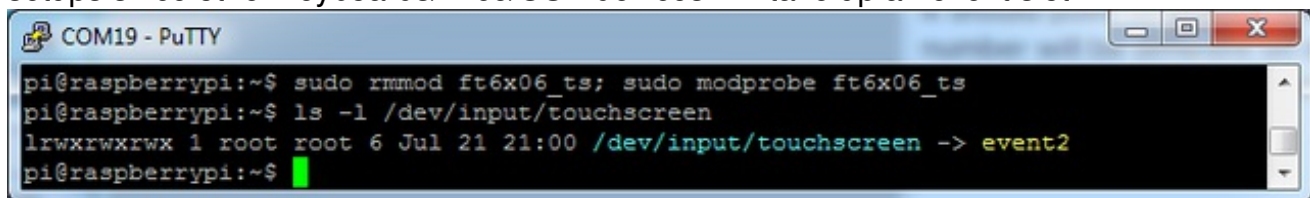
```
COM19 - PuTTY  
GNU nano 2.2.6 File: /etc/udev/rules.d/95-ft6206.rules  
SUBSYSTEM=="input", ATTRS{name}=="ft6x06_ts", ENV{DEVNAME}=="*event*", SYMLINK+$(  
[ Read 1 line ]  
^G Get Help ^O WriteOut ^R Read File ^Y Prev Page ^K Cut Text ^C Cur Pos  
^X Exit ^U Justify ^W Where Is ^V Next Page ^U UnCut Text ^T To Spell
```

Remove and re-install the touchscreen with

```
sudo rmmod ft6x06_ts; sudo modprobe ft6x06_ts
```

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



```
COM19 - PuTTY  
pi@raspberrypi:~$ sudo rmmod ft6x06_ts; sudo modprobe ft6x06_ts  
pi@raspberrypi:~$ ls -l /dev/input/touchscreen  
lrwxrwxrwx 1 root root 6 Jul 21 21:00 /dev/input/touchscreen -> event2  
pi@raspberrypi:~$
```

(<http://adafru.it/dlX>)

Event Testing

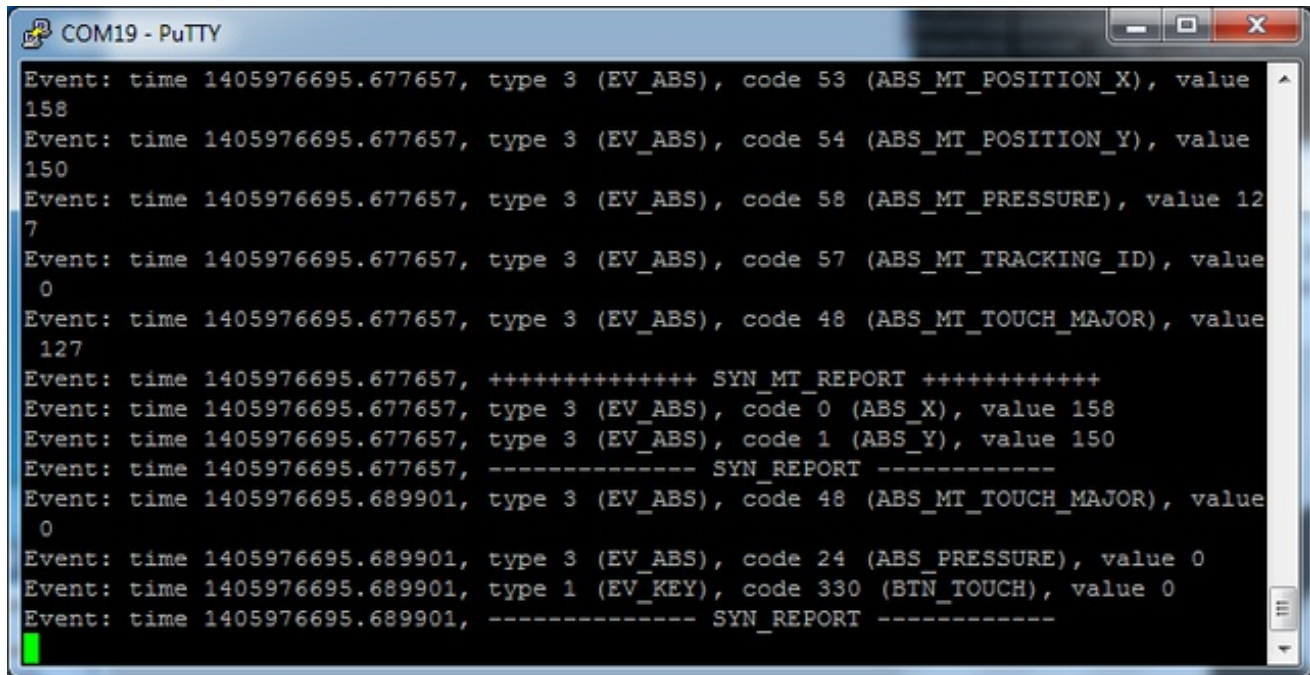
Even though capacitive touch screens don't require calibration, there are some useful tools we can use to debug the touchscreen. Install the "event test" and "touchscreen library" packages with

sudo apt-get install evtest tslib libts-bin

```
COM3 - PuTTY
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 127
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

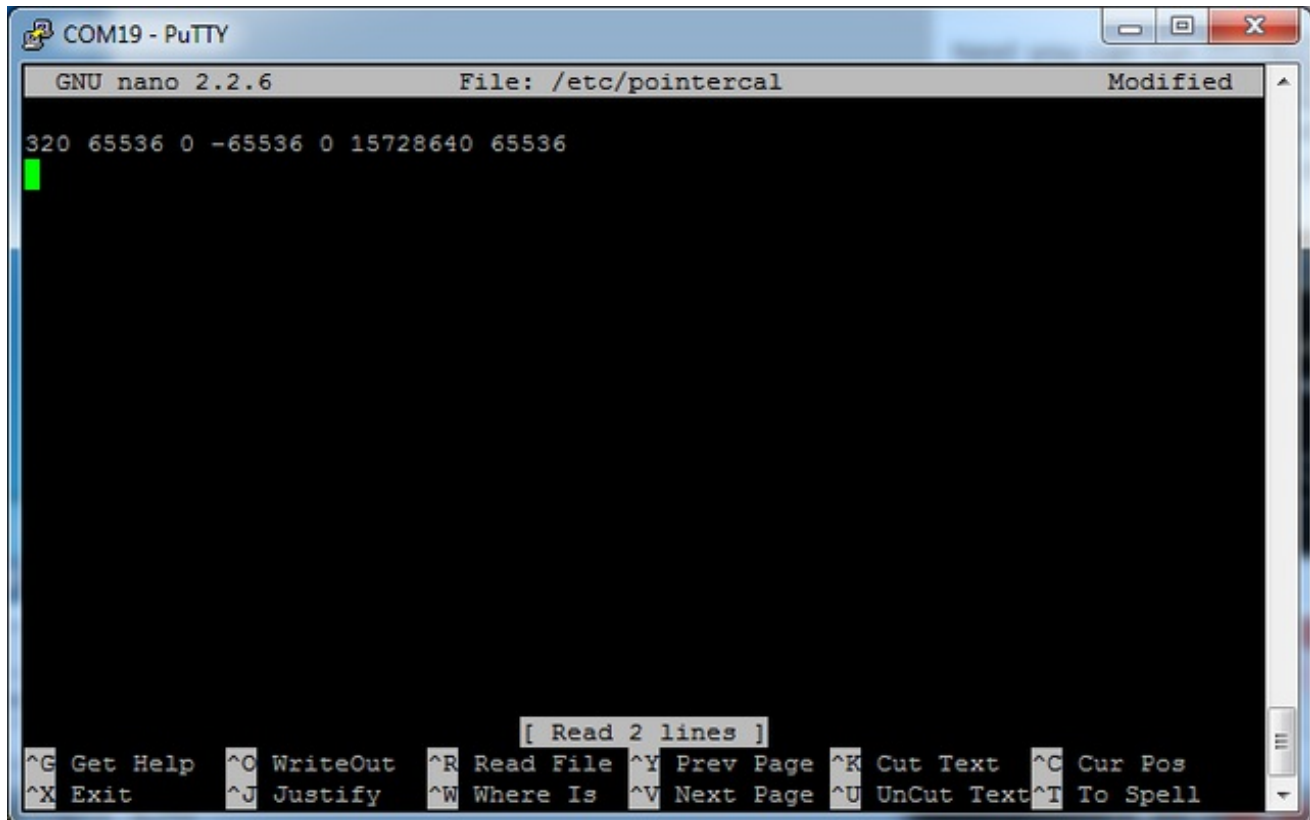
In order to use TSLIB - basically, the touchscreen without X11 - you'll need to set the calibration for TSLIB in `/etc/pointercal`

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
```

```
COM19 - PuTTY
GNU nano 2.2.6 File: /etc/pointerocal 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
```

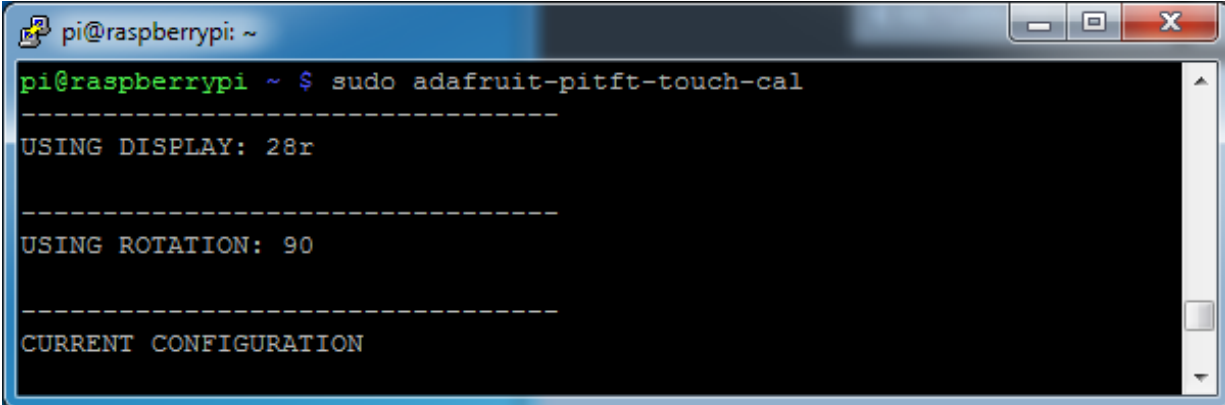
Calibration for other rotations

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 \(http://adafru.it/elu\)](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



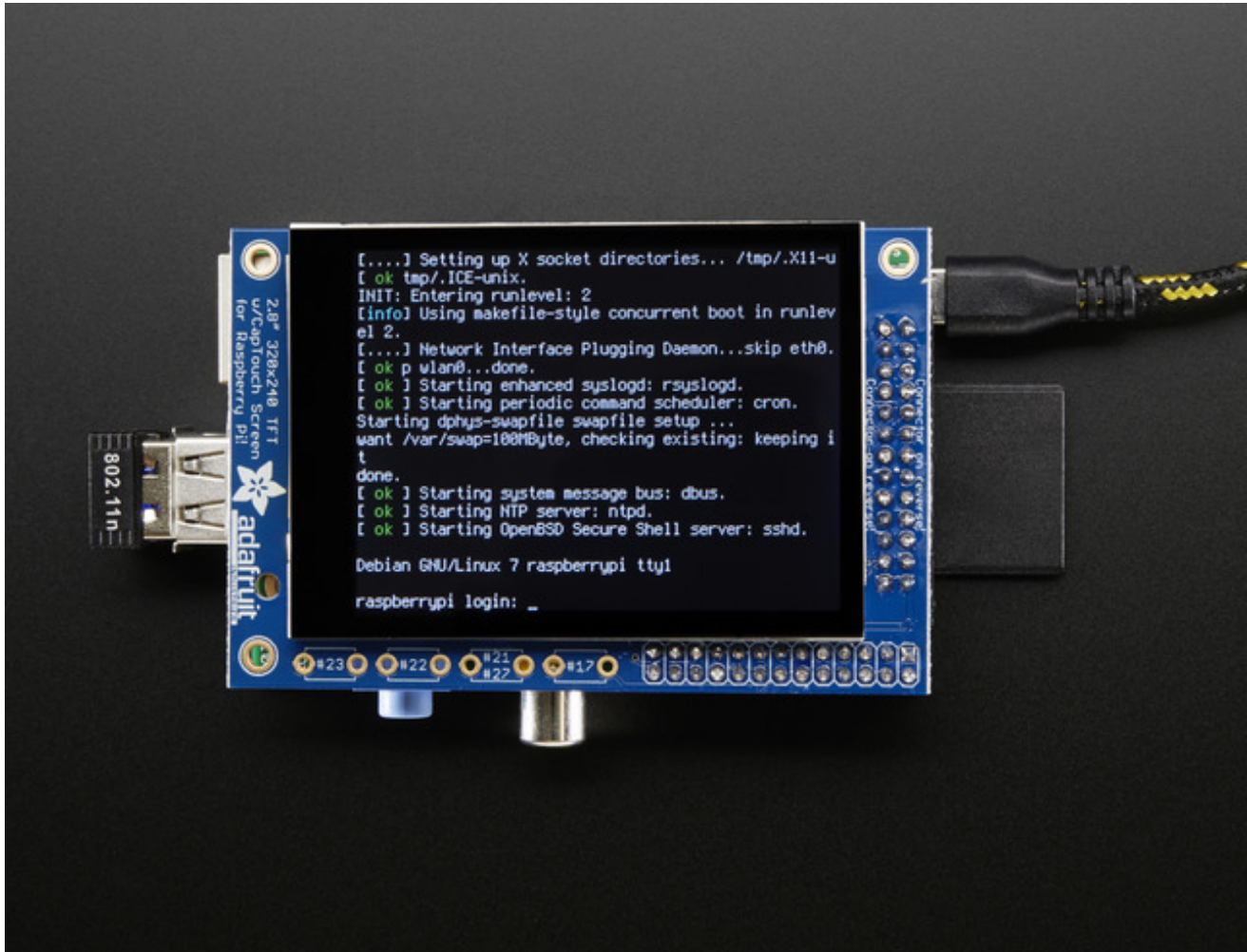
```
pi@raspberrypi: ~
pi@raspberrypi ~ $ sudo adafruit-pitft-touch-cal
-----
USING DISPLAY: 28r
-----
USING ROTATION: 90
-----
CURRENT CONFIGURATION
```

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 based 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.

```
pi@raspberrypi: ~  
-----  
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 ~ $ █
```

Using the Console

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



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

[Visit the Resistive 2.8" PiTFT tutorial on using the Console](#)

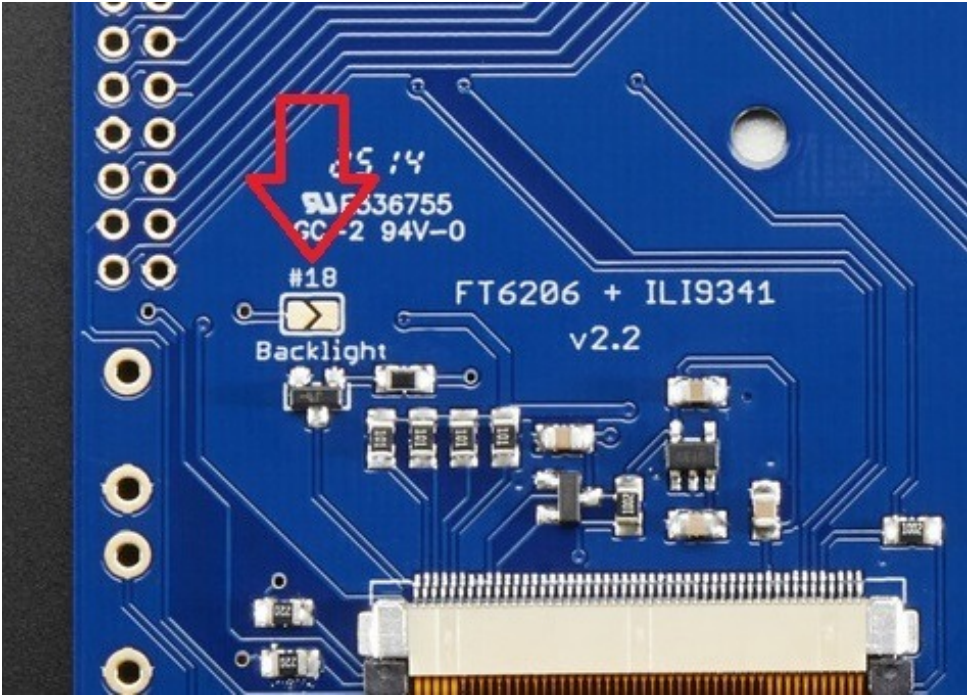
<http://adafru.it/dIZ>

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

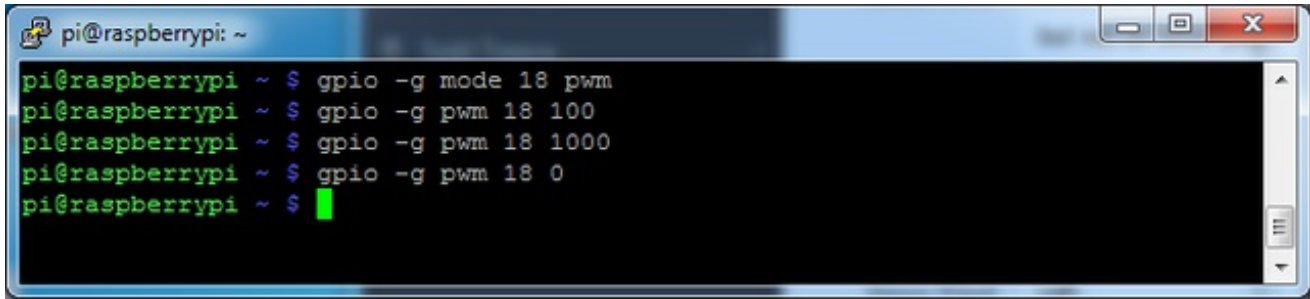
To enable using GPIO #18 as a backlight, solder closed the #18 backlight jumper on the PiTFT capacitive PCB!



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
```

A terminal window titled "pi@raspberrypi: ~" with standard window controls. The terminal shows a sequence of four commands: "gpio -g mode 18 pwm", "gpio -g pwm 18 100", "gpio -g pwm 18 1000", and "gpio -g pwm 18 0". The prompt "pi@raspberrypi ~ \$" is visible at the end of each line, with a green cursor on the final line.

```
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)!



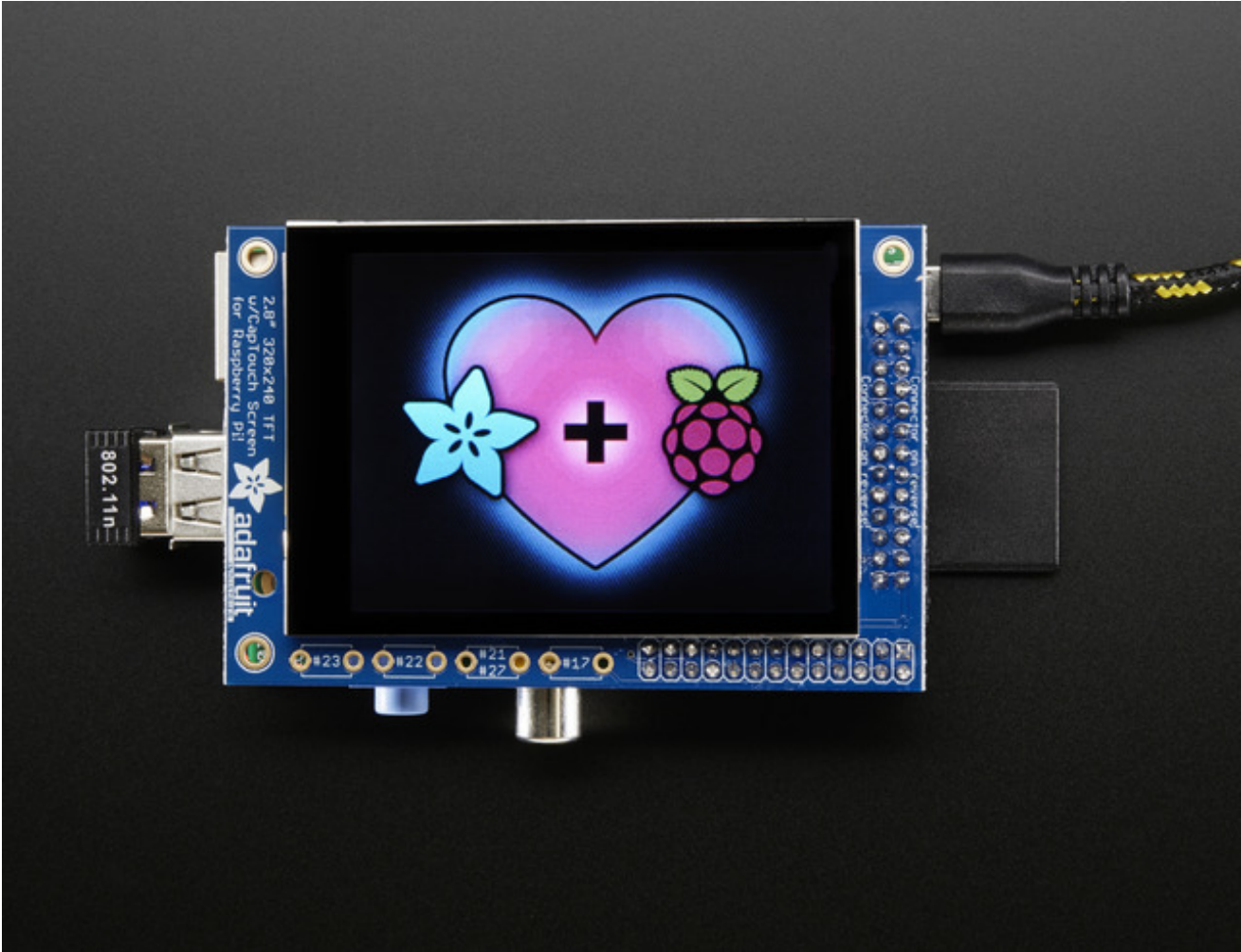
Playing Videos

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

[Visit the 2.8" resistive PiTFT video-playing tutorial](#)

<http://adafru.it/d1Y>

Displaying Images



This section is identical to the PiTFT Resistive 2.8" so please visit that page to display images on this Pi Plate

[Visit the 2.8" Resistive PiTFT page on displaying images](http://adafruit.com/blog/posts/2014-07-15-Displaying-Images-on-the-2.8-inch-PiTFT-Capacitive-Touch-Screen/)

<http://adafruit.com/dj0>

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 *fbc* (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)
<http://adafru.it/fbe>

Extras!

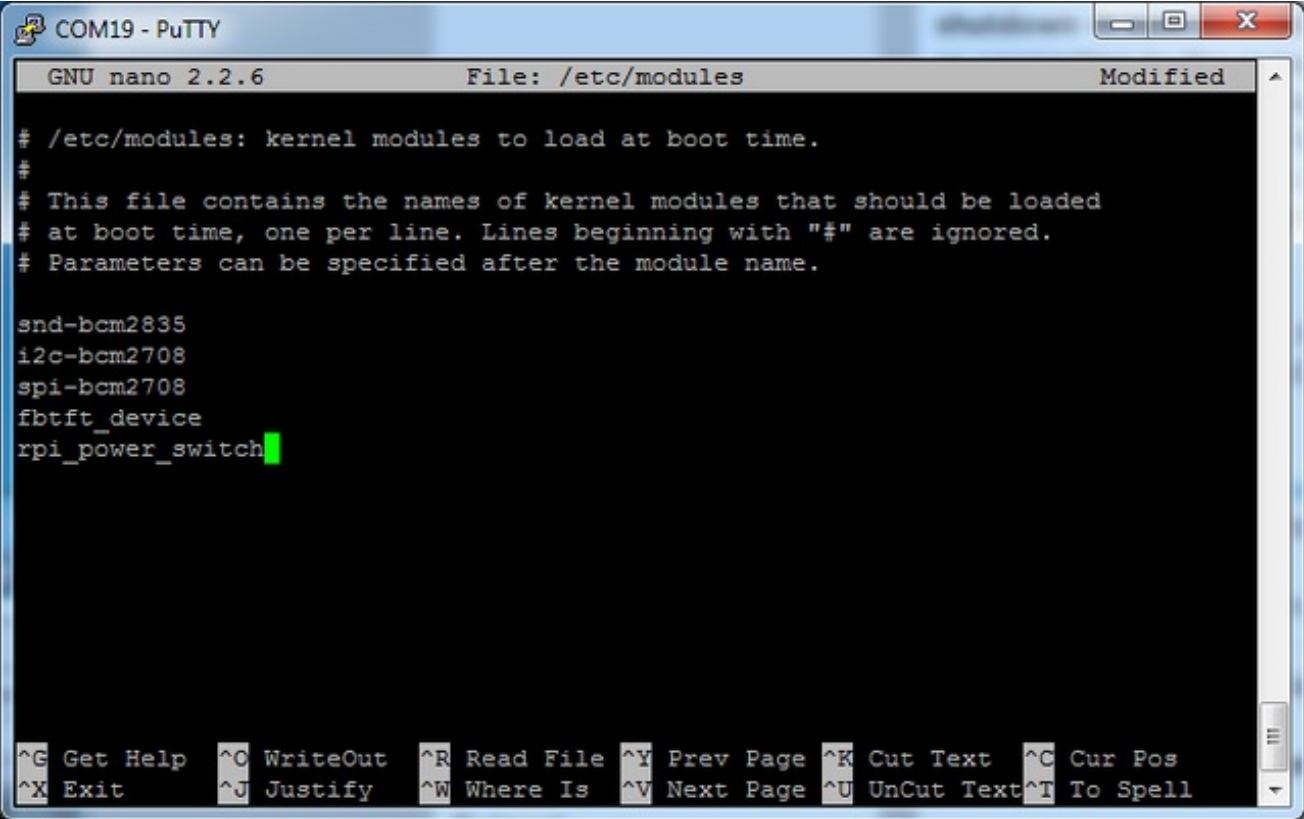
Tactile switch as power button

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. You can use any of them or other GPIO but #23's our favorite number anyways.

[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



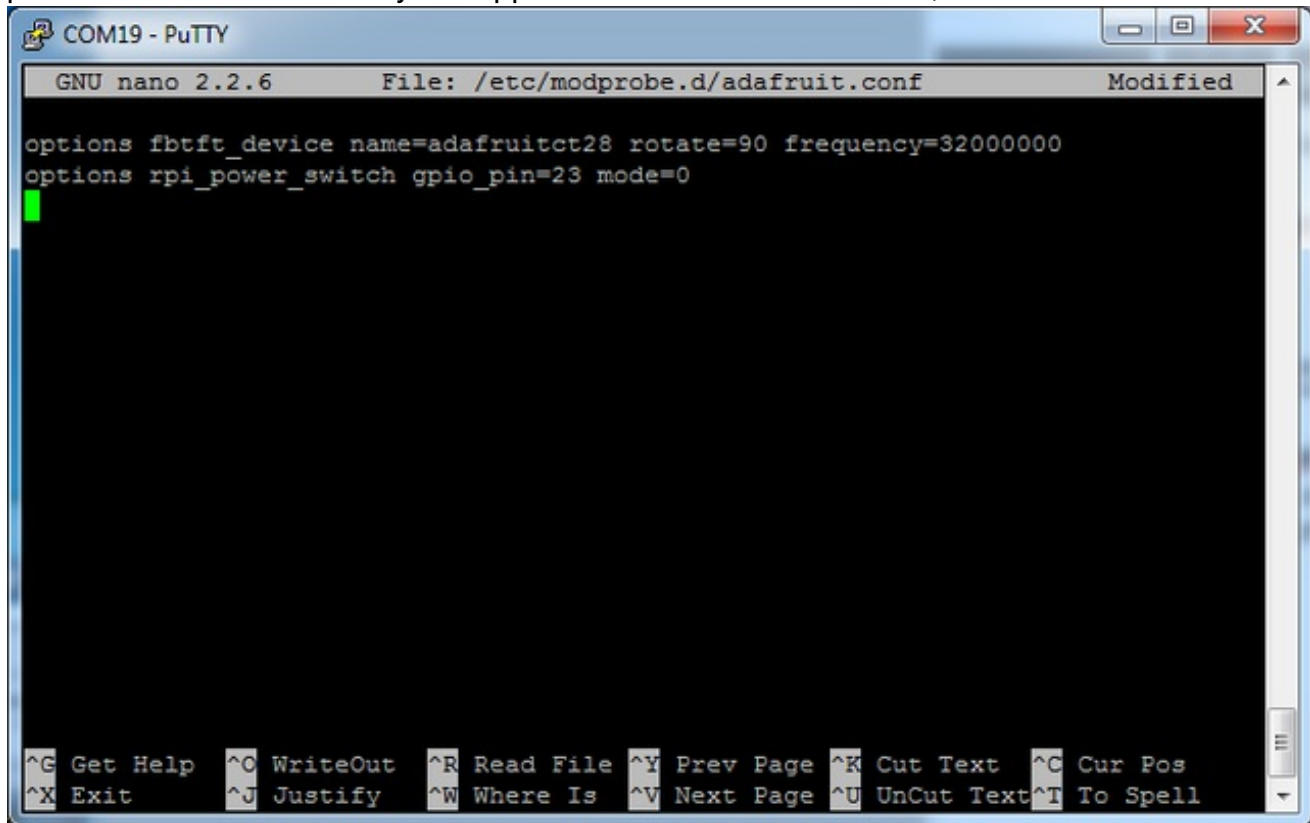
Now create a new conf file or edit our existing one with

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

and enter in the line

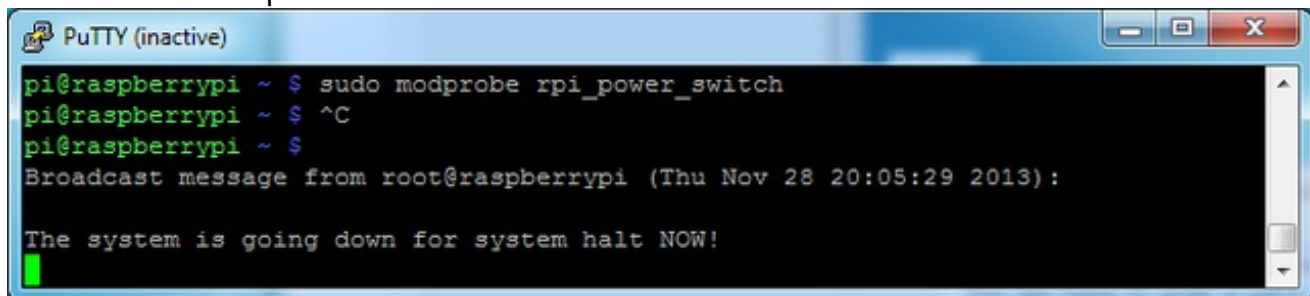
```
options rpi_power_switch gpio_pin=23 mode=0
```

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**



```
COM19 - PuTTY
GNU nano 2.2.6 File: /etc/modprobe.d/adafruit.conf Modified
options fbftft_device name=adafruitct28 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** and short the two pads of the GPIO #23 button



```
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

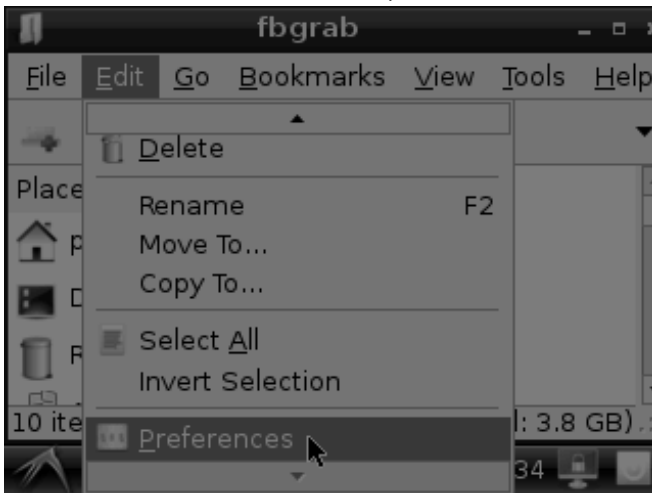
If you want to double-click on icons to launch something in X you may find it annoying to get it to work right. In LXDE you can simply set it up so that you only need to single click instead

of double.

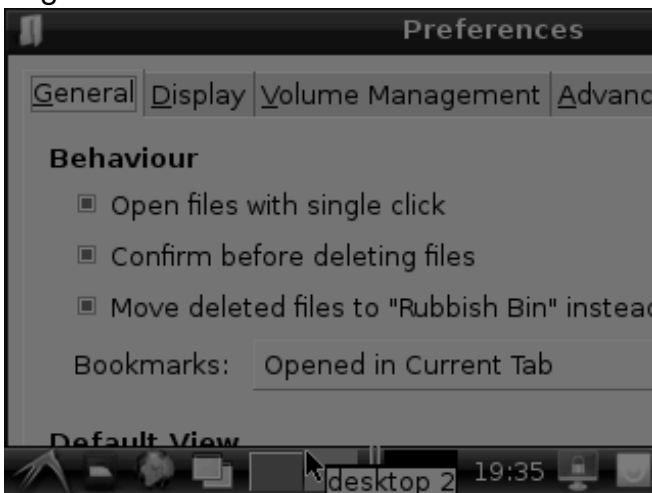
From LXDE launch the file manager (sorry these pix are grayscale, still figuring out how to screenshot the framebuffer!)



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)



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://adamaddin!) (<http://adafru.it/fH3>))



Gesture Input

This section is identical to the PiTFT Resistive 2.8" so please visit that page to try out gesture input on this Pi Plate

[Visit the 2.8" Resistive PiTFT Gesture Input page](http://adafru.it/dJ1)

<http://adafru.it/dJ1>

PiTFT Pygame Tips

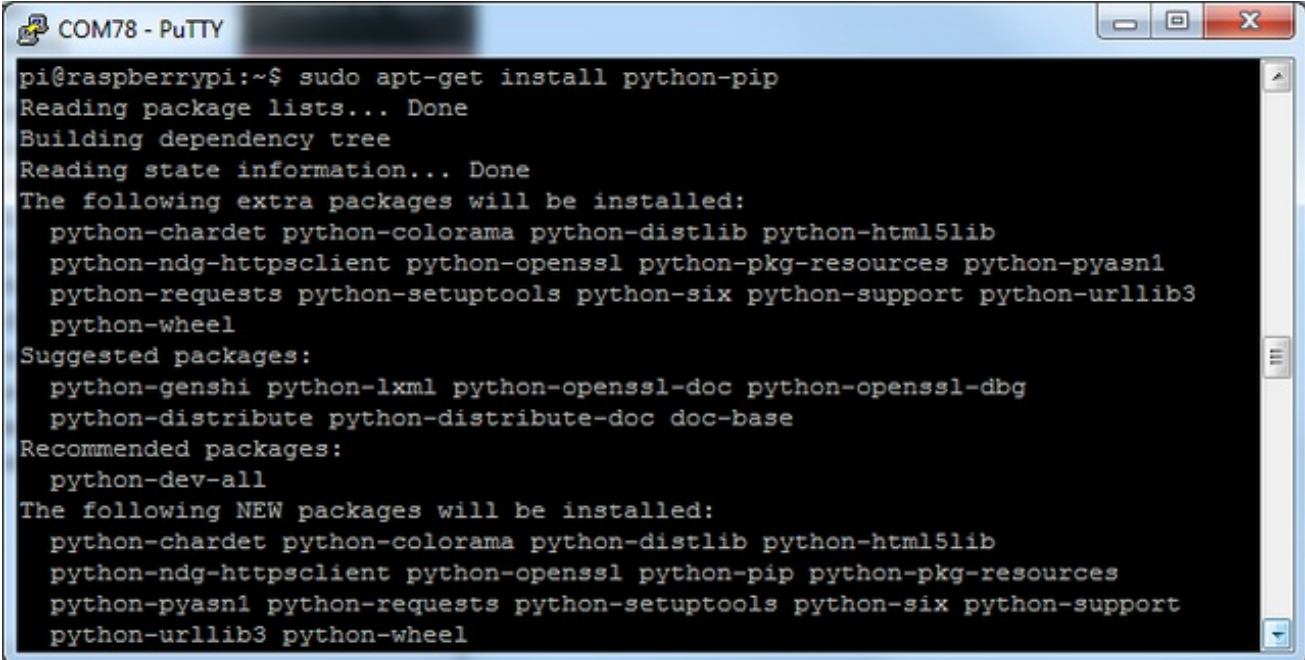
Since the PiTFT screen is fairly small, you may need to write custom UI programs. Pygame is the easiest way by far to do this.

[Jeremy Blythe has an excellent tutorial here on getting started.](http://adafru.it/kD2) (<http://adafru.it/kD2>)

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

Install pip & pygame

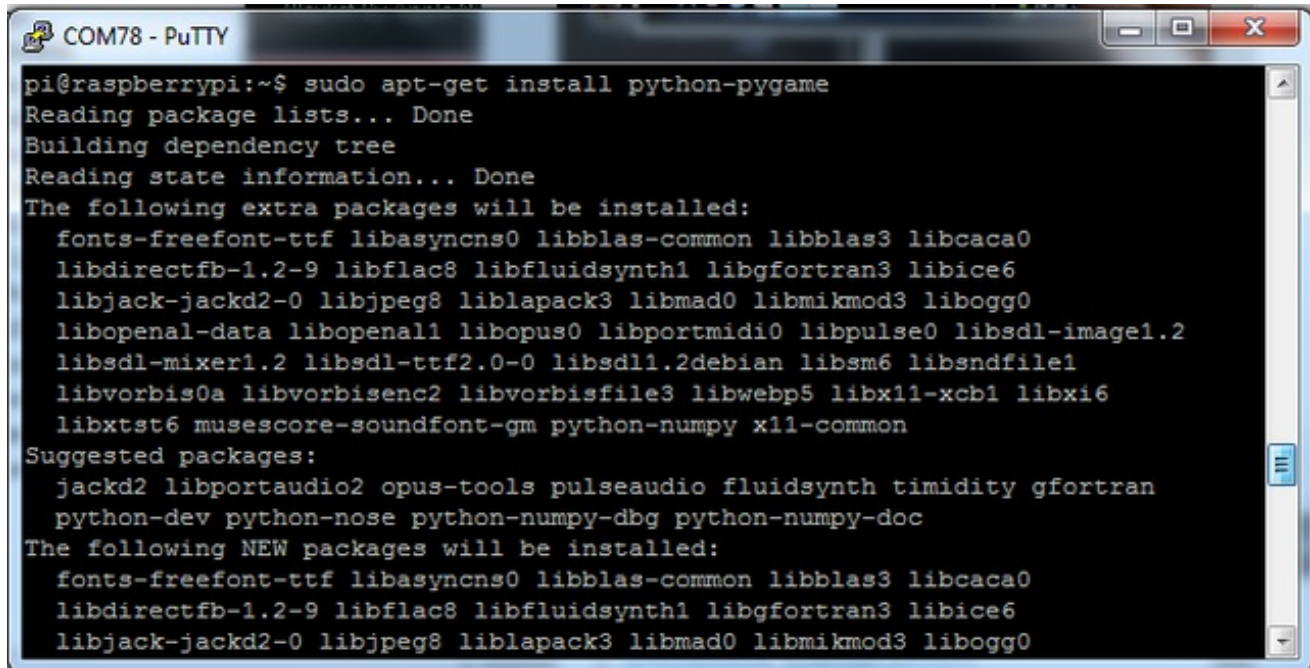
Install Pip: **sudo apt-get install python-pip**



```
COM78 - PuTTY
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-pyasnl
 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-pyasnl 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 libasynchns0 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 libasynchns0 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/fH3)(<http://adafru.it/fH3>))

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
```

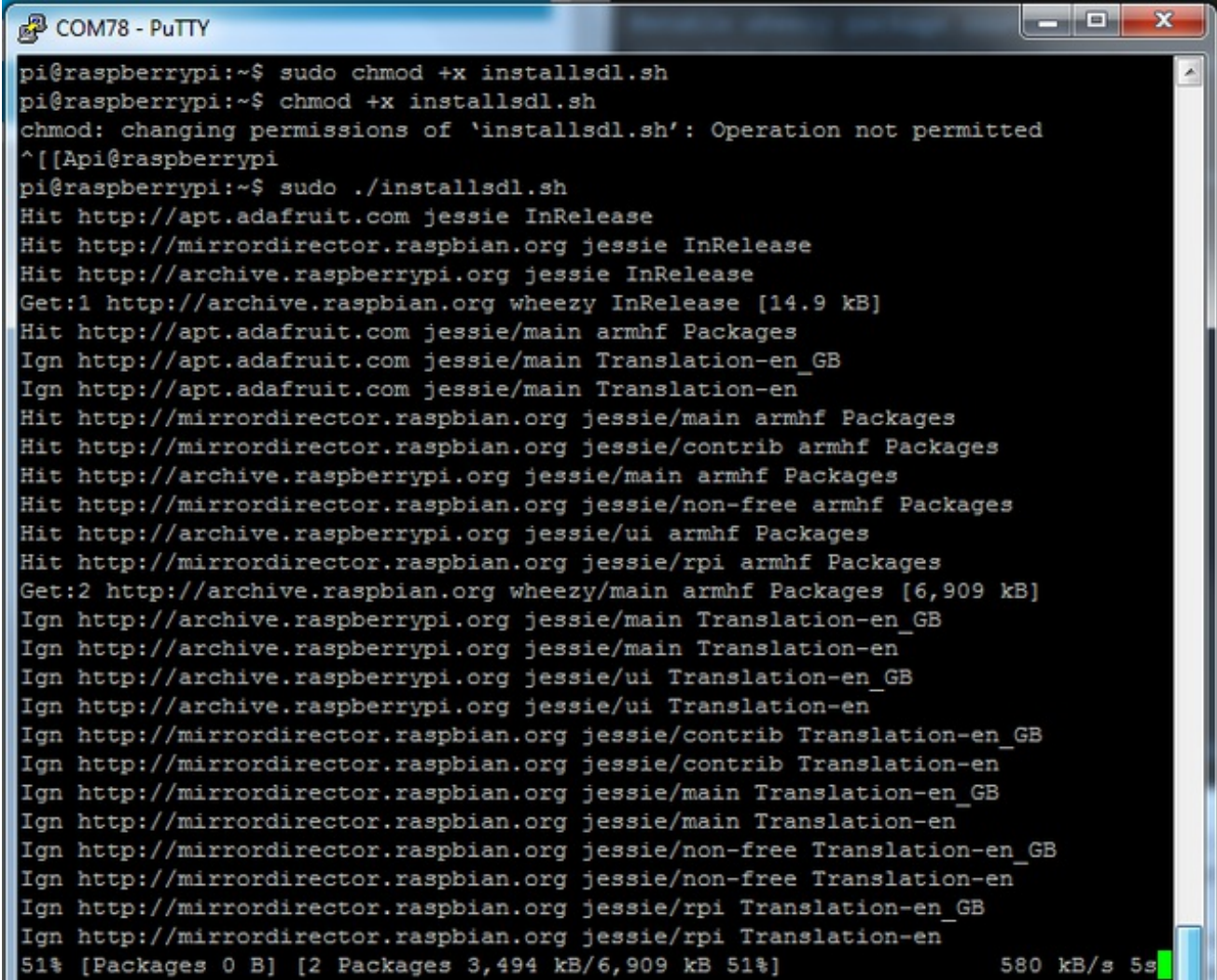


```
apt-get -y --force-yes install libsdl1.2debian/wheezy
```

run

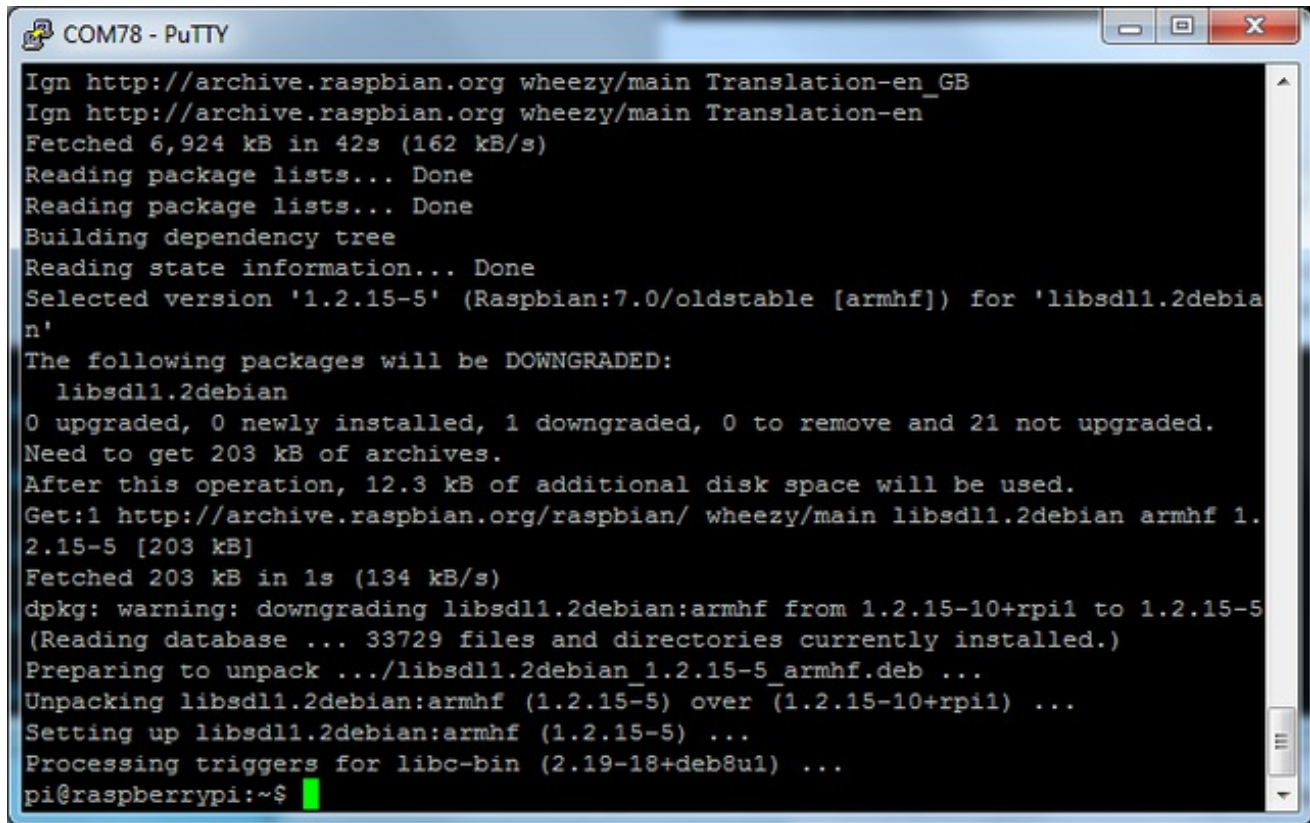
```
sudo chmod +x installsdl.sh
```

```
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 'libSDL1.2debian'
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+rp1 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+rp1) ...
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 doesn't

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) (<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) (<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)

<http://adafru.it/dJ2>

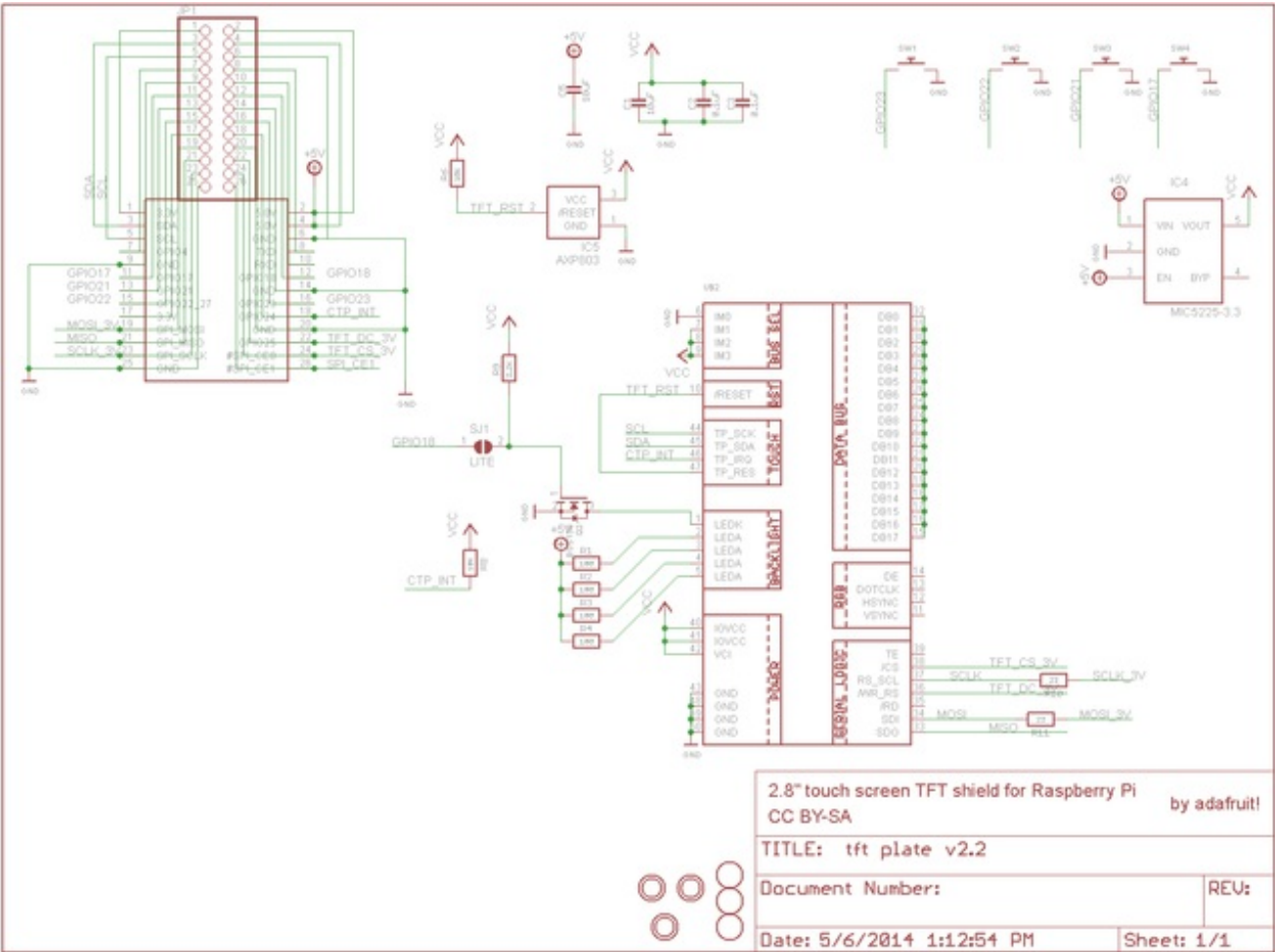


Downloads

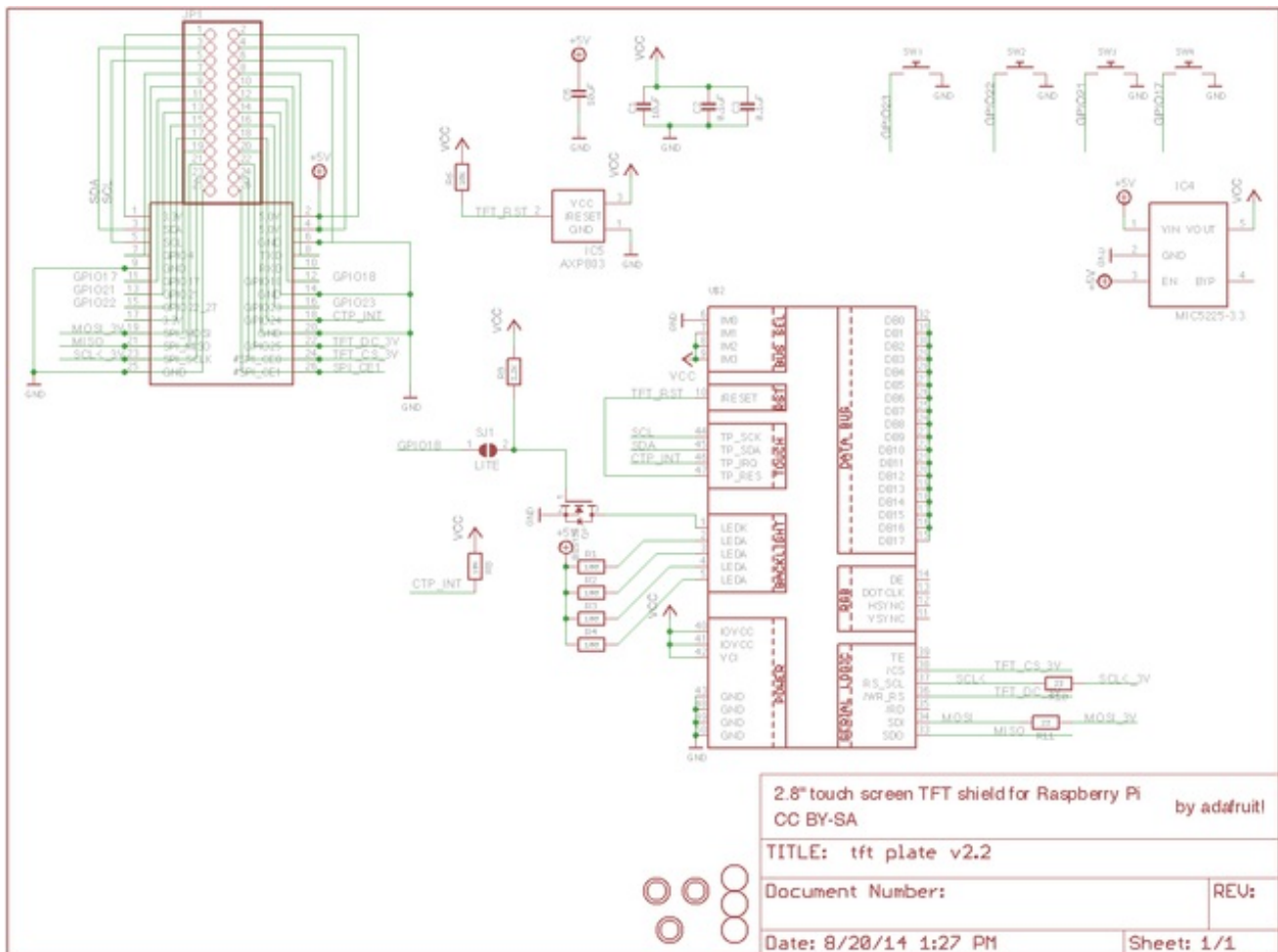
Files

- [The latest kernel fork that adds all the TFT, touchscreen, and other addons is here on github](http://adafru.it/aPa) (<http://adafru.it/aPa>)
- [Datasheet for the 'raw' 2.8" TFT display](http://adafru.it/d4m) (<http://adafru.it/d4m>)
- [FT6206 Datasheet](http://adafru.it/dRm) (<http://adafru.it/dRm>) & [App note](http://adafru.it/dRn) (<http://adafru.it/dRn>) (capacitive chip)
- [EagleCAD PCB files on GitHub](http://adafru.it/oYC) (<http://adafru.it/oYC>)

Schematic for Pi 1 Version

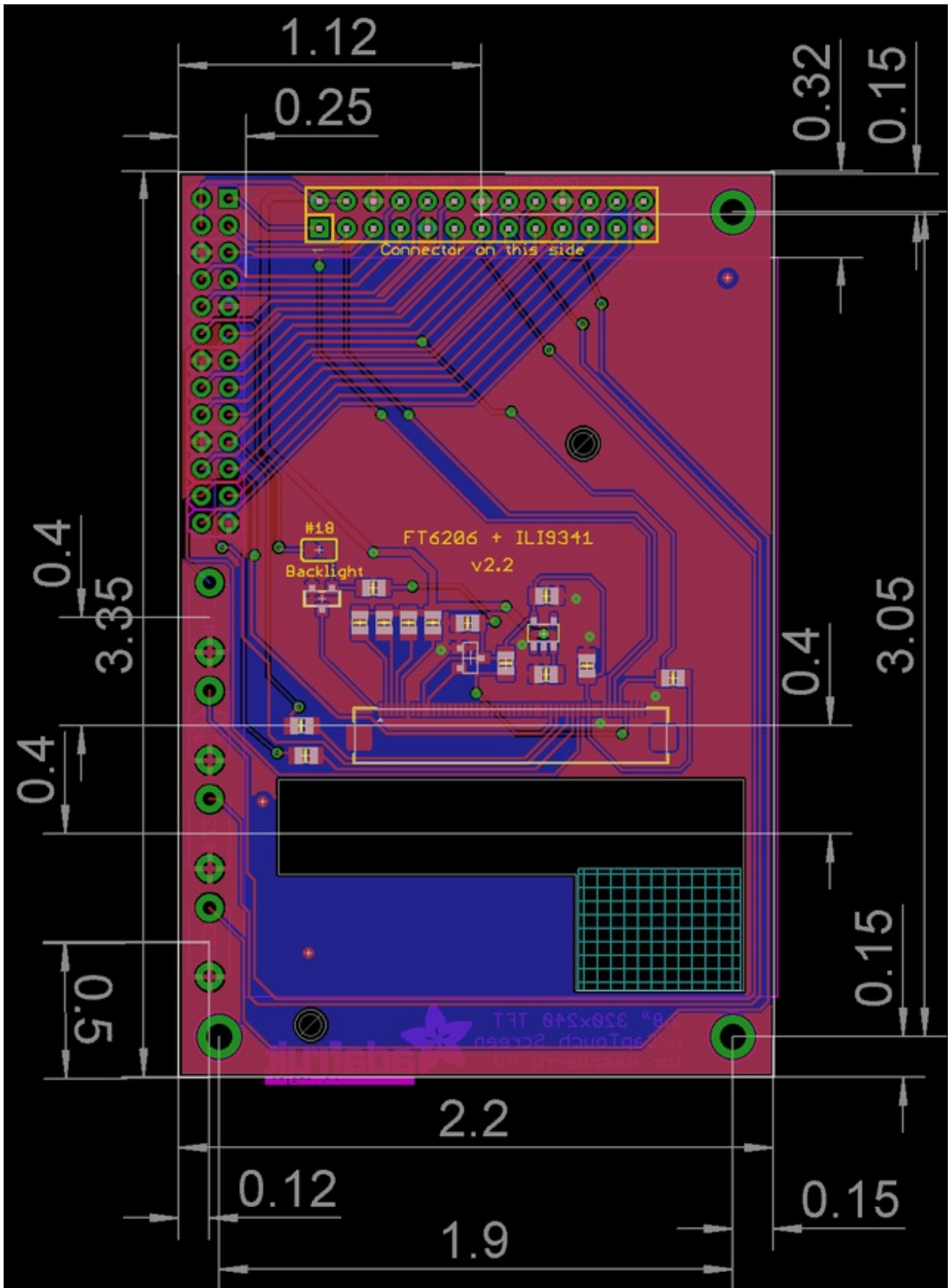


Schematic for PiTFT Plus (B+/Pi 2 shape)



Fabrication Print (Pi 1 Version)

Dimensions in Inches



Fabrication Print (B+/Pi 2 Version)

Dimensions in mm

