Cloudy Robotics Installation Guide

# 1. Parts

## 1.1 Required Parts

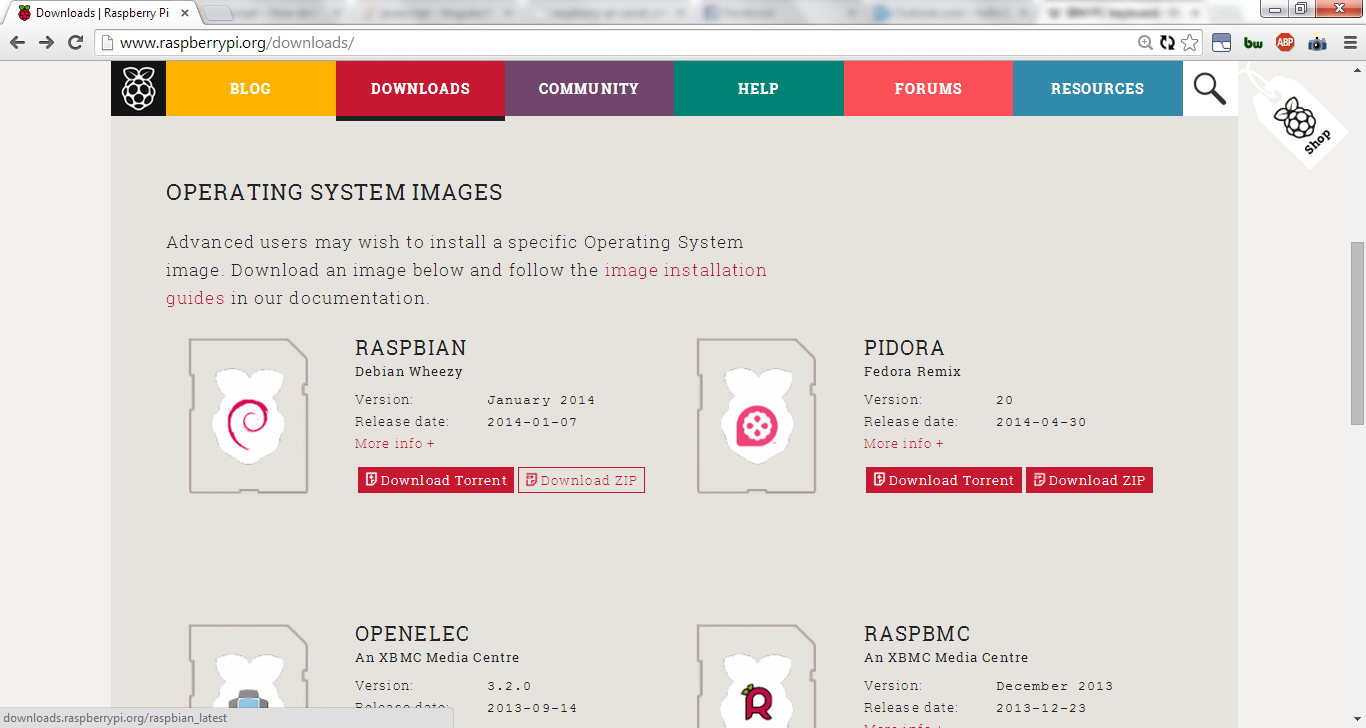
* Raspberry Pi (<http://www.adafruit.com/products/998>)
* 8GB or Larger SD Card
* Micro USB Cable
* LCD Monitor with HDMI Port
* HDMI Cable
* Keyboard
* Wireless Adapter (Look for a list of supported wireless adapters at <http://elinux.org/RPi_USB_Wi-Fi_Adapters>, make sure they don’t require a USB hub)
* Raspberry Pi Camera ([http://www.adafruit.com/products/1367)](http://www.adafruit.com/products/1367)%20) or Logitech C920 Webcam (<http://www.newegg.ca/Product/Product.aspx?Item=N82E16826104635>), the Raspberry Pi camera is cheaper and much easier to set up as it’s low power and doesn’t require a separate USB hub to power it, the C920 is faster but requires you to power a separate 5V USB hub on the robot
* 4 Male to Female Jumper Wires (<http://www.adafruit.com/products/824>)
* VEX Cortex
* USB A-to-A Cable to program Cortex from computer

## 1.2 Optional Parts

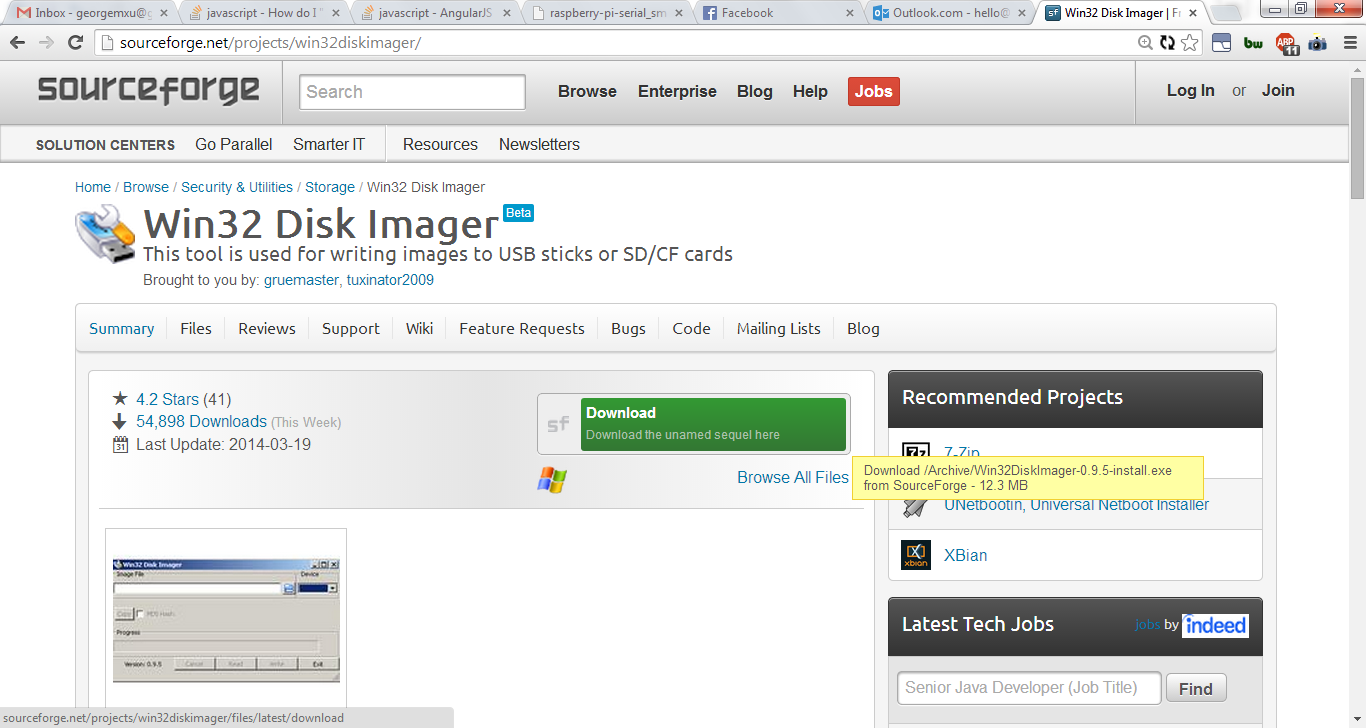
* USB Hub (Look for a list of supported USB hubs at <http://elinux.org/RPi_Powered_USB_Hubs>), only used for making setting up the wireless much easier, if you’re using a separate webcam instead of the Raspberry Pi camera then you’ll need a USB hub

# 2. Setting up the Raspberry Pi

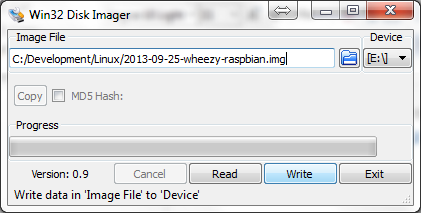
## 2.1 Setting up the OS



Download the Raspbianimage from<http://www.raspberrypi.org/downloads/>. Extract the image from the Zip file somewhere.



Download and install Win32 Disk Imager from <http://sourceforge.net/projects/win32diskimager/>.



Insert an 8 GB or larger SD card into your computer. Select whichever drive the SD card is in under **Device** and under **Image File**, navigate to where the Raspbian image is. Select **Write**, to start writing the image to the SD card.

The writing process will take about 10 minutes, once it’s completed, you can remove the SD card from the computer.

## 2.2 Connecting Everything

Insert the SD card to the Pi. Connect a USB hub to the Pi if you have one, it will make setting up the wireless a lot easier.

### 2.2.1 With USB Hub

Connect the following to the Pi if you have a USB hub:

1. Keyboard via USB
2. Wireless Adapter via USB
3. LCD Monitor via HDMI
4. Power Cable via Micro USB
5. Raspberry Pi Camera or Webcam
6. Mouse via USB

### 2.2.2 Without USB Hub

Connect the following to the Pi if you don’t have a USB hub:

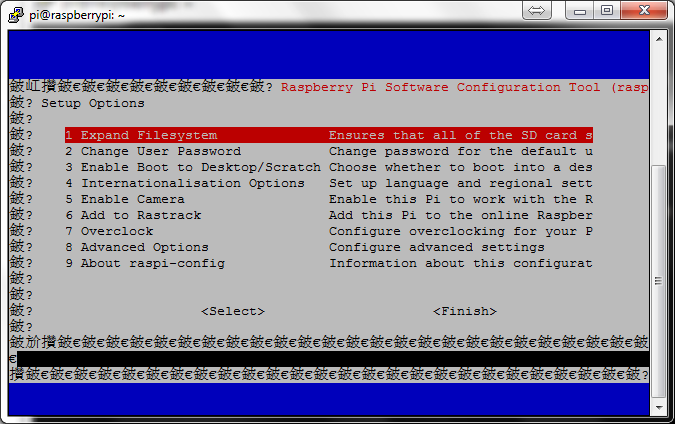
1. Keyboard via USB
2. Wireless Adapter via USB
3. LCD Monitor via HDMI
4. Power cable via Micro USB
5. Raspberry Pi Camera

The Pi starts once it receives power.

## 2.3 Initial Configuration



Login with username **pi** and password **raspberry**.



When the Pi boots, this screen will show up. Otherwise enter the following command to bring up this screen:

*sudo raspi-config*

Under the blue screen, select the following options:

1. **Expand Filesystem**
2. **Enable Camera** -> **Enable**
3. **Internationalisation Options** -> **Change Keyboard Layout** -> **Generic 104-key PC** -> **English (US)** -> **English (US)** -> **The default for the keyboard layout** -> **No compose key** -> **No**
4. Finish and reboot

## 2.4 Connecting to Wireless

If you have a USB hub, you can use the mouse, keyboard and the wireless adapter, making it much easier setup the wireless since you can do it via the graphical interface.

### 2.4.1 With USB hub and mouse

Enter the following command from the Pi to enter the graphical interface:

*startx*

Open the **WiFi Config** icon on the desktop.

Click **Scan** to bring up a list of local wireless networks.

Select your WiFi network to bring up another screen.

1. If authentication is **WPA-Personal** or **WPA2-Personal** then enter the password in **PSK**.
2. If authentication is **no authentication**, no password is required.
3. If authentication is something else, then talk to the network administrator for details.

Ignore the rest of the fields if you followed either 1 or 2 above, and **Add** the network.

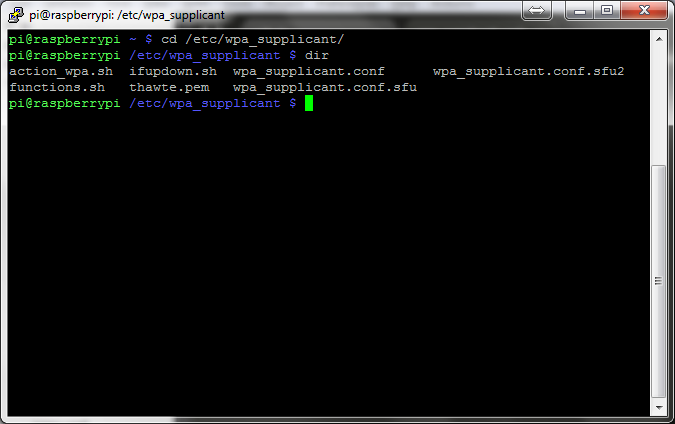
Select the network you added in the dropdown box and click **Connect** to connect to it.

Refer to the guides below for more info on connecting to the wireless through the GUI:

<http://thepihut.com/pages/how-to-setup-wifi-on-your-raspberry-pi-raspbian>

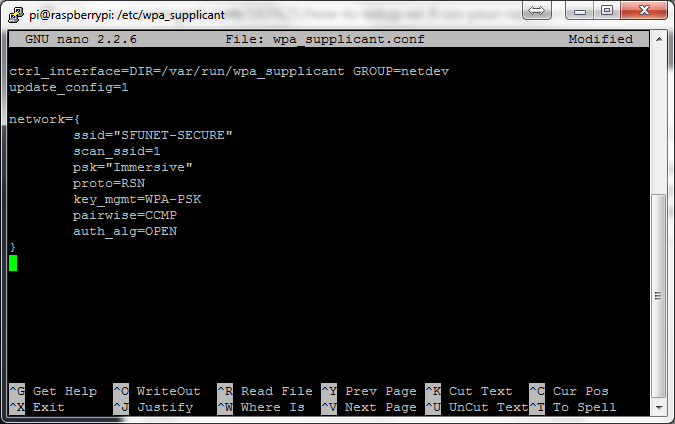
<https://learn.adafruit.com/adafruits-raspberry-pi-lesson-3-network-setup/setting-up-wifi-with-raspbian>

### 2.4.2 Without USB hub and mouse



Enter the following command to navigate to the wireless config folder:

*cd /etc/wpa\_supplicant/*



Enter the following command to edit the wireless config file:

*sudo nano wpa\_supplicant.conf*

Edit the config file according to the following rules, refer to image above for an example:

ctrl\_interface=DIR=/var/run/wpa\_supplicant GROUP=netdev  
update\_config=1

network={

ssid="YOURSSID"

scan\_ssid=1

psk="YOURPASSWORD"

# Protocol type can be: RSN (for WP2) and WPA (for WPA1)  
proto=WPA

# Key management type can be: WPA-PSK or WPA-EAP (Pre-Shared or Enterprise)  
key\_mgmt=WPA-PSK

# Pairwise can be CCMP or TKIP (for WPA2 or WPA1)  
pairwise=TKIP

#Authorization option should be OPEN for both WPA1/WPA2 (in less commonly used are SHARED and LEAP)  
auth\_alg=OPEN

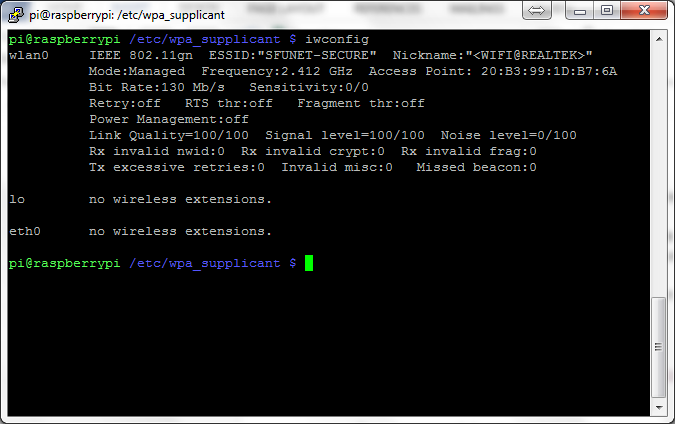
}

Under **ssid**, input your wireless connection name, under **psk**, input your password.

Refer to the guides below for more info on connecting to the wireless through command line:

<http://www.howtogeek.com/167425/how-to-setup-wi-fi-on-your-raspberry-pi-via-the-command-line/>

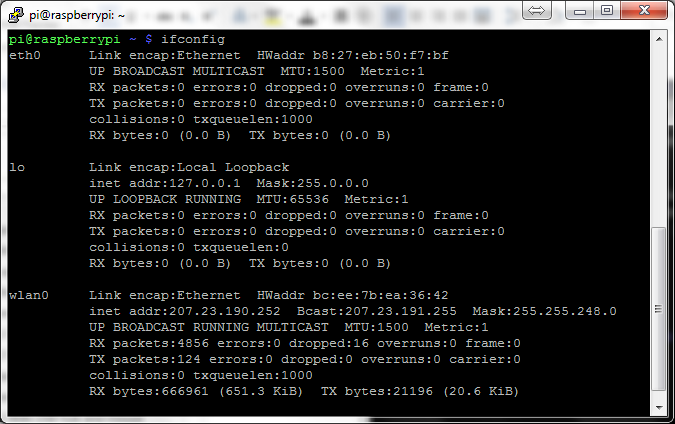
## 2.5 Verifying the Pi has Wireless



If you’ve successfully set up wireless you can type the following command and it’ll show the name of your wireless network:

*iwconfig*

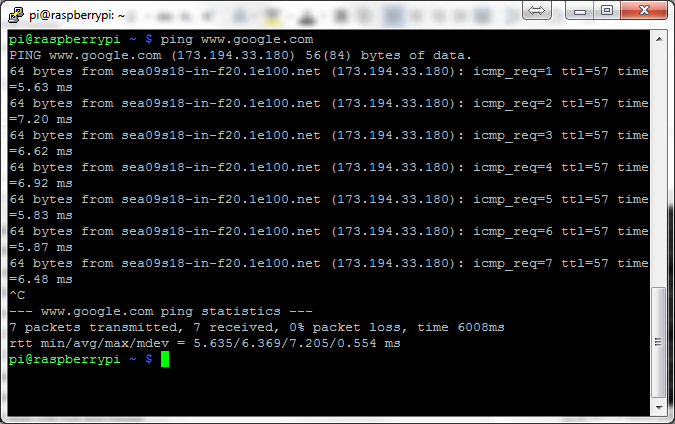
Next to **ESSID** under **wlan0** you can see the name of your wireless network. In my case it’s SFUNET-SECURE.



To see the Pi’s IP address enter the following command:

*ifconfig*

Look at **inet addr** under **wlan0**, to see your Pi’s address. In my case it’s 207.23.190.252. This will be used later.



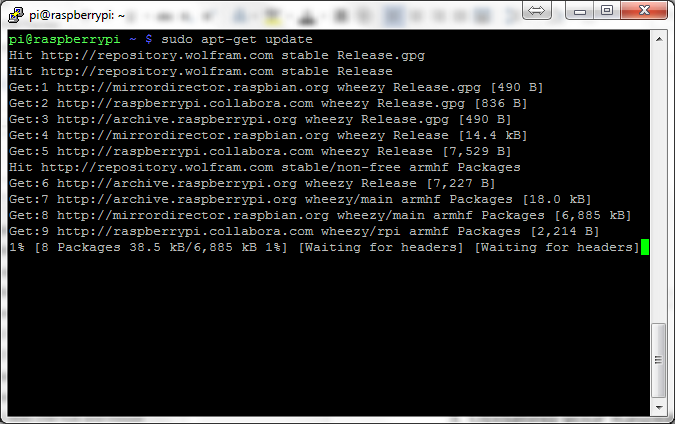
Trying pinging a server to see if your ping is heard and responded with this command:

*ping www.google.com*

If you get responses back like those in the image above, then your wireless network is connected to the outside internet. You can stop the pinging by pressing **CTRL+C**.

# 3. Updating and Installing Software

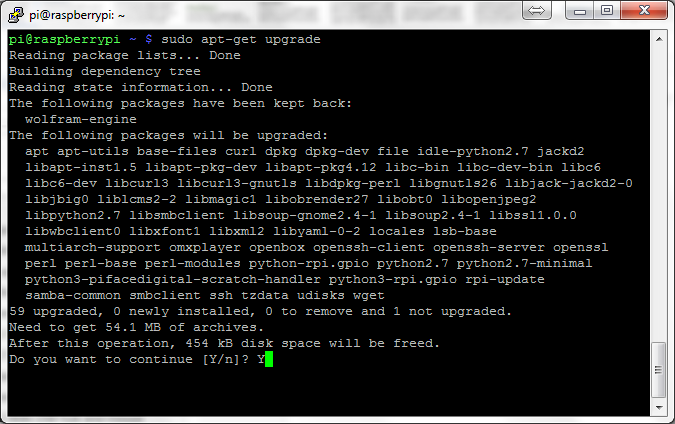
## 3.1 Updating the Pi’s OS



Type the following command to update your Pi’s OS and software to the latest available:

*sudo apt-get update*

If the installation prompts you for any confirmation, input **Y** to the command line. The process takes about 5 minutes.



Type the following command to update your Pi’s firmware to the latest available:

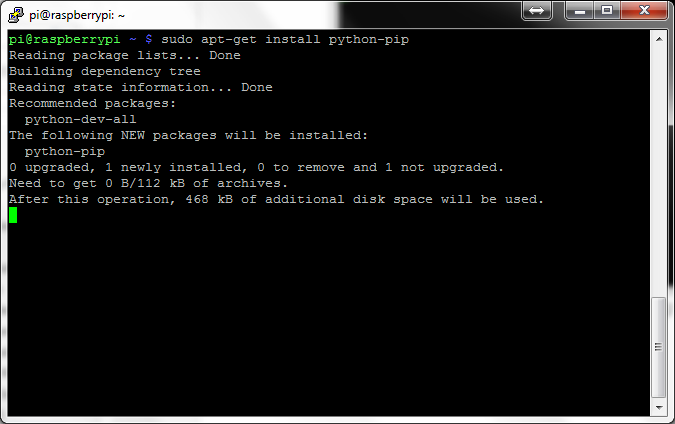
*sudo apt-get upgrade*

If the installation prompts you for any confirmation, input **Y** to the command line. The process takes about 15 minutes and may ask you to reboot the Pi.

Reboot the Pi by following the on-screen prompts or entering the following command if you weren’t prompted earlier:

*sudo reboot*

## 3.2 Installing Python Packages



Install the Python package installer pip by entering the following command:

*sudo apt-get install python-pip*

Confirm the installation prompt by entering **Y** to the command line.

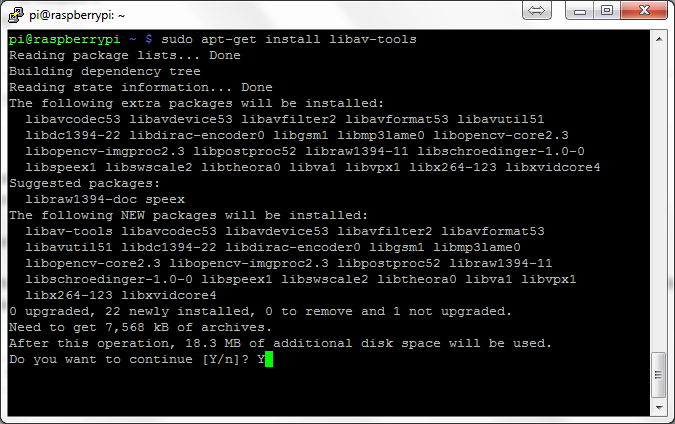
Install the Pyserial Python package by entering the following command:

*sudo pip install pyserial*

Install the SocketIO Client Python package by entering the following command:

*sudo pip install socketio-client*

## 3.3 Installing libav for Streaming



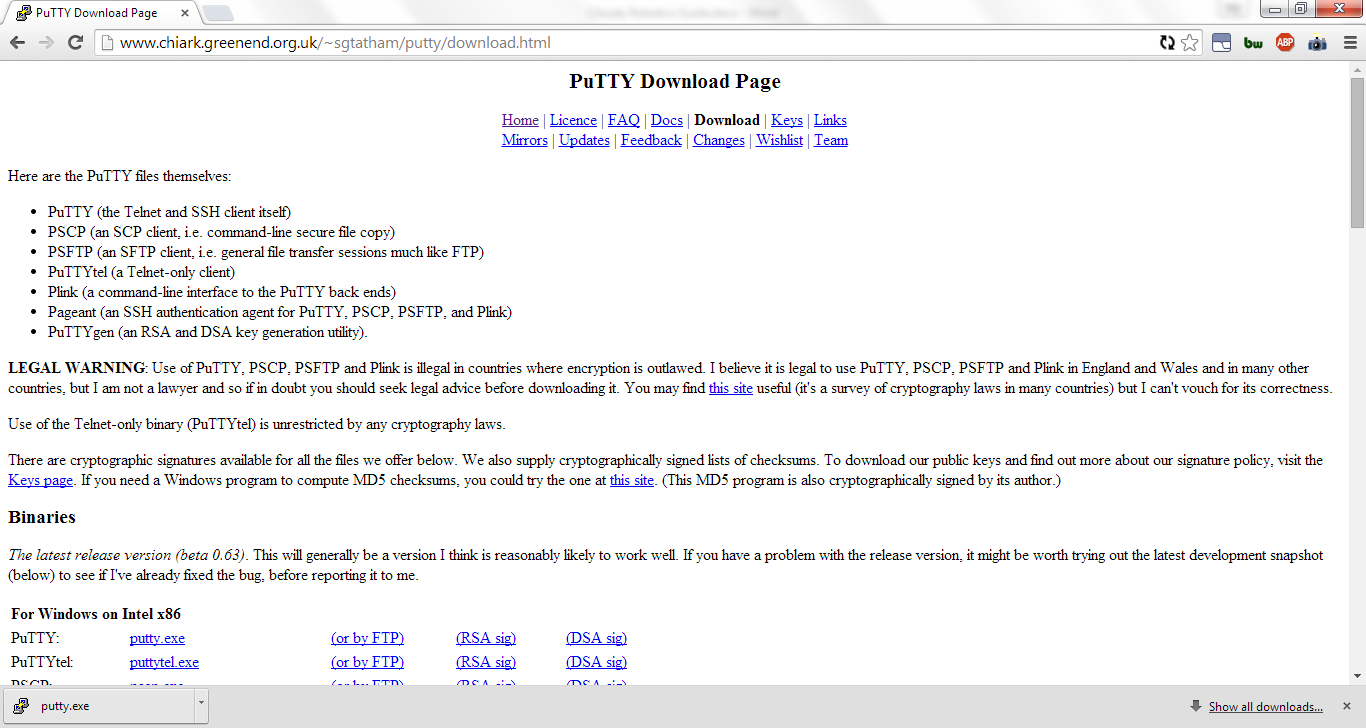
Install libav for streaming the Raspberry Pi Camera by entering the following command:

*sudo apt-get install libav-tools*

This will only be used if you’re using the Pi Camera which is recommended as it’s much simpler to set up compared to a dedicated webcam like the Logitech C920.

# 4. Additional Software to Use with Pi

## 4.1 Download PuTTY



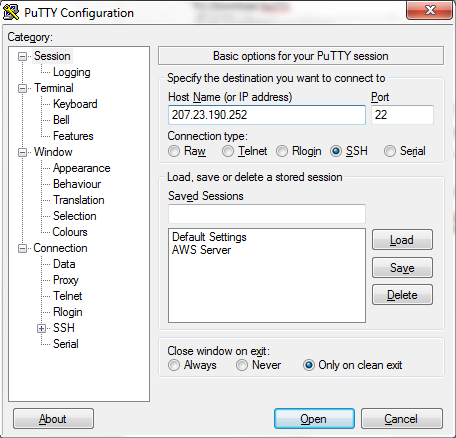
Download the PuTTY executable from<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>.

PuTTY can be used to SSH into your Raspberry Pi and you will be able to remotely use command line from somewhere else as long as the Pi is on and connected to the wireless.

If the wireless network you are on or the firewall has not opened port 22, you will not be able to SSH into it from outside your internet network. You will have to connect to the same wireless or internet network to SSH into it.

If you have opened port 22, then you will be able to SSH into to the Pi from anywhere in the internet.

We will use PuTTY to run scripts for controlling the robot and remotely program the Raspberry Pi later.

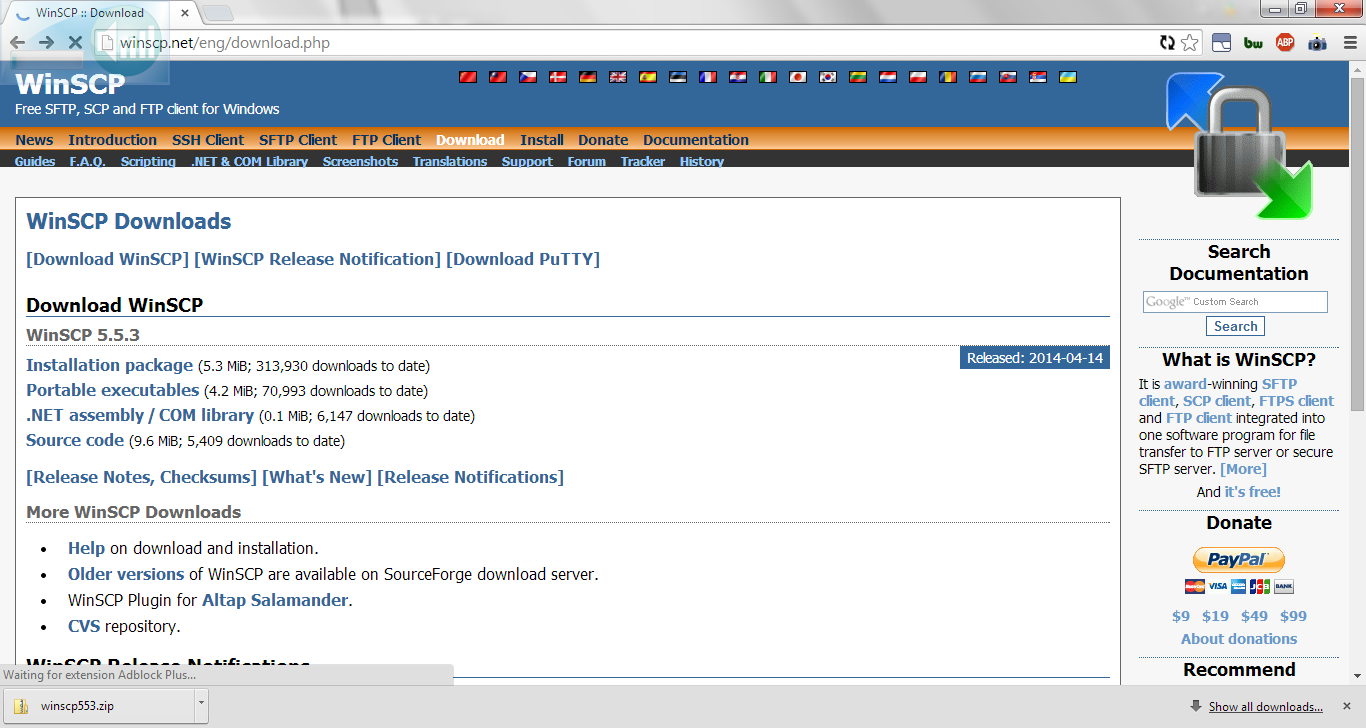


Type in the IP of the Pi you found earlier and click **Open** to connect to the Pi.



It will ask you to login. Use **pi** as username and **raspberry** as password.

## 4.2 Download WinSCP



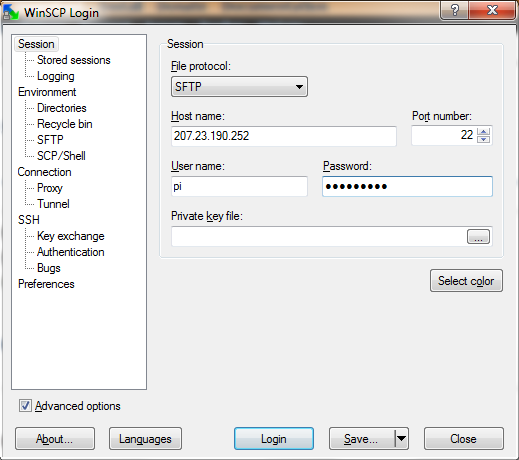
Download and install WinSCP from <http://winscp.net/eng/download.php>. The portable executable you can run directly, the installation package you will to install first.

WinSCP is a FTP client that also goes over port 22. You can use it to upload files and scripts to your Pi.

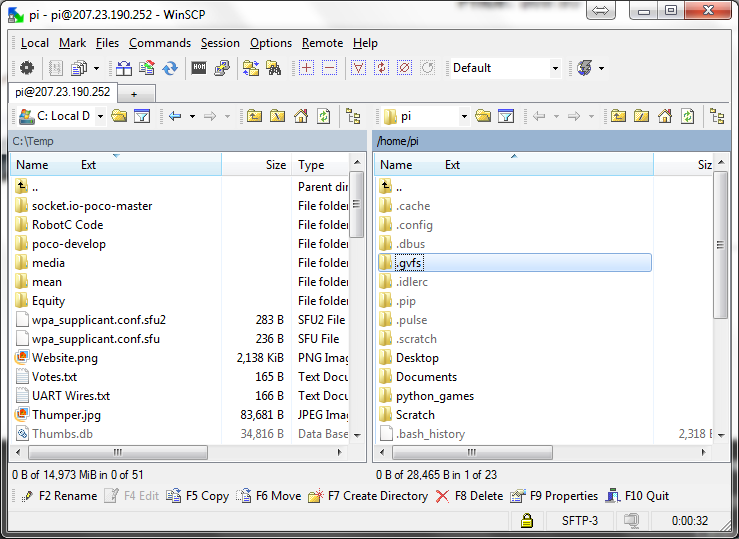
Like PuTTY, if the wireless network you are on or the firewall has not opened port 22, you will not be able to FTP into it from outside your internet network. You will have to connect to the same wireless or internet network to FTP into it.

If you have opened port 22, then you will be able to FTP into to the Pi from anywhere in the internet.

We will use WinSCP later to upload files to the Pi.



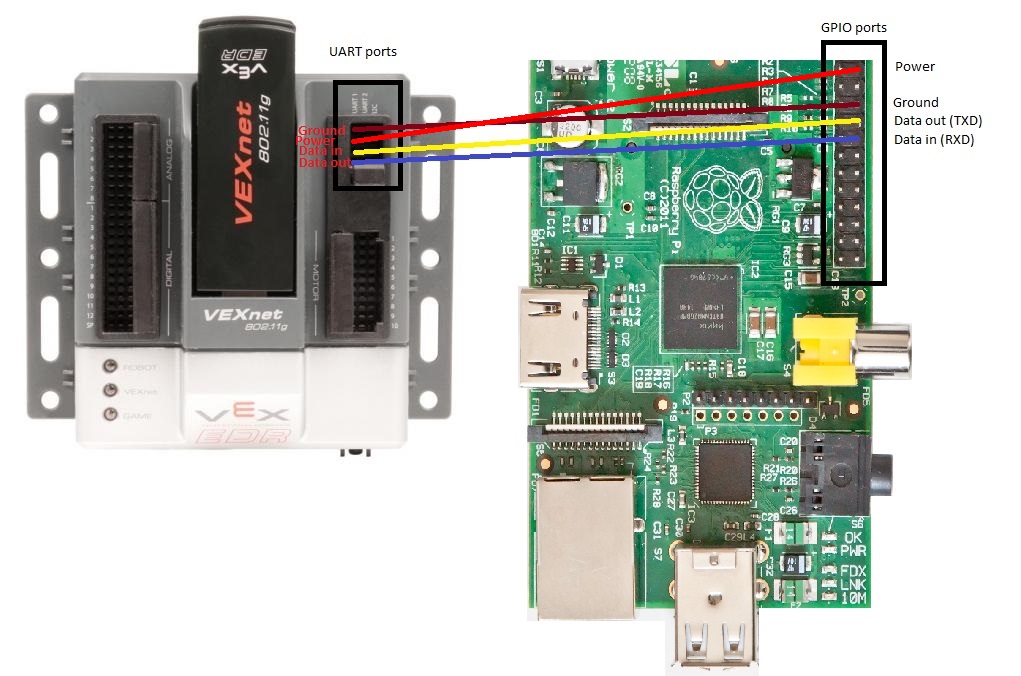
Type in the IP of the Pi you found earlier into **Host name**, use **pi** as the **User name** and **raspberry** as the **password**. Press **Login** to FTP into the Pi.



Inside the FTP client you will be able to drag files to and from the Pi to your computer.

# 5. Connecting the Cortex to the Pi

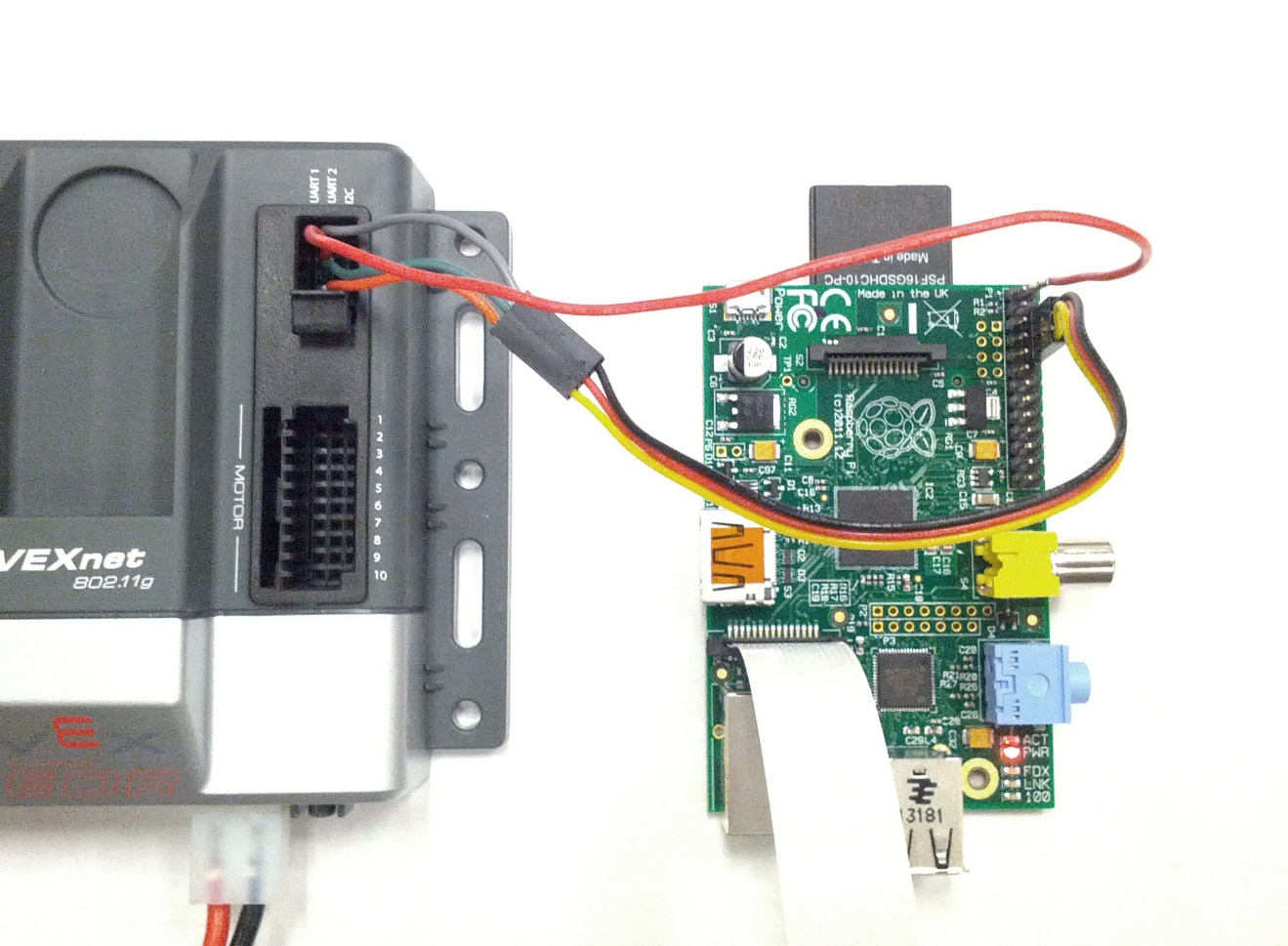
## 5.1 Wiring the Cortex to the Pi



Now it's time to connect the Cortex to the Pi. We are going to use UART 1 port on the Cortex to send commands from the Pi to the VEX Cortex.

The UART port will allow the Pi and the Cortex to send and receives commands from one another. Through the wireless adapter, the Pi receives commands from the Cloudy Robotics site which it will then send to the Cortex for it to take an action.

The UART port can also be used to power the Pi, so you can remove the micro USB cable that powers the Pi when you are using the Pi on the robot.



Use UART 1 on the Cortex and connect top port (ground) to the ground on the Pi. Second port on the Cortex (power) is connected to pin 2 on the Pi and supplies 5 volts to power the Pi. Third port (data in) connects to "data out" (pin 8) on the Pi. The last port "data out" connects to "data in" (pin 10) on the Pi.

Now you can send information from Raspberry Pi to the Cortex and vice-versa by having the appropriate RobotC code on the Cortex and Python script on the Pi uploaded.

# 6. Programming

## 6.1 Programming the VEX Cortex

First download RobotC for Cortex from below <http://www.robotc.net/download/cortex/>. A 30 day free trial is available.

Connect VEX Cortex to your computer using the USB A-to-A cable, open RobotC and update the firmware to the latest through the following commands on the menu bar:

**Robot** -> **Download Firmware** -> **Automatically Update VEX Cortex**

Then copy the sample RobotC code for communicating over UART from here [https://gist.github.com/timateus/1a817cbf1703896703e2#file-robotc-code-to-listen-for-uart-input-c](https://gist.github.com/timateus/1a817cbf1703896703e2%23file-robotc-code-to-listen-for-uart-input-c).

Refer to the sample program **UARTPortTest.c** under the **Advanced** folder if you need more examples.

In the sample code Cortex acts as a listener. The Cortex is listening to the commands that the Pi sends through UART. Below is a description of the commands the sample code reacts to:

1. An ‘a’ sent over UART will cause the motor in port 10 to take a PWM of 127
2. An ‘b’ sent over UART will cause the motor in port 10 to take a PWM of -127
3. An ‘c’ sent over UART will cause the motor in port 2 to take a PWM of 127
4. An ‘d’ sent over UART will cause the motor in port 2 to take a PWM of -127
5. An ‘e’ sent over UART will cause the motor in port 10 to take a PWM of 0

The UART port can receive only one character at a time. So the Cortex is listening to the characters the Pi sends and sends the corresponding commands to the motor.

The Pi and the Cortex should have the same speed of communication called baud rate. baud rate should be set to the same value on the Pi and the Cortex. In RobotC, the baud rate is set with this function:

*setBaudRate(uartOne, baudRate115200);*

The first parameter is the port you are using (UART 1 or UART 2) and the second parameter is the baud rate itself.

You may now upload compile and upload this sample code to your VEX Cortex after making any adjustments in the ports or commands. You may wish to add more if clauses to control more servos if you wish.

We will now code the Python script that will send these commands to Cortex through the UART port.

## 6.2 Programming the Pi

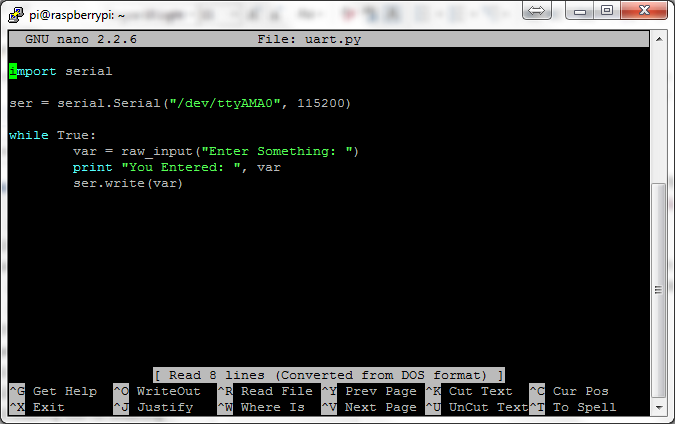
### 6.2.1 Upload Script to Pi

You may upload the Python script to the Pi by using WinSCP and downloading the following file, <https://gist.github.com/gmxu/61492e53ca5d08896ce2>, and dragging it to the Pi.

### 6.2.2 Create Script on Pi

You may also create the Python script remotely through PuTTY or directly. First boot up the Pi, and either through the keyboard and monitor or a PuTTY window, type the following command:

*sudo nano uart.py*



In the command line text editor, type in the following code:

**import** serial

ser **=** serial**.**Serial("/dev/ttyAMA0", 115200)

**while** True:

var **=** raw\_input("Enter something: ")

**print** "You entered: ", var

ser**.**write(var)

The first line, “import serial”, imports the Pyserial package into this script which is needed to communicate over UART.

The next line “ser **=** serial**.**Serial("/dev/ttyAMA0", 115200)”, initializes the serial port with a baud rate of 115200 which we specified earlier in the RobotC code. The /dev/ttyAMA0 signifies that the port we are using is the UART port on the Pi, which we connected to using the jumper wires.

The while loop is used to continuously send commands from the Python script to the Cortex. The “ser.write(var)” sends whatever is inside var over the serial port we’ve opened.

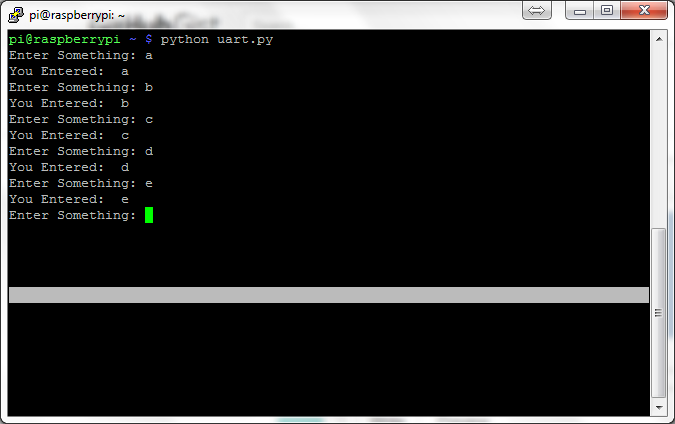
You won’t be able to run the script right now if the Cortex is not connected and powered on.

## 6.3 Running the Scripts

Connect the VEX Cortex to the Pi as seen in section 5.1. Upload the RobotC code to the Cortex and the Python script to the Pi as seen in sections 6.1 and 6.2. Connect whichever servos you are using to the Cortex and turn it on. Turn the Pi on.

Run the python script by entering the following command into the Pi:

*sudo python uart.py*



Enter the characters that were specified in the RobotC such as ‘a’, ‘b’, ‘c’, ‘d’, or ‘e’, and watch the servos react based on input from the Pi.