



www.phparch.com

January 2022
Volume 21 - Issue 1

php[architect]

The Magazine For PHP Professionals

Domain-Driven Resolutions

Introducing FilterIterators

Hack Your Home With a Pi

Effective Mentoring

ALSO INSIDE

The Workshop:

Apache and PHP Basics

Community Corner:

Patrick Allaert, PHP 8.1
Release Manager

Education Station:

Background Queues

PHP Puzzles:

Infamous Fizz Buzz

Security Corner:

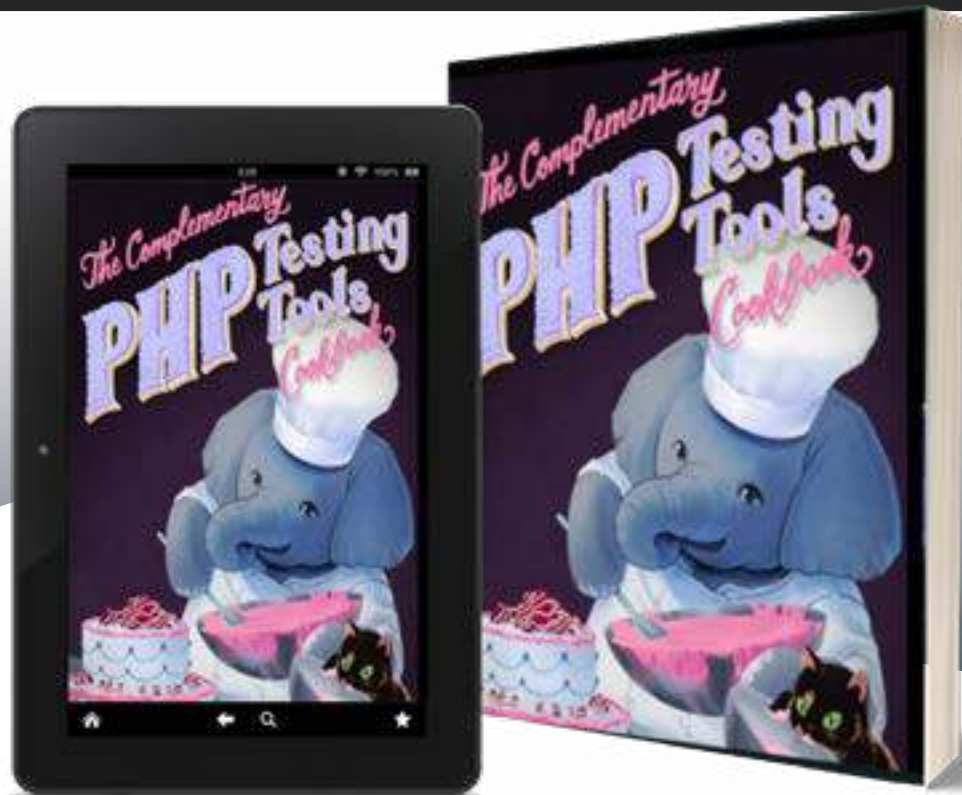
The Terrifying Scale of a
Security Bug

DDD Avenue:

Turning to Domain-
Driven Design

finally{ }:

Experts or Out-of-touch?



Learn how a Grumpy Programmer approaches improving his own codebase, including all of the tools used and why.

The Complementary PHP Testing Tools Cookbook is Chris Hartjes' way to try and provide additional tools to PHP programmers who already have experience writing tests but want to improve. He believes that by learning the skills (both technical and core) surrounding testing you will be able to write tests using almost any testing framework and almost any PHP application.

Available in Print+Digital and Digital Editions.

Order Your Copy
phpa.me/grumpy-cookbook

How to Hack your Home with a Raspberry Pi - Part 1 - Installing the OS and Configuring the Pi

Ken Marks

In September of 2017, I gave a talk called: 'Hack your Home with a Raspberry Pi' at the Madison PHP Conference. It was a half-day tutorial, walking participants through the process of configuring a Raspberry Pi, installing a LAMP stack on it, connecting a sensor, and ultimately having it send a text message to their phone.

Since then, I have been running this as a lab for my Advanced PHP class I teach at our local community college, Madison College.

One of the main reasons I gave this tutorial was to dispel the myth of the Internet of Things (IoT) being difficult. Sometimes people get scared away from writing code that talks to small embedded devices (i.e., sensors connected to Raspberry Pis or Arduino boards). My goal is to show you how easy it is to create a device that reports sensor data that you can get on a network. Along the way, you will see that deploying an application to a Raspberry Pi is just as simple as deploying to any Linux-based LAMP stack! [should I say, it's as easy as Pi :-)]

This article is the first of several in a series. I know php[architect] Magazine is a publication dedicated to PHP development, but I hope it doesn't scare you that not all the code you will be deploying/using in this series will be PHP 🤖. In addition to PHP (and SQL),

we will be using C/C++ and JavaScript. We will be creating a couple of UNIX services as well. So come along on this journey with me as I show you how to hack your home with a Raspberry Pi!

The goals for the whole series (can we call it a project?) are:

- Installing the Raspberry Pi operating system
- Configuring your Pi
- Installing a LAMP Stack on your Pi
- Connecting an accelerometer sensor to your Pi
- Logging data from an accelerometer to a database
- Creating and installing a Unix Service on your Pi
- Building a Web Service for querying the accelerometer data
- Building a plotting application to display the accelerometer data
- Sending a text message to your phone using SendMail

Wow! That's a pretty big list which is why it will take several articles to cover.

In this article, I will cover:

- Installing the Raspberry Pi operating system
- Configuring your Pi

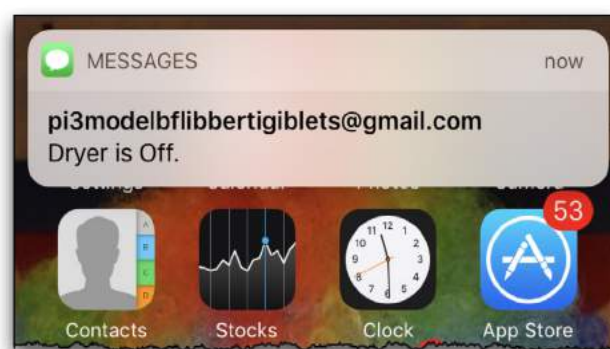
It starts with a story...

My initial idea and motivation for this series (project?) starts with a story about our old clothes dryer, which quit working one day. I was tasked with replacing it. Everything was great, and I got kudos for getting a decent dryer. However, there was one annoying flaw with this new dryer. We do our laundry in the basement, and this new dryer chimes at us — very quietly — when the clothes are done. Our old dryer buzzed its bloody head off when the dry cycle was complete, and I swear you could hear it from the street. Of course, being the forward-thinking Rube Goldberg kind of person I am, I said, "I can create an App for that!" So, since I was

New Samsung Dryer with Pi



Text Message from Pi that Dryer is Off

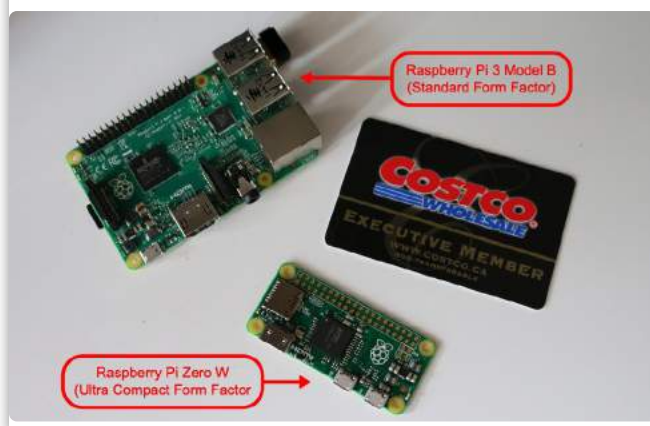


fairly familiar with Raspberry Pis and am always looking for an excuse to buy new geek toys, I set my sights on using the smallest Raspberry Pi on the market for my project.

List of essential components

The smallest Raspberry Pi I can use for this project is the *Raspberry Pi Zero W*:

Raspberry Pi Form Factors



If you use a *Raspberry Pi Zero W*, keep in mind that it only supports 2.4GHz wireless. So, if your wireless router only supports 5GHz, you might want to use another model of the Pi. The Pi 3 B+ and 4 B support 2.4 and 5GHz wireless ethernet. The upside of using the Pi 3 B+ or 4 B is that you won't have to do any soldering.

Do I really need to learn how to solder?

What did the soldering iron say to the capacitor? Go flux yourself!

All Raspberry Pis have a General Purpose Input Output

Raspberry Pi 40pin header

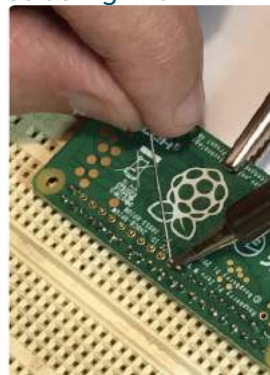


(GPIO) bus which is a dual-row male 40 pin header used to connect sensors and other devices to the Pi:

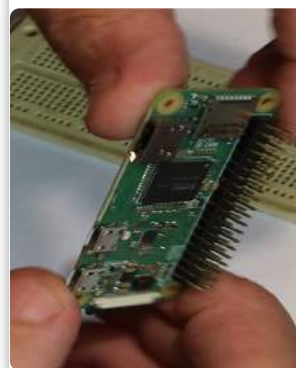
Later on, we'll be connecting an accelerometer sensor to the GPIO bus of the Pi. Because I'm using the Pi Zero W, I'll have to solder on a dual-row male 40 pin header onto the board to expose the GPIO bus pins since the Pi Zero W board does not come with it pre-populated.

I've included a few photos of me soldering the header onto the Pi and 6 pins onto one version of the accelerometer sensor.

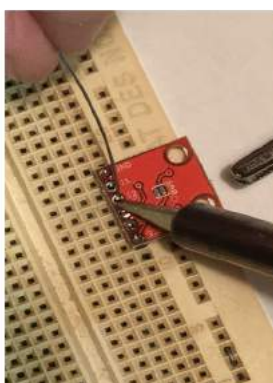
Soldering Pins



Head Soldered



Soldering Pins on Accelerometer



Pi Zero W in Case



Again, you can use any Raspberry Pi, but I thought it would be really cool to use a Raspberry Pi half the size of a credit card. I recommend you use a Raspberry Pi with wireless ethernet built-in since that will be the configuration I will document here.

Okay, let's start a numbered list of the five items we will need:

1. Your computer

You will need to have a computer connected to the internet to download the Raspberry Pi Operating System. Ideally, you will want to connect to your Pi via SSH from your computer. I'm going to assume you are running some flavor of Unix (Ubuntu, MacOS, etc.). However, there is plenty of information on the internet for communicating with your Raspberry Pi if you have a Windows PC.

2. Your home wireless router

I'm not going to go into how you configure this, as everyone is different. I will assume you have one and go through a generic method for connecting to it.

3. Raspberry Pi

I recommend getting a kit that includes a case, a power supply, and an SD card. I get mine from CanaKit¹, but you can get one from Amazon or any other Raspberry Pi reseller).

¹ CanaKit: <https://www.canakit.com>

Duel row 40 pin Male Header



Here is a list of things associated with your Pi you might want to have

- dual row 40 pin header (if you buy a Pi Zero W). You can get this from Amazon².
- soldering iron and solder (if you buy a Pi Zero W)
- case
- power supply for Pi
- SD card at least 8GB. (It's probably not worth getting one less than 16GB. I would recommend getting a card rated at UHS-1 or better. Most are).
- SD card reader. Many Raspberry Pi kits come with one that plugs into the USB port of your computer.
- HDMI cable (if you want to configure your Pi with a keyboard and monitor)
- keyboard (if you want to configure your Pi with a keyboard and monitor)
- monitor (if you want to configure your Pi with a keyboard and monitor)

4. Accelerometer

There are a lot of different accelerometer sensors available. I wanted one that uses the I2C bus for this project — more on that later. There is a small C program we will be using that communicates with an MMA8452Q triple-axis accelerometer. There are two slightly different board types:

Figure 1. MMA8452 from Sparkfun

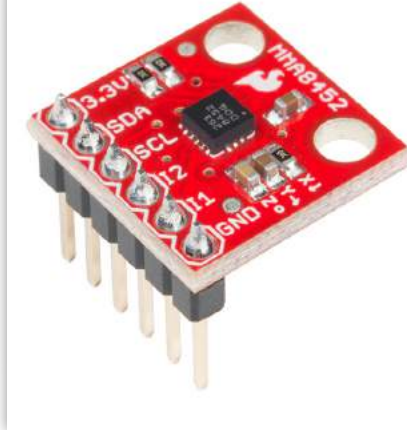


Figure 2. Keyes MMA8452Q from eBay



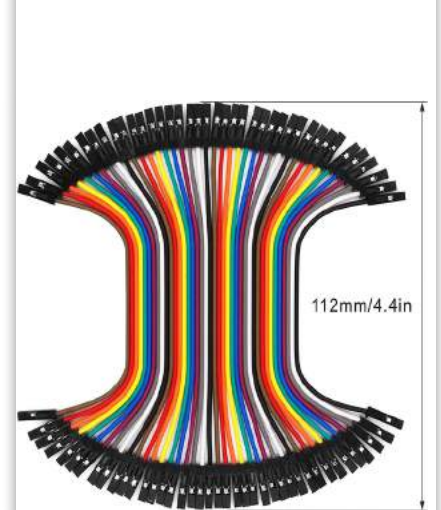
- The MMA8452 shown in Figure 1 can be purchased from Sparkfun³.
- And the Keyes MMA8452Q, which can be purchased from eBay⁴. See Figure 2.

The nice thing about the Keyes version is that it comes pre-populated with the 6 pins already soldered in

5. female to female 4 pin ribbon cable to connect the accelerometer to Pi

These typically come as a 40 pin ribbon cable, and you peel off 4 wires to connect the accelerometer to the GPIO bus of the Pi. You can purchase this from Amazon⁵.

40 Pin Ribbon Cable



Configuring your Raspberry Pi

Now that we have all of our essential equipment and components, we need to install an operating system and configure our Raspberry Pi.

Downloading the Raspberry Pi OS image

First, we need to get an image of the Raspberry Pi Operating System (OS) and get it onto our SD card. The best way to do that is to download the *Raspberry Pi Imager*⁶. There are links to download the imager for MacOS, Windows, or Ubuntu on the page. You can even use the *apt* package manager within a UNIX terminal window to install the *Raspberry Pi Imager*. After downloading the Imager package for your OS, follow the installation instructions.

Plug your SD card reader into your computer and insert the SD card you will be using for your Raspberry Pi.

³ Sparkfun:

<https://www.sparkfun.com/products/13926>

⁴ eBay: <https://phpa.me/ebay-accelerometer>

⁵ Amazon:

<https://phpa.me/amazon-fem-jumper-40>

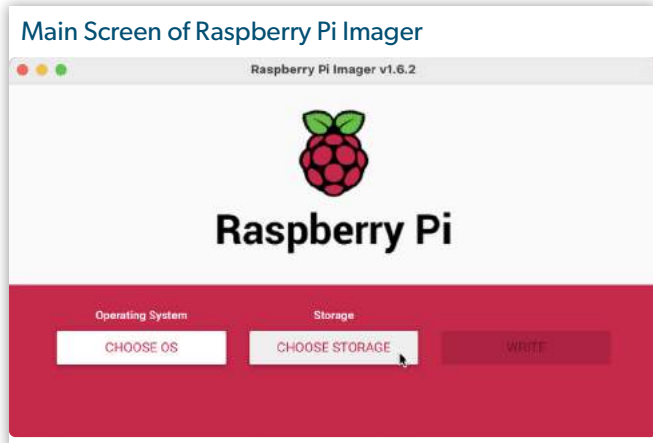
⁶ Raspberry Pi Imager:

<https://www.raspberrypi.com/software/>

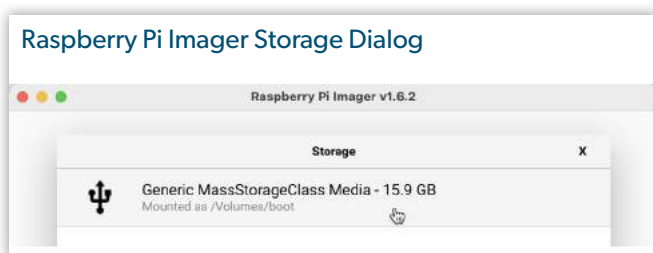
² Amazon:

<https://phpa.me/amazon-40pin-duel>

Launch the Raspberry Pi Imager, and you should see the main screen of the *Raspberry Pi Imager*:

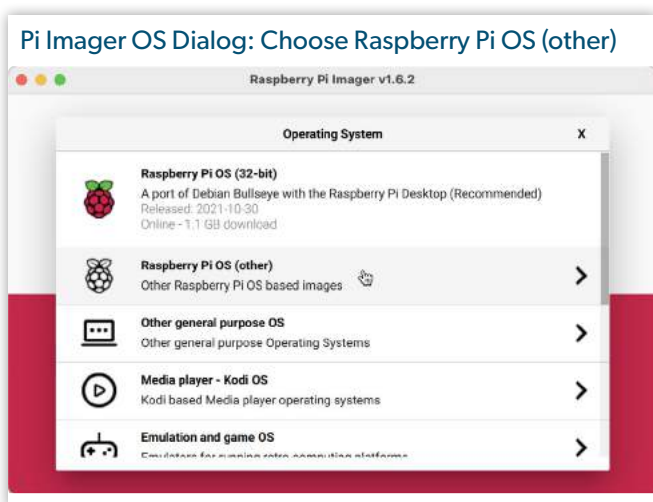


Select *CHOOSE STORAGE*, and a *Storage* dialog should be displayed:

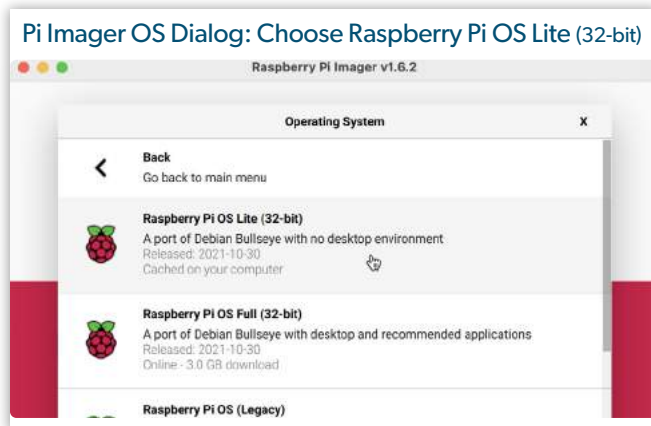


Select your SD card - typically listed as *Generic MassStorageClass Media* and its size.

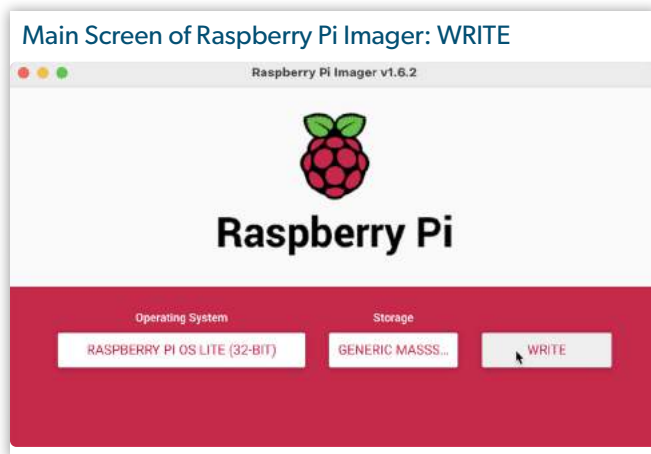
Now select *CHOOSE OS*, and the *Operating System* dialog should be displayed:



Select *Raspberry Pi OS (other)*, and the *Operating System* dialog will display a list of Raspberry Pi operating systems to choose from:



Select *Raspberry Pi OS Lite (32-bit)* - a Linux server installation that will minimize the amount of CPU utilization since we're not doing a full desktop installation. You will now be returned to the main screen of the *Raspberry Pi Imager*:



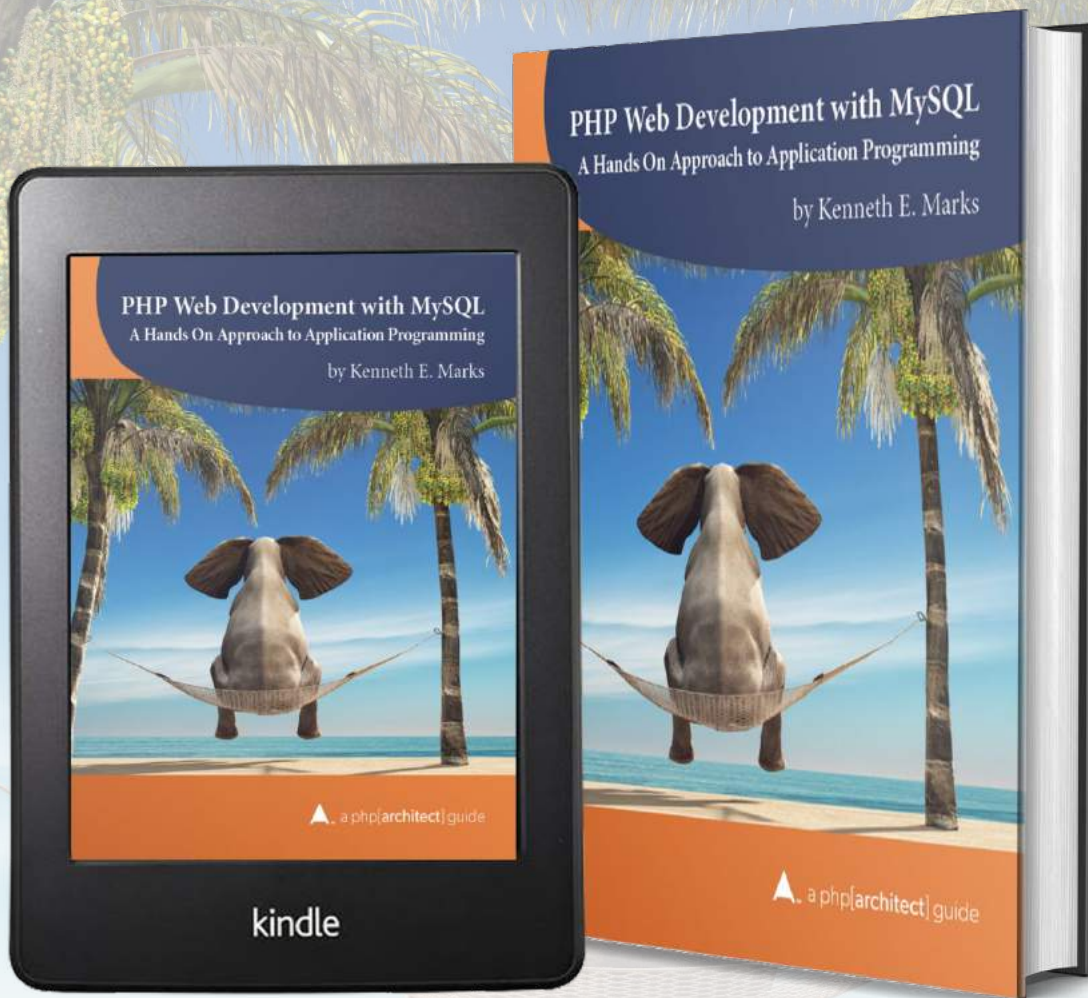
Now select *WRITE* to write the image to the SD card. You will now see a *Warning* dialog, letting you know all existing data on your SD card will be erased and ensure you want to continue. Select *YES*.

You will usually see a *Permissions* dialog pop up asking you to enter your admin password for your computer before you can write to your SD card.

After entering your credentials, the Raspberry Pi Imager will start writing the OS to your SD card. When the write operation is complete, a *Write Successful* dialog should be displayed. Select *CONTINUE*, and quit the *Raspberry Pi Imager* application. You can now remove your SD card from your SD card reader.

Now we need to configure our Raspberry Pi. There are two ways you can do this:

- Headless configuration
- Direct configuration



Learn how to build dynamic and secure websites.

The book also walks you through building a typical Create-Read-Update-Delete (CRUD) application. Along the way, you'll get solid, practical advice on how to add authentication, handle file uploads, safely store passwords, application security, and more.

Available in Print+Digital and Digital Editions.

Purchase Your Copy
<https://phpa.me/php-development-book>

I will first show you the headless configuration as I find it's the simplest. Then I will show you the direct configuration.

Headless configuration

So, let's say you have your Raspberry Pi, and you don't have a keyboard or extra monitor laying about, or like me, you forgot all of that (and the magic HDMI dongle that came with your Raspberry Pi Zero W kit)! Your only option at this point is to perform a headless configuration on your Pi. That's a good thing because it's usually the easiest option.

To create a headless configuration, we need to modify the contents of the SD card so that when the Raspberry Pi boots, it will:

- Allow us to SSH into the Raspberry Pi
- Connect the Pi to our Wi-Fi router

Re-insert your SD card back into your SD card reader.

Open up a terminal window and navigate to the root folder of your SD card. I'm on a Mac and will go to /Volumes/boot:

To enable SSH, we need to create an empty file called `ssh` in the boot folder of the SD card. Type the following command in the terminal window:

```
touch /Volumes/boot/ssh
```

Next, we need to add our Wi-Fi network information. For that, we need to create a file called `wpa_supplicant.conf` and put it in the boot folder of the SD card. Type the following command in the terminal window to create the `wpa_supplicant.conf` file:

```
touch /Volumes/boot/wpa_supplicant.conf
```

Next, we need to edit the `wpa_supplicant.conf` file and add

```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

network={
    ssid="NETWORK-NAME"
    psk="NETWORK-PASSWORD"
}
```

our Wi-Fi network SSID and password. Paste the following into the editor, and update the country code as needed, and your network SSID and password:

Save the file and exit your editor.

Eject the SD card and insert it into the Raspberry Pi

Mitch Allen has a complete set of instructions⁷ for how to create a headless configuration for your Raspberry Pi that covers all the major operating systems of your host computer.

Direct configuration

Direct configuration requires connecting a USB keyboard and HDMI monitor up to your Pi. Some Raspberry Pis

require dongles to connect a standard HDMI cable and USB keyboard. The Pi Zero even comes with a USB dongle.

Insert the SD card into your Raspberry Pi

Connect your Pi to a monitor and a keyboard.

Connect your power supply to your Pi and plug the power supply into a mains A/C wall socket.

After the Raspberry Pi boots up, log in with the user `pi` and the default password of `raspberrypi`.

Next, we need to add our Wi-Fi network information. For that, we need to create a file called `wpa_supplicant.conf` and put it into the `/etc/wpa_supplicant/` folder. We will be using the built-in nano editor to create the `wpa_supplicant.conf` file. Since this file requires root access, we have to create this file as *superuser*. Type the following command in the command-line terminal:

```
sudo nano /etc/wpa_supplicant/wpa_supplicant.conf
```

Now let's edit the `wpa_supplicant.conf` file and add our Wi-Fi network SSID and password. Paste the following into

```
country=US
ctrl_interface=DIR=/var/run/wpa_supplicant GROUP=netdev
update_config=1

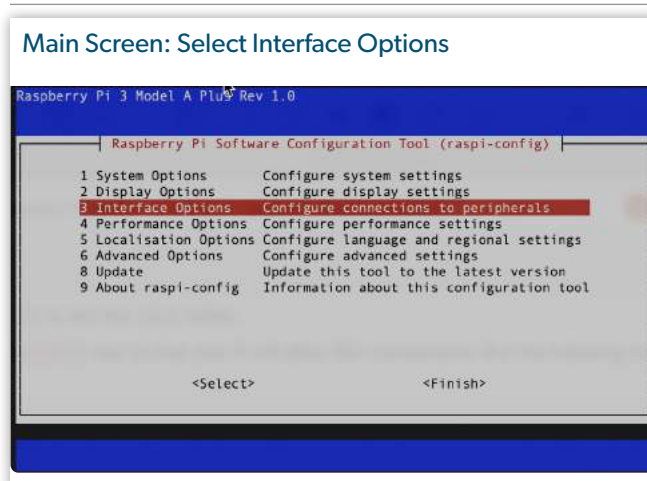
network={
    ssid="NETWORK-NAME"
    psk="NETWORK-PASSWORD"
}
```

the editor, and update the country code as needed, and your network SSID and password:

Type `^O` to write out the file and `^X` to exit the nano editor.

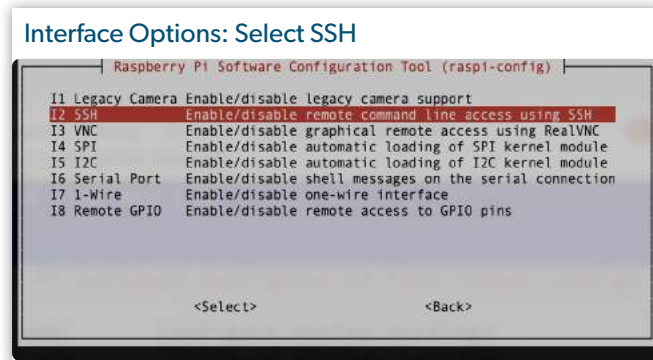
Next, we're going to run the `raspi-config` tool so that your Pi will allow SSH connections. Run the following command as *superuser* to display the main screen of the `raspi-config` tool.

```
sudo raspi-config
```



⁷ complete set of instructions: <https://phpa.me/desertbot-pi-wifi-setup>

Using the *arrow* keys, highlight 3 *Interface Options* and press Enter, and the *Interface Options* screen will be displayed:



Using the *arrow* keys, highlight I2 SSH and press Enter, and a dialog asking, “Would you like the SSH server to be enabled?” will pop up. Select Yes and press Enter.

Press Enter after the dialog, “The SSH server is enabled” pops up, and you will be returned to the *main* screen of the *raspi-config* tool. Using the tab key, select <Finish> and press Enter.

Type the following into the command-line terminal to shut down the Raspberry Pi:

```
sudo shutdown -h now
```

SSH into your Pi

Now connect the power supply to your Raspberry Pi and plug it into A/C mains.

After a few minutes, type the following into a terminal to remove any previous SSH keys generated for the host `raspberrypi.local`:

```
ssh-keygen -R raspberrypi.local
```

If you receive a *host not found* error, that’s okay. That just means you didn’t have any previous SSH keys generated for `raspberrypi.local`.

Next, SSH into your Pi with the user `pi` by typing the following command into the terminal:

```
ssh pi@raspberrypi.local
```

Type in the default password of `raspberry`. If all goes well, you should be logged in to your Raspberry Pi.

Run raspi-config

At this point, you want to configure your Raspberry Pi for your locale settings, along with changing the password. To do this, we will run the *raspi-config* program as *superuser*.

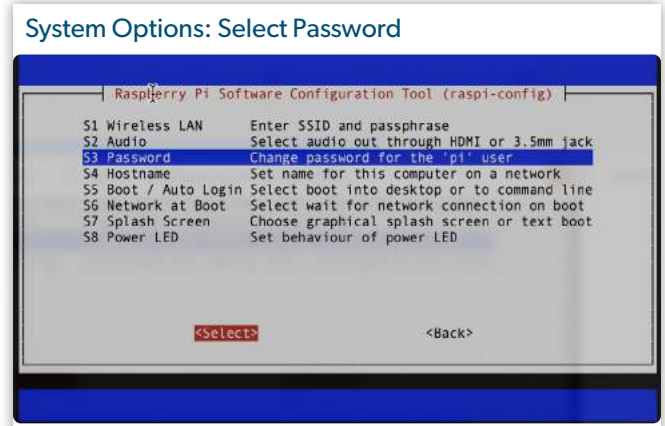
Assuming you are logged into your Pi using SSH, type the following command into the terminal:

```
sudo raspi-config
```

You should see the *raspi-config* main screen displayed again.

We first need to make sure we have the latest version of the *raspi-config* tool. Use the *arrow* keys to highlight 8 *Update* and tab over to the <Select> option and press Enter.

raspi-config will be updated to the latest version, and the *raspi-config* main screen will be redisplayed with the *System Options* highlighted. Tab over to the <Select> option and press Enter. The following *System Options* will be displayed, allowing you to change your password:



Use the *arrow* keys to highlight S3 *Password* and tab over to the <Select> option and press Enter. Follow the instructions to enter your new password. For purposes of this series, I’ll set mine to `pizerow`.

After changing your password, you will be taken back to the main screen of the *raspi-config* tool. Next, we need to modify the *Interface Options* to allow our Pi to use the GPIO bus and talk to our accelerometer sensor we will be connecting to it later on. Use the *arrow* keys to highlight 3 *Interface Options* and tab over to the <Select> option and press Enter.

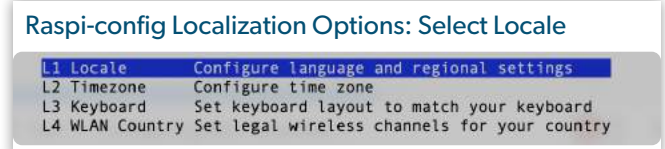
We specifically want to enable the *I2C* bus, as that is the protocol our accelerometer will use to communicate with the Pi. Use the *arrow* keys to highlight I5 *I2C* and tab over to the <Select> option and press Enter.

When you see this screen asking, “Would you like the ARM I2C interface to be enabled?”, tab over to <Yes> and press Enter.

Then select <Ok>, and you will be taken back to the main menu of the *raspi-config* tool.

Finally, we will change our localization options. Select 5 *Localisation Options* in the Main Screen.

We need to configure our *Locale*, *Timezone*, and *WLAN Country*:



First the *Locale*.

Using the *arrow* keys, I will scroll down until I find the selection of `en_GB.UTF-8 UTF-8`. Since I am not in the United Kingdom, I will disable `en_GB.UTF-8 UTF-8` by pressing the *spacebar*:

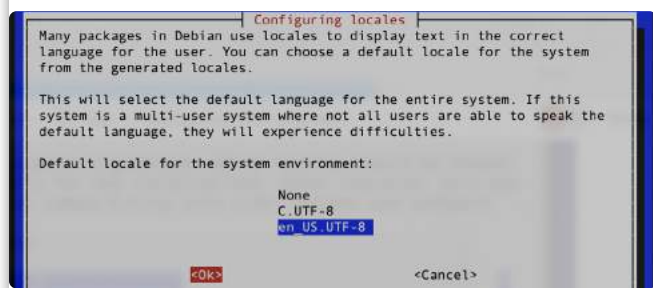
Configuring Locales: Disable en_GB.UTF-8



Which will make the selection (denoted by the asterisk) disappear. Then I will continue to use the *arrow* keys until I find my locale. I'll again use the *spacebar* to select my locale, which is `en_US.UTF-8 UTF-8` since I live in the US.

Then I'll tab over, and select `<Ok>` and I should see a screen asking me to pick the default locale for the system:

Configuring Locales: Select en_US.UTF-8 as Default

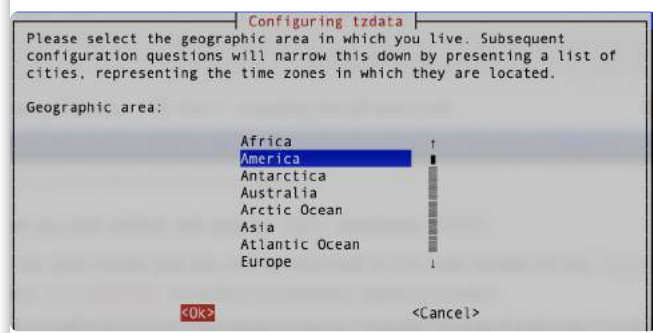


Use the *arrow* keys to select the *locale* you just added, tab over to `<Ok>`, and press Enter.

The necessary files will be generated for your *locale*, and you will be returned to the *main* screen of the *raspi-config* tool. Select the *Localization* option again and select L2 *Time-zone* from the *Localization Options* screen.

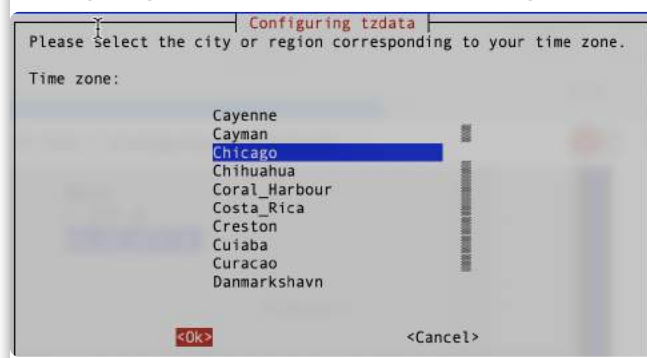
I'm in *America*, so that's what I'm selecting for my *Geographic area* from the *Configuring tzdata* screen:

Configuring tzdata - Geographic area : Select America



Doing so brings me to my *Time zone* selection. I am in Madison, Wisconsin, so I'll pick *Chicago* on the *Configuring tzdata* screen:

Configuring tzdata - Time zone: Select Chicago



The necessary settings will be changed for your time zone, and you will be returned to the main screen of the *raspi-config* tool. Select the *Localization* option again and select L4 *WLAN Country*. Again, I'm in the United States, so that is what I'll select.

The necessary settings will be changed for your *WLAN Country*, and you will be returned to the main screen of the *raspi-config* tool. Tab over to `<Finish>` and press Enter.

You will be asked if you would like to reboot the Pi. Select `<Yes>`. You will be logged out, and your SSH connection closed out of your terminal while the Raspberry Pi reboots.

Wait about a minute and log back into your Pi from your terminal window on your host computer. You will need to use your new password now to log in.

Conclusion

You now have a Raspberry Pi that is fully configured! This concludes our first installment of *How to Hack your Home with a Raspberry Pi*. In the next installment, we will install a full LAMP stack. Until next month, I hope you have fun playing around with your Pi!



Ken Marks has been working in his dream job as a Programming Instructor at Madison College in Madison, Wisconsin, teaching PHP web development using MySQL since 2012. Prior to teaching, Ken worked as a software engineer for more than 20 years, mainly developing medical device software. Ken is actively involved in the PHP community, speaking and teaching at conferences. @FlibertiGiblets



Learn how a Grumpy Programmer approaches testing PHP applications, covering both the technical and core skills you need to learn in order to make testing just a thing you do instead of a thing you struggle with.

The Grumpy Programmer's Guide To Testing PHP Applications by Chris Hartjes (@grmpyprogrammer) provides help for developers who are looking to become more test-centric and reap the benefits of automated testing and related tooling like static analysis and automation.

Available in Print+Digital and Digital Editions.

Order Your Copy
phpa.me/grumpy-testing-book



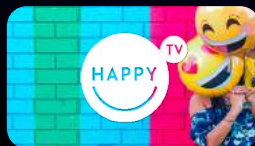
ATMOSPHERE

Build the future of TV streaming with Atmosphere.

Help us become the global leader in entertainment for business with our modern tech stack.

Laravel 8 | PHP 8 | Vue.js | SQL | tvOS and iOS

Revolutionizing the TV Experience for Businesses



Employee-First Culture | Creative Office Space | Open Vacation Policy | Full Benefits | Daily Lunches | Company Outings



See our job openings at
atmosphere.tv/careers