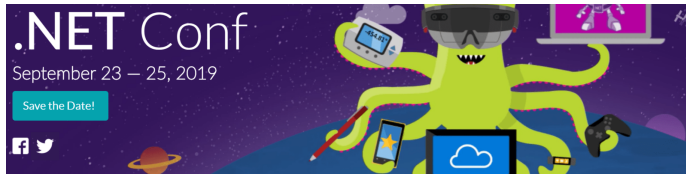


{ Pete Codes }



{Blog}



Explorations in Dot Net Core 3 for Raspberry Pi - Part 1 - Installation

Recent Talks

Build a Robot Arm with .NET 5, a Raspberry Pi, Blazor and SignalR – A Workshop

A Short History of IoT

IoT Lunch and Learn – Online

DevOps and IoT – Webcast

A tour of Azure IoT – Webcast

Upcoming Presentations

Festive Tech Calendar -
December 2020

Past Presentations

Brighton Web Development
Meetup - 29th October 2020
HackSoc - 13th October 2020
Techfast - 27th August 2020

and Hello World

🕒 September 6, 2019 👤 PeteCodes 💬
4 Comments

*This is part 1 of a (at least) 5
part blog series on Dot Net Core
on the Raspberry Pi....*

*You can see Part 2 – GPIO –
right here...*

*You can see Part 3 – Sending
Azure IoT Hub Messages – right
here...*

*You can see Part 4 – Receiving
Azure IoT Hub Messages – right
here...*

*You can see Part 5 – Remote
Deployment and Debugging –
right here...*

*Note: This post is up to date as
of 01/05/2020, for the following*

Virtual Azure Community Day -
28th July 2020
NeXt Generation - 13th May
2020
MS How To - 5th May 2020
MS How To - 29th April 2020
Azure Bootcamp Virtual - 23rd
April 2020
Dot Net York - 5th March 2020
Dot Net Sheff - 3rd March 2020
DDD North - 29th February 2020
DevOps Notts - 25th February
2020
Dot Net Oxford - 21st January
2020
AI Bootcamp - Nottingham -
14th December 2019
DDD Reading - 12th October
2019
HackSoc Nottingham - 10th
October 2019
LATi Bar - 3rd October 2019
Code Club and STEM
Ambassador Meetup - 2nd
October 2019
.net Liverpool - 26th September
2019
Notts IoT - 25th September
2019
Digital Lincoln - 30th July 2019
Tech Nottingham - 13th May
2019
Global Azure Bootcamp
Nottingham - 27-04-19
Notts IoT - 25th April 2019

versions;

- *Dot Net Core 3.1.201*
- ASP.NET Core 3.1.3
- Blazor Preview 3.2.0-RC1
- 20223.4

On September 25th 2019, the Dot Net Team released version 3.0 of the Dot Net Core Framework at .Net Conf.

To join in the fun, I held a special with Notts IoT, the IoT group I organise in Nottingham, where I gave a talk on Dot Net Core 3.0 on the Raspberry Pi.

This blog post is what I've learnt along the way to preparing for the talk...!

Thanks Shawty

I started out by following along with some great instructions by Peter "Shawty" Shaw. This took me through step by step how to get an ASP.Net Preview 4 of Dot Net Core Blazor up and running on a Raspberry Pi.

IoT Leeds - 18th March 2019
 DDD North - 2nd March 2019
 Lancashire Tech Talks - 28th February 2019
 Tech Nottingham - 11th February 2019
 Notts IoT - 24th January 2019
 Derbyshire Dot Net - 29th November 2018
 PHPem Unconference - 24th November 2018
 Lincoln Hack - 11-11-18
 Beeston Raspberry Jam - 20th October 2018
 Notts IoT Lightning Talks - 18th October 2018
 Notts Dev Workshop - 04-09-18
 Tech On Toast - 22nd August 2018
 Code Club Meetup - 09-07-18
 Notts IoT - 21st June 2018
 Notts IoT - 19th April 2018
 Notts IoT - 15th March 2018
 Notts Dev Workshop - 6th February 2018

Recent Posts

Install and use Microsoft Dot NET 5 with the Raspberry Pi
 An ESP8266 based, Morse Code Decoding Telegraph
 Short History of the Telegraph with DIY Science Experiments

I had to make a few adjustments to bring this up to date with the release version of Dot Net Core, but overall the original instructions worked really well... So thanks Shawty!

What you'll need

- A recent Raspberry Pi – Go shopping at the Pi Hut.
- You'll also need to setup a Samba Share, create a folder in your home directory called “**share**” and share that – Here's a link to a good guide.
- Then you'll need a SSH Client like PUTtY – Get it here.
- Next up you'll need a Code Editor – Use Visual Studio Code!

I'll be using a Raspberry Pi 3B+... It's important to note that Dot Net Core will only work on an ARMv7 processor or above... So that's a Raspberry Pi 2 and upwards... Unfortunately this rules out the Pi Zero and Pi Zero W as they have V6 processors.

Recent Comments

My Web App Journey – Taming the fire | Jonathan Tweedle on Explorations in Dot Net Core 3 for Raspberry Pi – Part 1 – Installation and Hello World

Install DotNet Core 3.1.0 onto Raspberry PI – TheCoreCoder on Explorations in Dot Net Core 3 for Raspberry Pi – Part 1 – Installation and Hello World

Install your first Hello World .NET Core 3.0 app on Raspberry – TheCoreCoder on Explorations in Dot Net Core 3 for Raspberry Pi – Part 1 – Installation and Hello World

Categories

Blog

Uncategorized

Meta

Log in

Entries feed

Comments feed

WordPress.org

There's now a Script!

Whoop... I've now created a script that installs everything below in one go!

On your Pi, just go ahead and run the following command;

```
wget -O - https://raw.githubusercontent.com
```

This runs the install script as root, so obviously take your own precautions here.... But, if you want to see what's inside, feel free to click the expandable section below...

► **Click here to see the contents of the install.sh file**

The “Bash Commands Only” Version...

► **Click here if you'd like the “Straight to the point” Bash Commands Only**

version..

Setting the Pi up

The first thing that we need to do is get the latest version of Dot Net Core onto the Pi.

Step by Step

Following along with the instructions, we need to make sure that our Pi is up to date... We can do this by running the following two instructions;

```
sudo apt-get -y update  
sudo apt-get -y upgrade
```

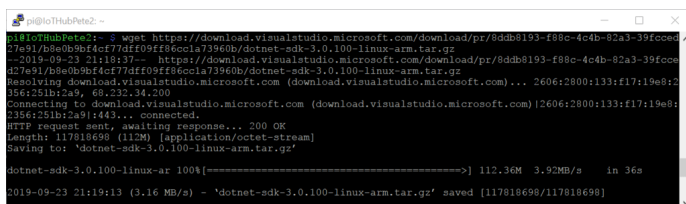
Next up, we need to install a couple of dependencies which are needed to run Dot Net Core apps on the Pi. These packages are required by Dot Net Core applications but aren't installed as standard with Raspbian. Further, as we don't have an install package for Dot Net Core for Raspbian, we'll need to install these ourselves;

```
sudo apt-get -y install libunwind8 gett
```

Now we download the Dot Net Core SDK and ASP.Net Core Runtimes.

First up the Dot Net Core SDK. This is the ARM32 set of binaries;

```
wget https://download.visualstudio.microsoft.com/download/pr/8ddb193-f88c-4c4b-82a3-39fccc0d7e91/b8e0b9b4cf77df09ff86cc1a73960b/dotnet-sdk-3.0.100-linux-arm.tar.gz
```



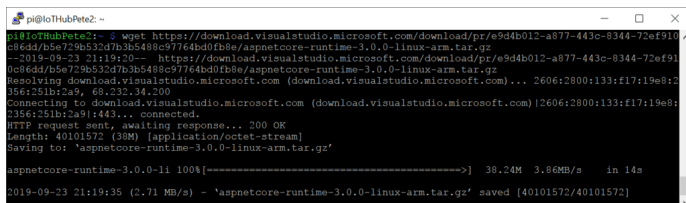
```
pi@ioTHubPete2: ~$ wget https://download.visualstudio.microsoft.com/download/pr/8ddb193-f88c-4c4b-82a3-39fccc0d7e91/b8e0b9b4cf77df09ff86cc1a73960b/dotnet-sdk-3.0.100-linux-arm.tar.gz
--2019-09-23 21:18:37-- https://download.visualstudio.microsoft.com/download/pr/8ddb193-f88c-4c4b-82a3-39fccc0d7e91/b8e0b9b4cf77df09ff86cc1a73960b/dotnet-sdk-3.0.100-linux-arm.tar.gz
Resolving download.visualstudio.microsoft.com (download.visualstudio.microsoft.com)... 2606:2800:133:f17:19e8:2:356:251b:2a9, 68.232.34.200
Connecting to download.visualstudio.microsoft.com (download.visualstudio.microsoft.com)|2606:2800:133:f17:19e8:2:356:251b:2a9:442... connected.
HTTP request sent, awaiting response... 200 OK
Length: 117818698 (112M) [application/octet-stream]
Saving to: 'dotnet-sdk-3.0.100-linux-arm.tar.gz'

dotnet-sdk-3.0.100-linux-ar 100%[=====>] 112.36M 3.52MB/s in 36s
2019-09-23 21:19:13 (3.16 MB/s) - 'dotnet-sdk-3.0.100-linux-arm.tar.gz' saved [117818698/117818698]
```

Download Dot Net Core 3 Linux Arm 32 Binaries

After this download the ASP.Net core Runtime, again, we're downloading the ARM32 version here;

```
wget https://download.visualstudio.microsoft.com/download/pr/e9d4b012-a877-443c-8344-72ef910c86d4/b5e729e532d7b3b5488c97764b0fb8e/aspnetcore-runtime-3.0.0-linux-arm.tar.gz
```



```
pi@ioTHubPete2: ~$ wget https://download.visualstudio.microsoft.com/download/pr/e9d4b012-a877-443c-8344-72ef910c86d4/b5e729e532d7b3b5488c97764b0fb8e/aspnetcore-runtime-3.0.0-linux-arm.tar.gz
--2019-09-23 21:19:20-- https://download.visualstudio.microsoft.com/download/pr/e9d4b012-a877-443c-8344-72ef910c86d4/b5e729e532d7b3b5488c97764b0fb8e/aspnetcore-runtime-3.0.0-linux-arm.tar.gz
Resolving download.visualstudio.microsoft.com (download.visualstudio.microsoft.com)... 2606:2800:133:f17:19e8:2:356:251b:2a9, 68.232.34.200
Connecting to download.visualstudio.microsoft.com (download.visualstudio.microsoft.com)|2606:2800:133:f17:19e8:2:356:251b:2a9:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 40101572 (38M) [application/octet-stream]
Saving to: 'aspnetcore-runtime-3.0.0-linux-arm.tar.gz'

aspnetcore-runtime-3.0.0-l 100%[=====>] 38.24M 3.86MB/s in 14s
2019-09-23 21:19:35 (2.71 MB/s) - 'aspnetcore-runtime-3.0.0-linux-arm.tar.gz' saved [40101572/40101572]
```

Download ASP.NET Core 3 Linux ARM 32 Runtime

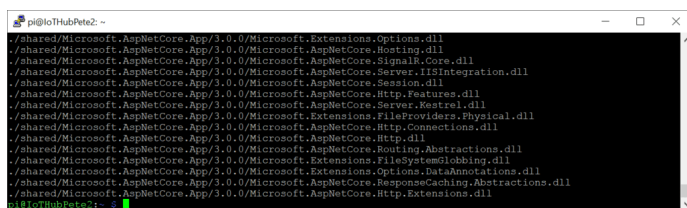
Once the files have finished downloading, we need to extract them... First we need

to store the files in a place where the whole operating system can gain access to them. So create a directory in *opt*;

```
sudo mkdir /opt/dotnet
```

First extract the SDK;

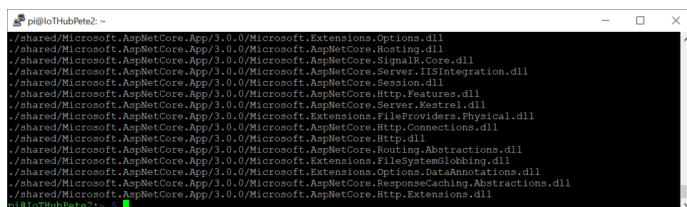
```
sudo tar -xvf dotnet-sdk-3.1.201-linux-a
```



Dot Net Core Binaries Extracted

Next we extract the ASP.Net runtime;

```
sudo tar -xvf aspnetcore-runtime-3.1.3-1
```



ASP.NET Core Runtime Extracted

Once the files have been extracted we need to make a symbolic link between the *opt/dotnet* directory to the *bin*

directory so that the operating system has access to them when we run our Dot Net Core apps. This in essence means that the SDK and runtime exist at both locations;

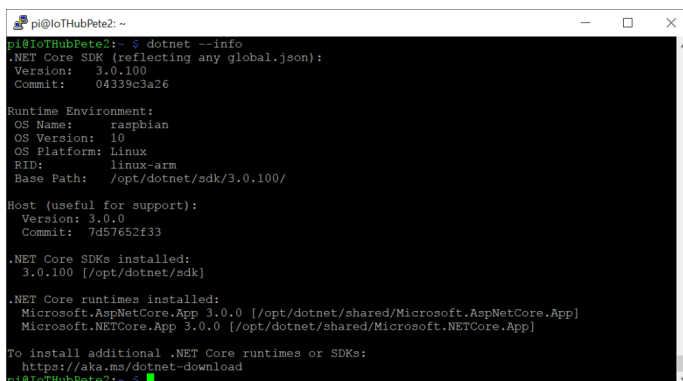
```
sudo ln -s /opt/dotnet/dotnet /usr/local
```

Are we there yet?

At this stage the setup of Dot Net Core should be done... So time to find out if everything is working... Run the following to see if it's all ok;

```
dotnet --info
```

If all went well, you should see the Dot Net Version Information reported in the terminal window;



```
pi@IoTHubPete2: ~  
pi@IoTHubPete2:~$ dotnet --info  
.NET Core SDK (reflecting any global.json):  
Version: 3.0.100  
Commit: 04339c3a26  
  
Runtime Environment:  
OS Name: raspbian  
OS Version: 10  
OS Platform: linux  
RID: linux-arm  
Base Path: /opt/dotnet/sdk/3.0.100/  
  
Host (useful for support):  
Version: 3.0.0  
Commit: 7d57652f33  
  
.NET Core SDKs installed:  
3.0.100 [/opt/dotnet/sdk]  
  
.NET Core runtimes installed:  
Microsoft.AspNetCore.App 3.0.0 [/opt/dotnet/shared/Microsoft.AspNetCore.App]  
Microsoft.NETCore.App 3.0.0 [/opt/dotnet/shared/Microsoft.NETCore.App]  
  
To install additional .NET Core runtimes or SDKs:  
https://aka.ms/dotnet-download  
pi@IoTHubPete2:~$
```

Dot Net Core Version Information

Hello (*Blazor*) World

Now that we have Dot Net Core up and running (Whoop by the way!), it's time to get our first ASP.Net Core website working...

Before we create our first ASP.net Core app though, we need to install the Blazor Template Pack;

```
dotnet new -i Microsoft.AspNetCore.Components.WebAssembly.Template
```

What's this *Blazor* thing I just installed then?

Originally developed as a pet project by Steve Sanderson at Microsoft, Blazor is Web UI Framework which is based on C#, Razor and HTML. Blazor compiles down to WebAssembly, which makes it super (or blazingly) fast (Hence the Razor with a "B").

Blazor allows developers to write client side .net... You may say “Oh, like Silverlight?”... Nope, no plugins needed... It’s just supported natively in most modern browsers.

One last thing to setup

We now need to make sure that the Operating system knows where to find our new Dot Net Core binaries.

So we don’t need to run a command every time we log in to the Pi, we need to add a line to our .bashrc file. This file will be run up each time the Pi starts up;

```
cd /home/pi  
sudo nano .bashrc
```

Scroll right to the bottom and add the following line;

```
export DOTNET_ROOT=/opt/dotnet
```

Now hit “**ctrl+x**” to exit and press “**y**” to save the file.

Time to scaffold our app

Firstly we need to create a directory for our project;

```
cd share  
mkdir rpiblazor  
cd rpiblazor
```

Now we can scaffold our new ASP.Net Core Blazor App;

```
dotnet new blazorserver
```

Scaffolding an ASP.Net Core Blazor App

Show me the files!

At this stage we need access to the files on our Pi as we need to make some edits....

If you've followed along with the guide I mentioned above, you'll have set up a shared folder on your Pi called Share.

On a Windows machine, we can map this shared folder to a drive on our machine, so we have an easy way of accessing it.

Go ahead and open Windows Explorer and navigate to your Pi. If you've simply installed vanilla Raspbian on your Pi, it's no doubt still called "RaspberryPi". So you can type the following into the Windows Explorer address bar;

```
\\raspberrypi
```

In my case, I've renamed my Pi, but overall it'll look something like;

Network Share

Now, if you right click on the "share" folder, you'll see a menu which includes a "Map Network Drive..." option;

Map Network Drive Menu

Hitting this menu option will show the
“Map Network Drive” dialog;

Map Network Drive Dialog

Just press the “Finish” button to map your
Network Drive.

Are you listening?

Now we have access to our files, we can
make the edits we need...

Basically, before we can test that our
“Hello World” style Blazor app is working,
we need to allow the app to be accessed
from an external PC (That is, unless
you’re happy to test directly on the Pi).

We can accomplish this by adding an extra line to the “program.cs” file in the root of our project.

First, let’s open our project in Visual Studio Code... if you’re on Windows and you’ve followed along with the steps above, you can enter your “rpiblazor” folder, find some blank space and right click. The menu shown will include a “Open with Code” item;

Open with Code Menu Item

Hitting “Open with Code” will open the whole folder in Visual Studio Code, allowing us easy access to all the files.

Alternatively, you can open VS Code yourself and choose the “Open Folder” option from the File Menu;

Open Folder Menu Item

Once you have your project loaded in VS Code, then open the “***Program.cs***” file from the Explorer on the left;

Program.cs

Towards the bottom of the file is the “CreateHostBuilder” function... Here’s where we set up various settings for our web applications including Dependency Injection, Middleware etc

Here is where we can set our application to listen on from all IP addresses rather than just on localhost, which, as the name suggests, can only be accessed locally.

Below the “*webBuilder.UserStartup<Startup>()*,” line add the following line;

```
webBuilder.UseUrls("http://*:5000");
```

This will instruct Dot Net to run this webapp at Port 5000 and listen out too all IP addresses (The * is a wildcard here).

Your finished code should now look like;

```
Finished Program.cs
```

Go ahead and save your file now.

Run Forest Run!

We're now ready to run our Web App.

Return to your SSH client and make sure you're in your "rpiblazor" directory.

Then run the following command to build your ASP.Net Core Blazor Web App;

```
dotnet build
```

This may take some time on a Pi (2 or possibly even 3 minutes), as there's quite a lot to do for this diminutive device! So just leave it be and wait for the magic to complete. It may look like nothing is happening for a while, so just be patient!

Dot NET Build Command

You can see in the above image that this took 2 minutes and 13 seconds on my Pi 3B+.

Next up, we can run our app... YEY! Still in the SSH client, run the following command to spin up the server;

```
dotnet run
```

This will take slightly less time than building, so you should see some results pretty quickly;

Dot NET Run Command

Once you get to this point, all that remains is to open your favourite browser and navigate to your Pi... For the most part, the address will be;

```
http://raspberrypi:5000
```

The address is of course different for my Pi, as I changed the name... However, all being well, you should be shown your brand new Dot Net Core 3.0 ASP.Net Blazor Web App...

Dot Net Core 3 ASP.Net Blazor Hello World Web
App

So, with that, you've gotten your first ASP.Net Core 3 Blazor Web App up and running...

In the next blog post, I'll take you through how to get a simple console App up and running and how to toggle an LED on and off...

Thanks for reading!

Next time...

In the next post we'll look at how to control the Raspberry Pi GPIO pins to control LEDs and Buttons.

***Part 2 – Controlling Raspberry
Pi GPIO with Dot NET Core 3 >***

Posted in: Blog

Filed under: Dot Net Core, Electronics, IoT,
Raspberry Pi

Explorations in Dot Net Core 3 for
Raspberry Pi – Part 2 –
Controlling GPIO →

**4 thoughts on
“Explorations in Dot Net
Core 3 for Raspberry Pi
- Part 1 - Installation
and Hello World”**

Pingback: Setting up .NET Core 3 on a Raspberry

PI – TheCoreCoder

Pingback: Install your first Hello World .NET Core

3.0 app on Raspberry – TheCoreCoder

Pingback: Install DotNet Core 3.1.0 onto

Raspberry PI – TheCoreCoder

Pingback: My Web App Journey – Taming the fire |

Jonathan Tweedle

Leave a Reply

You must be logged in to post a comment.

MICROSOFT AZURE MVP

STEM AMBASSADOR

PLURALSIGHT AUTHOR

Copyright © 2020 PJG Creations Ltd