HOW TO RUN .NET ON HEROKU

OCTOBER 9, 2016

.NET Core continues Microsoft's shift towards a more open development philosophy. It is open-source and cross-platform. With a little bit of setup, you can get a .NET Core web app running on Linux-powered Heroku. This guide will walk you through it.

This guide is catered for development on Windows, even though the resulting web app runs on Linux via Heroku. .NET Core (the framework) is cross-platform, but Visual Studio (the IDE) is not... yet. However, there are solutions for running Visual Studio on Mac.

Prerequisites

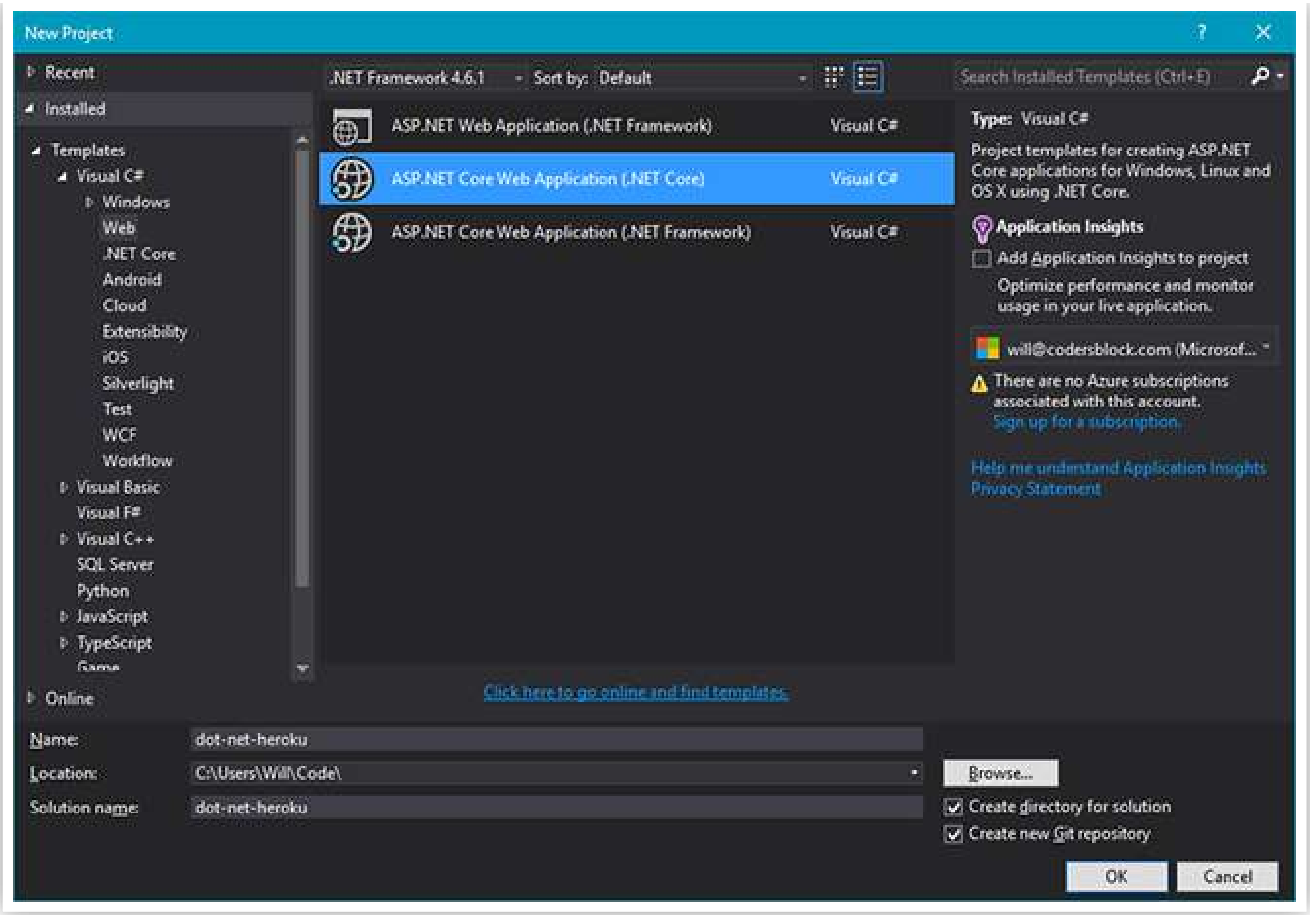
Make sure you have all of the following prerequisites ready to go.

- Heroku account created (sign up here).
- Heroku CLI installed (download here).
- Git installed (download here).
- Visual Studio 2015 (the free community version is fine) and .NET Core VS 2015 Tooling installed (download both here).
- NuGet installed (download here).

Create the Solution

Fire up Visual Studio, then go to **File** > **New** > **Project**.

Navigate to **Installed** > **Templates** > **Visual C#** (or **Visual Basic**, if that's your thing) > **Web**. Select **ASP.NET Core Web Application (.NET Core)**. Pick your **Name** (I'm using "dot-net-heroku" here) and set **Location** to the directory you want the solution directory to be created within. Check both **Create directory for solution** and **Create new Git repository**. Click **OK** to proceed.


New Project
Recent
Installed
Templates
Visual C#
Windows
Web
.NET Core
Android
Cloud
Extensibility
iOS
Silverlight
Test
WCF
Workflow
Visual Basic
Visual F#
Visual C++
SQL Server
Python
JavaScript
TypeScript
Game
Online
ASP.NET Web Application (.NET Framework) Visual C#
ASP.NET Core Web Application (.NET Core) Visual C#
ASP.NET Core Web Application (.NET Framework) Visual C#
Search Installed Templates (Ctrl+E)
Click here to go online and find templates.
Browse...
OK
Cancel

.NET Framework 4.6.1 · Sort by: Default

Type: Visual C#

Project templates for creating ASP.NET Core applications for Windows, Linux and OS X using .NET Core.

Application Insights

☐ Add Application Insights to project

Optimize performance and monitor usage in your live application.

will@codersblock.com (Microsof...

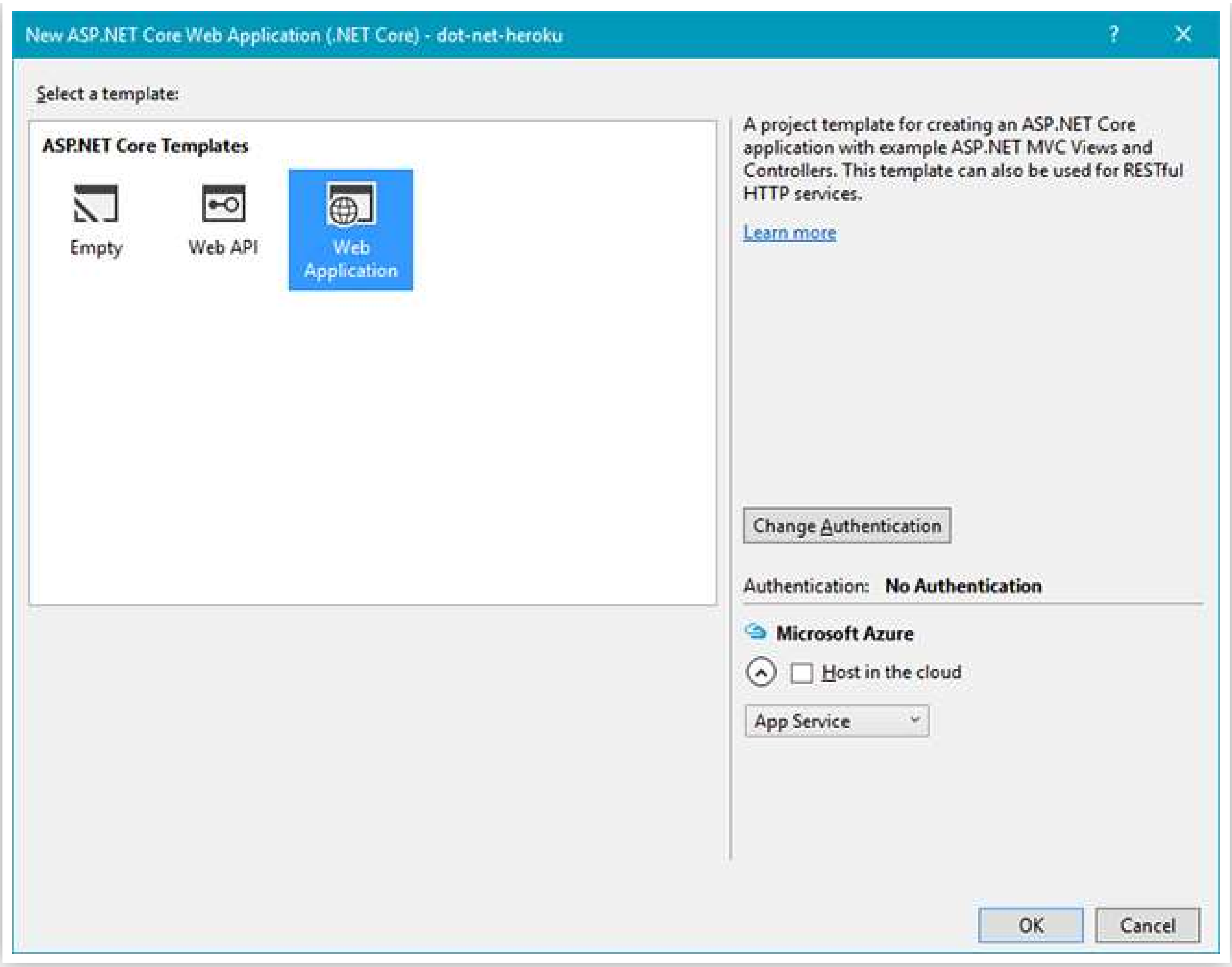
There are no Azure subscriptions associated with this account. Sign up for a subscription.

Help me understand Application Insights
Privacy Statement

Name: dot-net-heroku
Location: C:\Users\Will\Code\
Solution name: dot-net-heroku

✓ Create directory for solution
✓ Create new Git repository

On the next screen, select **Web Application** (**Web API** works too, if that's what you want). Click **OK**.


New ASP.NET Core Web Application (.NET Core) - dot-net-heroku
Select a template:
ASP.NET Core Templates
Empty
Web API
Web Application
A project template for creating an ASP.NET Core application with example ASP.NET MVC Views and Controllers. This template can also be used for RESTful HTTP services.
Learn more
Change Authentication
Authentication: No Authentication
Microsoft Azure
Host in the cloud
App Service
OK
Cancel

At this point you have a brand new solution. We'll need to tweak it a bit, though.

Modify the Solution

Still in Visual Studio, go to **View > Other Windows > Package Manager Console**. This will give you a console with a **PM>** prompt. Execute the following command.

```
Install-Package Microsoft.Extensions.Configuration.CommandLine
```

Now open up `Program.cs` for some edits. Near the top, add the following `using` directive.

```
using Microsoft.Extensions.Configuration;
```

Then update the `Main()` function as seen below.

```
public static void Main(string[] args)
{
    var config = new ConfigurationBuilder().AddCommandLine(args
    var host = new WebHostBuilder()
        .UseKestrel()
        .UseContentRoot(Directory.GetCurrentDirectory())
        .UseConfiguration(config)
        .UseIISIntegration()
        .UseStartup<Startup>()
        .Build();

    host.Run();
}
```

Your `Program.cs` should look like mine. With that, the solution is ready for deployment. Now to take care of things on the Heroku side.

Create the Heroku App

Open up your command line terminal of choice and navigate to the solution directory. Make sure you're logged into Heroku.

```
heroku login
```

Create the Heroku app with whatever name you want.

```
heroku create your-app-name
```

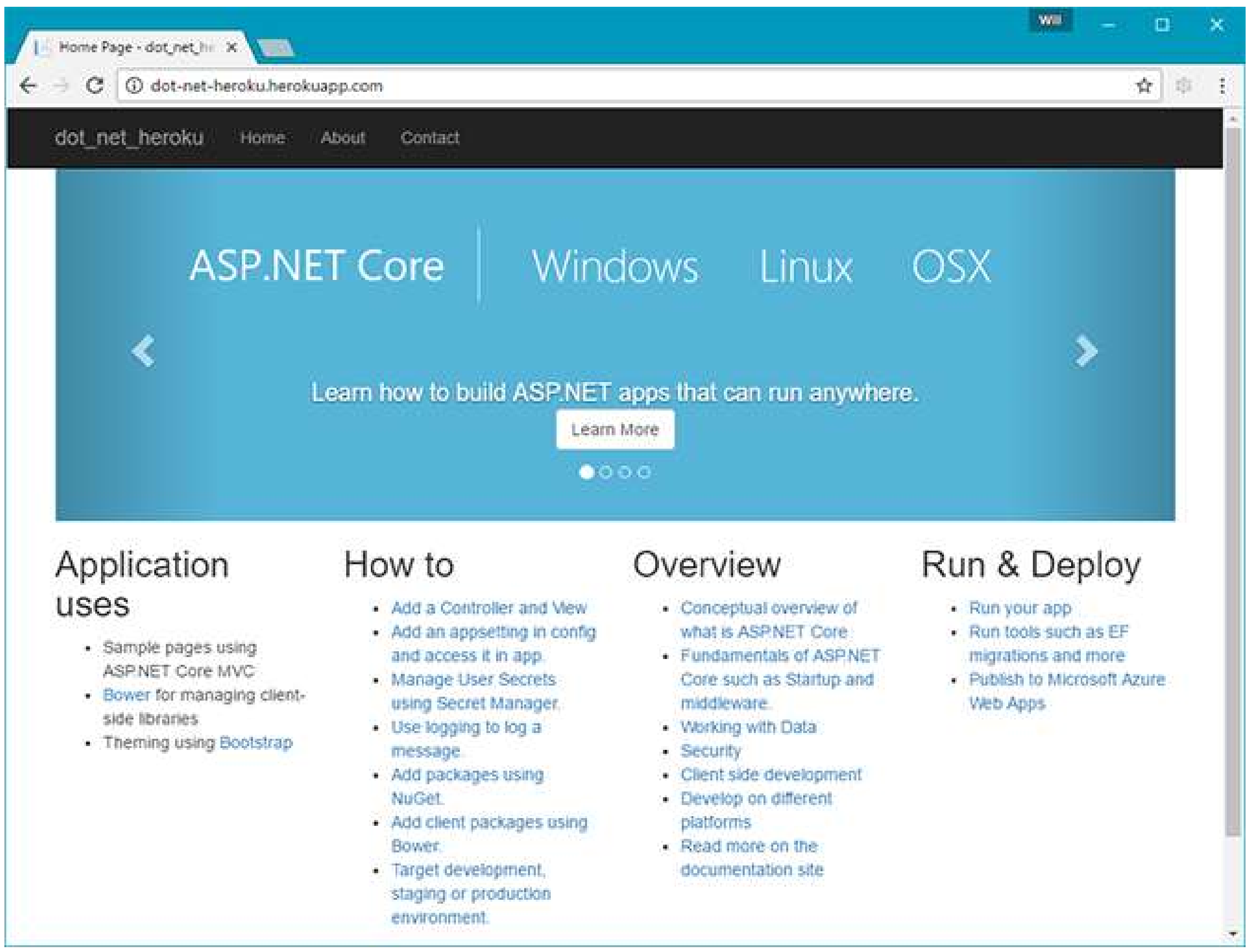
Now to add the buildpack. I've had the best results with noliar's fork, so that's what I'm using here.

```
heroku buildpacks:set http://github.com/noliar/dotnet-buildpack.
```

Finally, set up Git to deploy to Heroku and push.

```
heroku git:remote -a your-app-name
git commit -am "Setup for Heroku"
git push heroku master
```

Give the deployment some time to finish up, and there you have it. Go to http://your-app-name.herokuapp.com/ (with your app name, obviously) and you'll see your .NET web app running on Heroku. It should look like mine.



Application uses

- Sample pages using ASP.NET Core MVC
- Bower for managing client-side libraries
- Theming using Bootstrap

How to

- Add a Controller and View
- Add an appsetting in config and access it in app.
- Manage User Secrets using Secret Manager.
- Use logging to log a message.
- Add packages using NuGet.
- Add client packages using Bower.
- Target development, staging or production environment.

Overview

- Conceptual overview of what is ASP.NET Core
- Fundamentals of ASP.NET Core such as Startup and middleware.
- Working with Data
- Security
- Client side development
- Develop on different platforms
- Read more on the documentation site

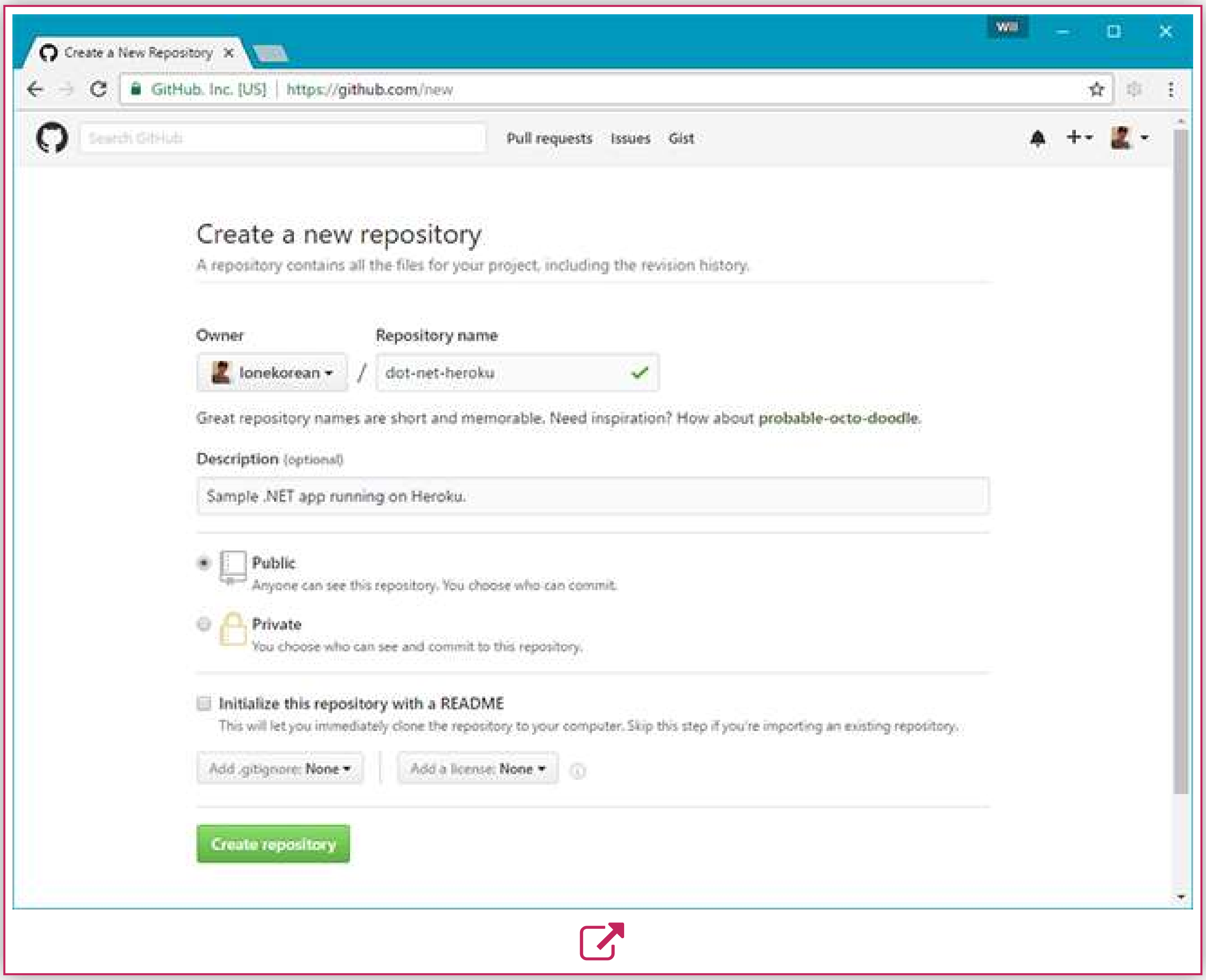
Run & Deploy

- Run your app
- Run tools such as EF migrations and more
- Publish to Microsoft Azure Web Apps

What you're seeing is the default index page for the template we used. At this point you're ready to delete the default pages and start working on your own.

Bonus: Hooking up to GitHub

You've got your .NET web app inside a local Git repo, but maybe now you want to add it to GitHub. Start by creating a new repo. To avoid conflicts, don't initialize with the `readme`, `license`, or `.gitignore` files (you can add these later).



Create a new repository

A repository contains all the files for your project, including the revision history.

Owner: lonekorean / Repository name: dot-net-heroku

Great repository names are short and memorable. Need inspiration? How about probable-octo-doodle.

Description (optional): Sample .NET app running on Heroku.

[✓ Public] Anyone can see this repository. You choose who can commit.

Private: You choose who can see and commit to this repository.

☐ Initialize this repository with a README

This will let you immediately clone the repository to your computer. Skip this step if you're importing an existing repository.

Add .gitignore: None | Add a license: None

Create repository

Back in your terminal, run these commands. Replace "your-username" and "your-repo-name" as appropriate.

```
git remote add origin https://github.com/your-username/your-repo
git push -u origin master
```

If you get a pop-up about security credentials, cancel out of it and enter your credentials in the terminal.

If the terminal won't accept your credentials, it may be because, like me, you have 2-factor authentication enabled. In this case, create a personal access token with **repo** access and use it in place of your password.

Closing Remarks

It's great seeing .NET break out of its Windows-only ecosystem to embrace the more open and cross-platform community of modern web development. Also, big thanks to noliar and all the other contributors for maintaining the .NET buildpack I'm using in this guide.

Quick Recap of CSSConf 2016

Coder's Block v5