



# Fast track your Angular 2 and .NET Core web app development

"If you're an enterprise .NET developer who is used to a mature, integrated, rock-solid toolchain, HTML/CSS/JavaScript development will feel to you like a bag of cats. And the bag is on fire."

*'badlife' (Reddit)*

Modern front-end development is a mess of competing frameworks, task runners and build systems that may just have you running back to the safety of ASP.NET and Visual Studio before you've even begun.

## The slow road

With Angular 1, you could take your existing ASP.NET web application, add a few script references and get on with adding some SPA goodness to your site.

Angular 2 changes all that, there is an entire ecosystem that you need to get up and running. You'll be faced with setting up Grunt/Gulp or WebPack, NodeJS, Typescript and making it all work together before you can even think about writing "Hello World".

Ignoring the obvious debate about the merits of this approach and whether all this complexity is really worth it (that's a conversation for another day) you're left wondering if there's any way to test out Angular 2 and build something without learning every nuance of Node, WebPack or anything else deemed necessary by the Angular gods (AKA Google).

## The highway

Before you run screaming back to the (relative) safety of Visual Studio. Here's another option.

### Update – 21st February 2017

If you're using Visual Studio 2017, check out an [alternative way of creating your project using the dotnet new command](#).

If you're using Visual Studio 2015 then you can take a shortcut by using the **ASP.NET Core template pack**. The pack includes a template for creating an Angular 2 with .NET Core project which will have you at "Hello World" in a few clicks.

That's not to say you won't need to learn all about WebPack and Node Modules at some point in the future, but **you don't need to derail your learning by getting all that set up from day one.**

Make sure you have the following prerequisites installed.

[Visual Studio 2015 Update 3](#) – Update 3 is required.

[.NET Core Tooling Preview 2 or later](#) (.NET Core Runtime 1.0.x and 1.1.0 are supported).

[TypeScript 2](#)

[Node.js version 4 or later](#)

Then grab and install the templates.

[Download the ASP.NET Core Template Pack.](#)

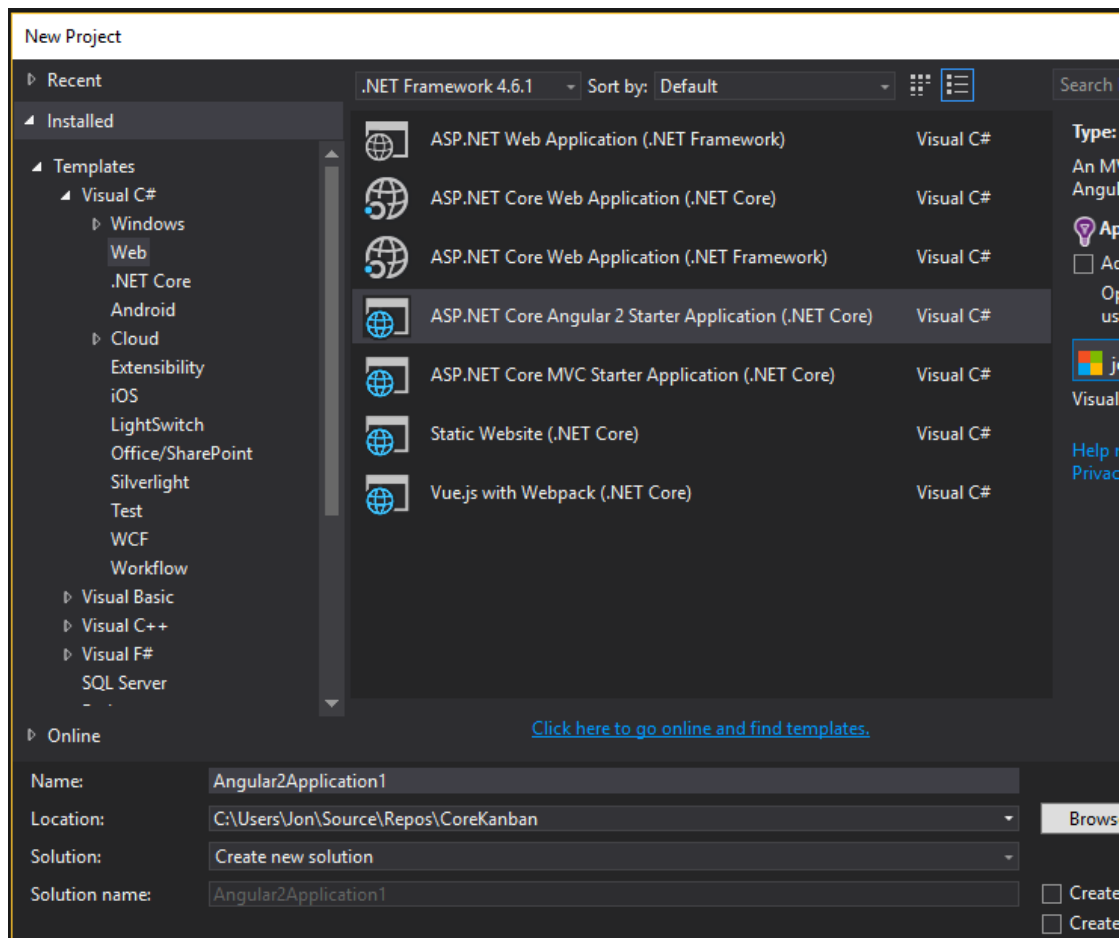
For more detailed background information about the template pack (and a few gotchas you might run into), check out [Steve Sanderson's announcement post](#).

## Hello World

Phew, that's still a lot of prerequisites and dependencies but now you can sit back, relax, and let the template set up your shiny new web application for you.

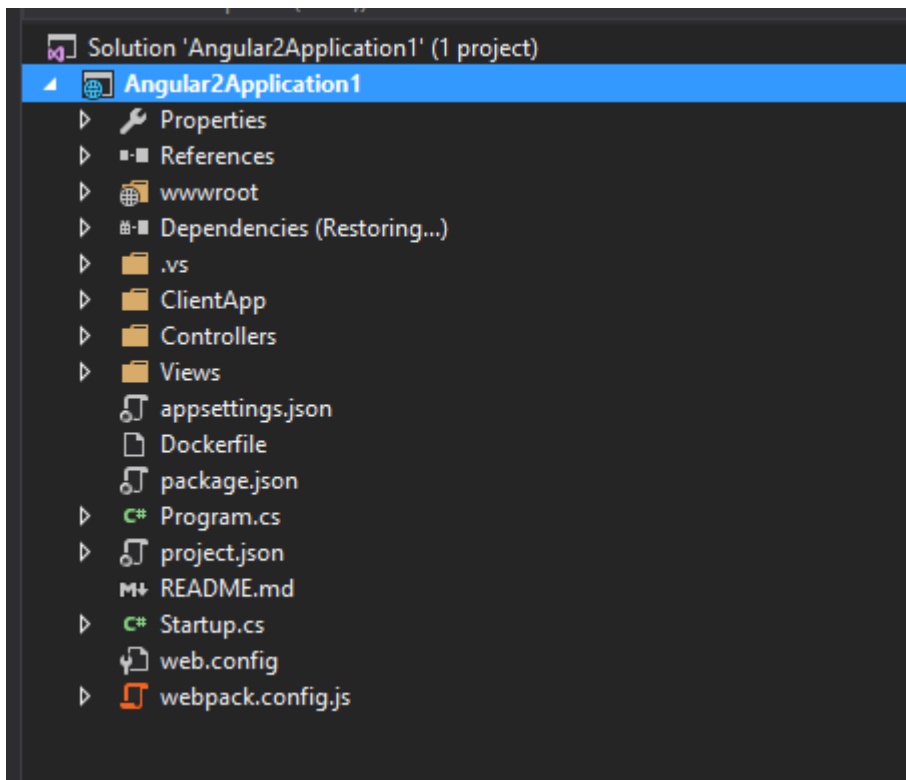
Open up Visual Studio and create a new project.

You should find some new templates available under “Web”.

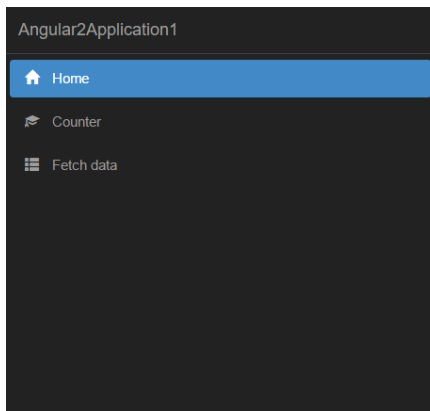


Choose **ASP.NET Core Angular 2 Starter Application (.NET Core)** and fill out the usual details (name, location etc).

Once Visual Studio's finished, you'll see something like this.



Before we go into the details, hit CTRL-F5 to preview your app.



## Hello, world!

Welcome to your new single-page application, built with:

- ASP.NET Core and C# for cross-platform server-side code
- Angular 2 and TypeScript for client-side code
- Webpack for building and bundling client-side resources
- Bootstrap for layout and styling

To help you get started, we've also set up:

- **Client-side navigation.** For example, click *Counter* then *Back* to return here.
- **Server-side prerendering.** For faster initial loading and improved SEO, your Angular 2 app is prerendered on the server.
- **Webpack dev middleware.** In development mode, there's no need to run the `webpack` build tool. You can use the `webpack-dev-server` to serve the application.
- **Hot module replacement.** In development mode, you don't even need to reload the page after making changes to the code.
- **Efficient production builds.** In production mode, development-time features are disabled, and the `webpack` build tool is used to create an optimized production bundle.

## Made It

So you did it, you got to Hello World without having to manually set up WebPack with Angular 2 and Typescript. But what can you do now?

In [the next post we'll explore the solution and add our first component.](#)

photo credit: tudedude [Arduino Controller For Mini Mill Stepper Motor](#) via [photopin](#) (license)

All posts in the **Checking the weather with Angular 2 and ASP.NET Core** series.

- **Fast track your Angular 2 and .NET Core web app development**
- [Angular 2 and .NET Core – your first component](#)
- [Angular 2 and .NET Core – route directly to your components](#)
- [Display the weather using Angular 2 and .NET Core Web API](#)
- [Fetch the current weather using ASP.NET Core Web API and OpenWeather](#)
- [Send form input via an Angular 2 component to ASP.NET Core Web API](#)

## Want to learn .net Core?

- Want to learn .NET Core but **don't know where to start?**
- **Don't have time to keep up** with everything Microsoft is putting out?
- Stuck on legacy apps when you **want to build something new?**

Enter your details below and get regular updates; **learn how to build better .NET web apps.**

First Name

Email

Learn .NET Core today

---

© 2018 - Jon Hilton