

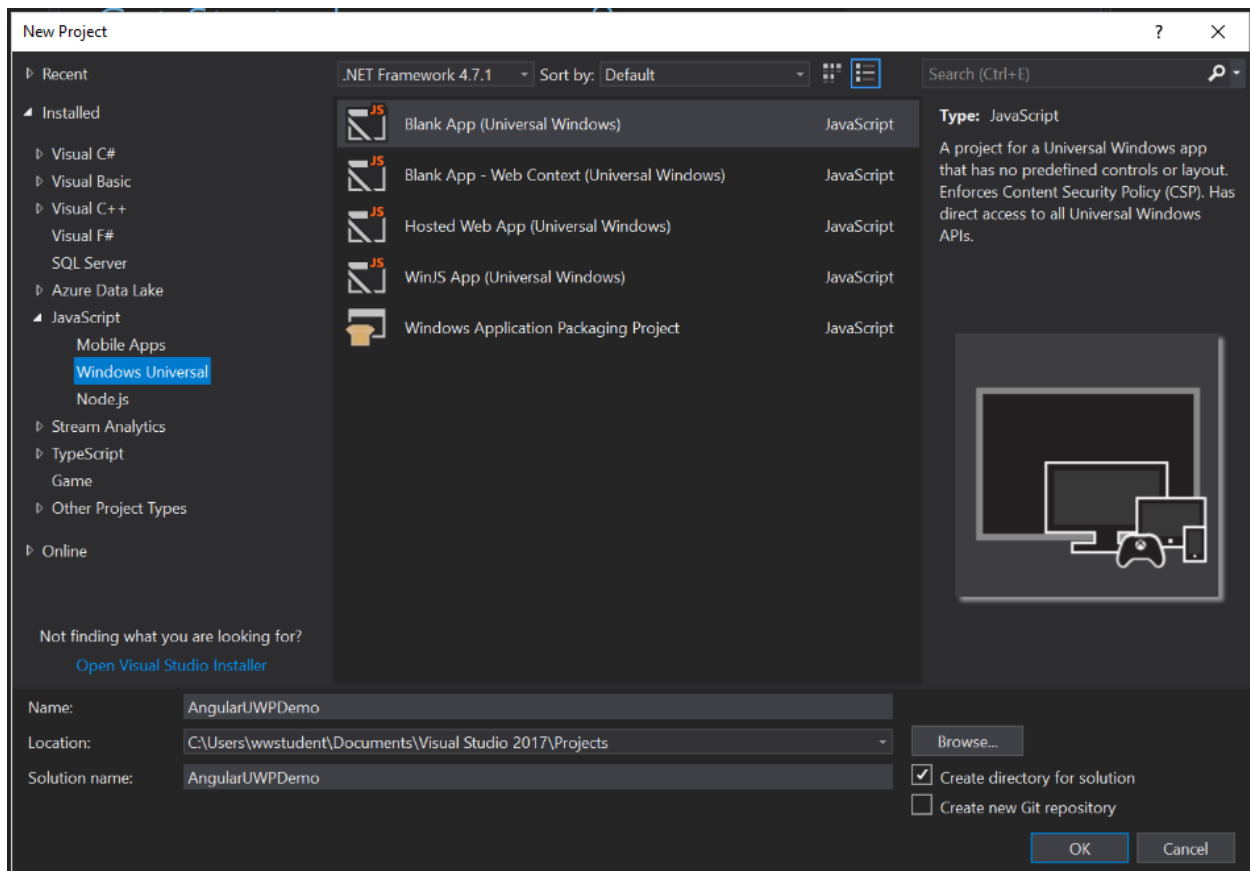
Adding Angular to UWP the unofficial write-up

Assuming you have the following:

- VS Community or better installed: <https://www.visualstudio.com/downloads/>
- NodeJs installed: <https://nodejs.org/en/download/>
- TypeScript installed (npm through cmd or powershell): `npm install -g typescript`
- Angular CLI installed (npm through cmd or powershell): `npm install -g @angular/cli`

Okay, now that is out of the way open Visual Studios and create a new project.

On the new project window find the JavaScript→Windows Universal tag on the left and select blank app on the right as seen below.



Now open the powershell or cmd and navigate to the application you just created. Inside the folder where the .sln file is located. Once there type `ng new AngularUWPDemo`. Where AngularUWPDemo is the name of the application that was just created. This will add the Angular framework and TypeScript configuration files to your project.


The output should look something like this:

```
Windows PowerShell
Copyright (C) 2016 Microsoft Corporation. All rights reserved.

PS C:\Users\wwstudent> cd "C:\Users\wwstudent\Documents\Visual Studio 2017\Projects\AngularUWPDemo"
PS C:\Users\wwstudent\Documents\Visual Studio 2017\Projects\AngularUWPDemo> ng new AngularUWPDemo
  create AngularUWPDemo/e2e/app.e2e-spec.ts (297 bytes)
  create AngularUWPDemo/e2e/app.po.ts (208 bytes)
  create AngularUWPDemo/e2e/tsconfig.e2e.json (235 bytes)
  create AngularUWPDemo/karma.conf.js (923 bytes)
  create AngularUWPDemo/package.json (1320 bytes)
  create AngularUWPDemo/protractor.conf.js (722 bytes)
  create AngularUWPDemo/README.md (1030 bytes)
  create AngularUWPDemo/tsconfig.json (363 bytes)
  create AngularUWPDemo/tslint.json (2985 bytes)
  create AngularUWPDemo/.angular-cli.json (1250 bytes)
  create AngularUWPDemo/.editorconfig (245 bytes)
  create AngularUWPDemo/.gitignore (516 bytes)
  create AngularUWPDemo/src/assets/.gitkeep (0 bytes)
  create AngularUWPDemo/src/environments/environment.prod.ts (51 bytes)
  create AngularUWPDemo/src/environments/environment.ts (387 bytes)
  create AngularUWPDemo/src/favicon.ico (5430 bytes)
  create AngularUWPDemo/src/index.html (301 bytes)
  create AngularUWPDemo/src/main.ts (370 bytes)
  create AngularUWPDemo/src/polyfills.ts (2667 bytes)
  create AngularUWPDemo/src/styles.css (80 bytes)
  create AngularUWPDemo/src/test.ts (1085 bytes)
  create AngularUWPDemo/src/tsconfig.app.json (211 bytes)
  create AngularUWPDemo/src/tsconfig.spec.json (304 bytes)
  create AngularUWPDemo/src/typings.d.ts (104 bytes)
  create AngularUWPDemo/src/app/app.module.ts (314 bytes)
  create AngularUWPDemo/src/app/app.component.html (1120 bytes)
  create AngularUWPDemo/src/app/app.component.spec.ts (986 bytes)
  create AngularUWPDemo/src/app/app.component.ts (207 bytes)
  create AngularUWPDemo/src/app/app.component.css (0 bytes)
Installing packages for tooling via npm.
Installed packages for tooling via npm.
Successfully initialized git.
Project 'AngularUWPDemo' successfully created.
PS C:\Users\wwstudent\Documents\Visual Studio 2017\Projects\AngularUWPDemo>
```

Now from here cd AngularUWPDemo. Once inside the folder type npm start. This will start the node server, give you the location to access the app (<http://localhost:4200>), and compile the application. Navigate to the site and check it out (I recommend to stay away from Internet Explorer). In the powershell or cmd window use **ctrl and c together twice to exit the server**.

Welcome to app!



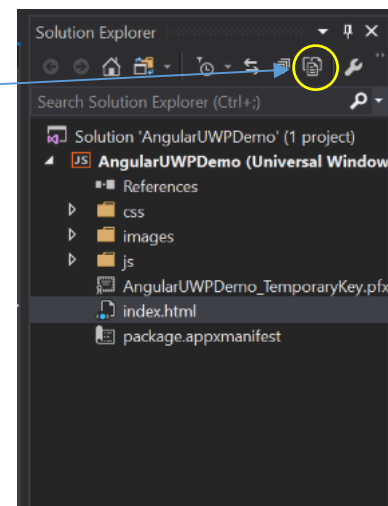
Here are some links to help you start:

- [Tour of Heroes](#)
- [CLI Documentation](#)
- [Angular blog](#)

Leave powershell open and go back to your VS Window.

Now time for some attention to detail stuff.

Click this icon on the solution explorer.



Open the index.html file from the root and remove the div and script tag. Add a tag that says `<app-root></app-root>`.

Once clicked, you will have to select the following then right click and select **“include in project”** from the context menu:

- Src
- .angular-cli.json
- .editorconfig
- Karma.conf.js
- Package.json
- Package-lock.json
- Protractor.conf.js
- Tsconfig.json
- Tslint.json

Optionally, you can delete the following:

- Css
- Js

Create the following folders in the root of the solution:

- dist
- assets

Find src folder in the root of the solution explorer and expand it. Right click on the src folder and **add** a new folder name styles. Then drag the styles.css to this folder.

Just a little more setup to go. Open the tsconfig.json file. Replace the contents of the file with the following and then save changes and restart VS.

```
{
  "compileOnSave": false,
  "compilerOptions": {
    "outDir": "./dist/",
    "baseUrl": "src",
    "sourceMap": false,
    "declaration": false,
    "moduleResolution": "node",
    "emitDecoratorMetadata": true,
    "experimentalDecorators": true,
    "target": "es5",
    "watch": false,
    "typeRoots": [
      "node_modules/@types"
    ]
  }
}
```

```
],  
  "lib": [  
    "es2016",  
    "dom"  
  ]  
}  
}
```

Save the changes and close the file. Open the .angular-cli.json file and replace the contents with the following:

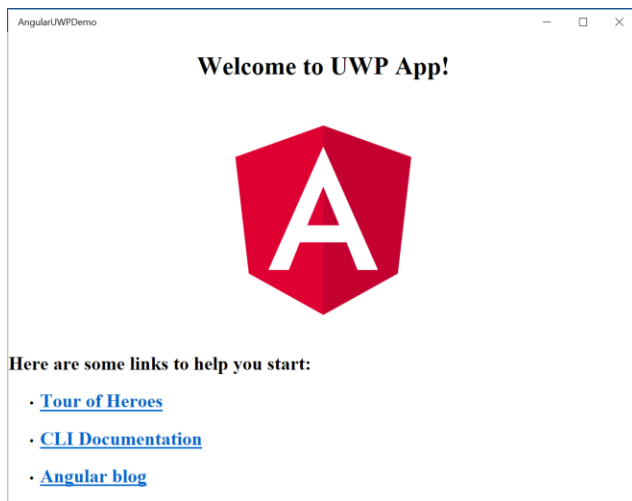
```
{  
  "$schema": "./node_modules/@angular/cli/lib/config/schema.json",  
  "project": {  
    "name": "angular-uwpdemo"  
  },  
  "apps": [  
    {  
      "root": "src",  
      "outDir": "dist",  
      "assets": [  
        "assets"  
      ],  
      "index": "./index.html",  
      "main": "main.ts",  
      "polyfills": "polyfills.ts",  
      "test": "test.ts",  
      "tsconfig": "tsconfig.app.json",  
      "testTsconfig": "tsconfig.spec.json",  
      "prefix": "app",  
      "styles": [  
        "./styles/styles.css"  
      ],  
      "scripts": [],  
      "environmentSource": "environments/environment.ts",  
      "environments": {  
        "dev": "environments/environment.ts",  
        "prod": "environments/environment.prod.ts"  
      }  
    }  
  ],  
  "e2e": {  
    "protractor": {  
      "config": "./protractor.conf.js"  
    }  
  },  
  "lint": [  
    {  
      "project": "src/tsconfig.app.json",  
      "exclude": "**/node_modules/**"  
    },  
    {  
      "project": "src/tsconfig.spec.json",  
      "exclude": "**/node_modules/**"  
    }  
  ]  
}
```

```
},  
{  
  "project": "e2e/tsconfig.e2e.json",  
  "exclude": "**/node_modules/**"  
}  
],  
"test": {  
  "karma": {  
    "config": "./karma.conf.js"  
  }  
},  
"defaults": {  
  "styleExt": "css",  
  "component": {}  
}  
}
```

Now that is complete, restart VS and reopen the project!

Go to your powershell and run the command `ng build`. After building is complete return to VS find the dist folder. Right-click and exclude it and then right-click and include it again. This will pickup all the files added by the ng build into the project.

In VS debug the application and you should get the below!



Updating the code leaves you with two choices. You can use VS Code and npm start to watch and make changes that show instantly the results via a webpage or make the changes in VS Community. Doing the latter will require a bit more work as you will have to use the command line with ng build to package the application for you. After packaging the application delete the extra script tags from the index.html file. There should only be 5 script tags as shown below. Make sure you exclude and include the dist folder before building the UWP application.

Make any edits to the src folder in the root document including adding your components. For beginners, please refer to angular.io to get started. The build will end up in the dist folder, so you should not edit any documents there directly.

Enjoy!

Robert Wilson

```
1 <body>
2   <app-root></app-root>
3   <script type="text/javascript" src="inline.bundle.js"></script>
4   <script type="text/javascript" src="polyfills.bundle.js"></script>
5   <script type="text/javascript" src="styles.bundle.js"></script>
6   <script type="text/javascript" src="vendor.bundle.js"></script>
7   <script type="text/javascript" src="main.bundle.js"></script>
8 </body>
9 </html>
```