Getting started with Angular 2 Application

## **Introduction**

In this guide, we'll build a Giphy search application to learn Angular 2's basics and some advanced skills.

For those who come from the Angular 1 world, I will draw comparisons throughout the tutorial, but you don't need to be familiar with Angular 1. However, you do need to be familiar with JavaScript to understand this tutorial. [There](https://www.codecademy.com/learn/javascript) [are](https://developer.mozilla.org/en-US/Learn/JavaScript) [plenty](https://www.codementor.io/learn-javascript-online) of resources online to get you started, even [free](https://github.com/getify/You-Dont-Know-JS) [books](http://jsbooks.revolunet.com/).

## **Prerequisites**

Make sure that the following tools are installed:

* [Node.js](https://nodejs.org/en/) - step by step [guide](https://hackhands.com/how-to-get-started-on-the-mean-stack/#installing-node-js) for both Windows and Mac
* [Git](https://git-scm.com/) - free interactive tutorial [here](https://try.github.io/)
* [Bower](https://bower.io/) - comprehensive tutorial [here](https://www.digitalocean.com/community/tutorials/how-to-manage-front-end-javascript-and-css-dependencies-with-bower-on-ubuntu-14-04)
* [Gulp](http://gulpjs.com/) - getting started tutorial [here](https://github.com/gulpjs/gulp/blob/master/docs/getting-started.md)

## **Transitioning from Angular 1?**

So, you've skimmed a few lines of this new shiny Angular 2.0 and you were left thinking, "Huh?" Trust me; I share your pain. When I first looked at it, I thought I was losing my grip. I was pretty comfortable with how Angular 1 did things and this seemed so different.

However, now a converted man, I'll tell you the same thing you've probably heard before: Angular 2.0 is much better than its predecessor.

Adjusting to a new technology is difficult, but it may prove necessary. Plus, once you become familiar with Angular 2, you won't remember why you didn't like it in the beginning.

There is a fast, easy method of migrating apps from Angular 1s. If you want to get into the details of how to migrate apps to Angular 2, start [*here*](https://angular.io/docs/ts/latest/guide/upgrade.html).

## **Demo App**

As I said in the Introduction paragraph, we're going to build an application for searching (and showing) gifs from the [Giphy](http://giphy.com/) website by using [their API](https://github.com/Giphy/GiphyAPI). In the end, we'll also deploy our app to [Github pages](https://pages.github.com/).

You can [try the app here](http://hitman666.github.io/giphy-search/) and you can fork the complete source code on [Github](https://github.com/Hitman666/GiphySearch_Angular2).

## **Angular CLI**

Setting up an Angular 2 application from the ground up can actually be a lot of work, in terms of installing and configuring settings. Folks at [Ember](http://emberjs.com/) had one great advantage: a tool called ember-cli. This tool bootstraps the application and helps with some common project operations and scaffolding.

Sure, we previously had [Yeoman](http://yeoman.io/) and its [Angular generators](https://github.com/yeoman/generator-angular), but they did not become widely-known conventions and they definitely weren't defacto tools like Ember.

Now, enter [Angular CLI](https://cli.angular.io/). Although this project is currently in beta, we can use it to:

* Create a scaffolded app with proper directory structure and included tests, with one simple command (ng new).
* Generate components, routes, services, and pipes. The CLI will also create simple test shells for all of these.
* Put your application in production (ng serve) easily.
* Deploy the app via GitHub Pages (ng github-pages:deploy).
* Run unit tests or end-to-end tests.
* Execute the official Angular2 linter (ng lint).

### **Installing angular-cli**

It's as simple as running:

npm install -g angular-cli

You can confirm that the installation went well if you run:

ng --help

You'll know that the installation went well because you'll get a bunch of output and help on various angular CLI commands.

Just for reference (in case you follow this guide at a later stage and something is not the same as I output it here), my version (ng --version) as of this writing is angular-cli: 1.0.0-beta.9.

## **Starting a New App with Angular CLI**

We'll call our app GiphySearch. Start a new app using angular-cli:

ng new GiphySearch

You should get an output similar to this:

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

29

30

31

32

33

34

35

36

37

38

39

40

41

42

43

44

45

# nikola in ~/DEV/Angular2 [13:57:55]

→ ng new GiphySearch

Could not start watchman; falling back to NodeWatcher for file system events.

Visit http://ember-cli.com/user-guide/#watchman for more info.

installing ng2

create .editorconfig

create README.md

create src/app/app.component.css

create src/app/app.component.html

create src/app/app.component.spec.ts

create src/app/app.component.ts

create src/app/environment.ts

create src/app/index.ts

create src/app/shared/index.ts

create src/favicon.ico

create src/index.html

create src/main.ts

create src/system-config.ts

create src/tsconfig.json

create src/typings.d.ts

create angular-cli-build.js

create angular-cli.json

create config/environment.dev.ts

create config/environment.js

create config/environment.prod.ts

create config/karma-test-shim.js

create config/karma.conf.js

create config/protractor.conf.js

create e2e/app.e2e-spec.ts

create e2e/app.po.ts

create e2e/tsconfig.json

create e2e/typings.d.ts

create .gitignore

create package.json

create public/.npmignore

create tslint.json

create typings.json

Successfully initialized git.

⠸ Installing packages for tooling via npm

├── es6-shim (ambient)

├── angular-protractor (ambient dev)

├── jasmine (ambient dev)

└── selenium-webdriver (ambient dev)

Installed packages for tooling via npm.

After this command finishes, let's cd into the project and run it:

1

2

cd GiphySearch

ng serve

bash

You should get:

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

# nikola in ~/DEV/Angular2/GiphySearch on git:master ● [14:05:03]

→ ng serve

Could not start watchman; falling back to NodeWatcher for file system events.

Visit http://ember-cli.com/user-guide/#watchman for more info.

Livereload server on http://localhost:49154

Serving on http://localhost:4200/

Build successful - 889ms.

Slowest Trees | Total

----------------------------------------------+---------------------

BroccoliTypeScriptCompiler | 484ms

vendor | 332ms

Slowest Trees (cumulative) | Total (avg)

----------------------------------------------+---------------------

BroccoliTypeScriptCompiler (1) | 484ms

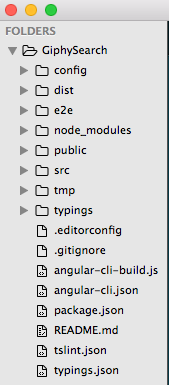
vendor (1) | 332ms

Your browser should show you the string app works! when you visit the app on the link: <http://localhost:4200/>.

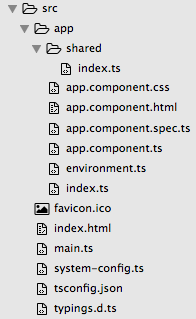
*In case you're curious about the 'Could not start watchman' output above. You can learn more about it*[*here*](https://ember-cli.com/user-guide/#watchman)*. In short, the article explains using brew install watchman if you're on Mac.*

## **Folder Structure**

Let's open this project in the editor of your choice (I'm using Sublime Text 3) and you should see something like this:



As I said, this is an introductory tutorial to get you running fast, so I won't be going into any specific details this time. Here we'll only focus on the src folder. The contents of that folder should be something like this:

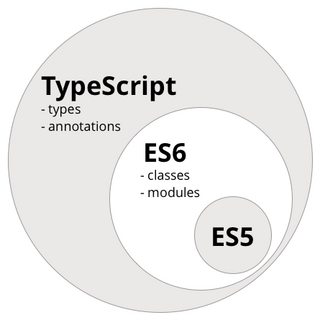


## **What Is This Typescript Thing?**

Now, you might be wondering, what are all these .ts files? These are TypeScript files and, even though you don't need to use TypeScript with Angular 2, almost everyone does.

So, what is TypeScript? On [their website](http://www.typescriptlang.org/) they state that: "TypeScript is a superset of JavaScript that compiles to clean JavaScript output."

You may have already seen this image before:



From this, we can see that this TypeScript contains ES6 (EcmaScript6), which then again contains ES5 (EcmaScript5), which yet again means that TypeScript contains ES5.

Since most browsers today can't run ES6 or yet alone TypeScript, we use the so-called [transpilers](https://en.wikipedia.org/wiki/Source-to-source_compiler) which turn our TypeScript or ES6 code into ES5 code which is a 'plain ol' JavaScript that you're probably most familiar with and that, basically, all of today's browsers understand.

Some of the features that TypeScript brings to the table are:

* [types](https://www.typescriptlang.org/docs/handbook/basic-types.html)
* [classes](https://www.typescriptlang.org/docs/handbook/classes.html)
* [decorators](https://www.typescriptlang.org/docs/handbook/decorators.html)
* [annotations](https://www.typescriptlang.org/docs/tutorial.html#type-annotations)
* [imports](https://www.typescriptlang.org/docs/handbook/module-resolution.html)
* [generics](https://www.typescriptlang.org/docs/handbook/generics.html)
* [...](https://www.typescriptlang.org/docs/tutorial.html)

I won't go into details here; you can learn about each of the above features by clicking on the links. However, let's take a look at a few things to get you started.

You now define variable as:

var myVar: string;

Notice that we defined the type of our variable using : string. This is new.

Similarly, you now have the ability to specify the return value type of the function:

1

2

3

function myFunc(msg: string): string {

return "I like to repeat what you said, therefore: " + msg;

}

ts

Just for reference, we have the following types:

* string
* number
* array
* enum
* any
* void

### **TypeScript Commands**

If you want to play with TypeScript on your own (in a separate project), you have to install it by running the following command:

npm install -g typescript

Then, you can write any TypeScript code, save it in a file, run it through the TypeScript compiler, and finally run the output of the TypeScript compiler with Node.

For a quick example, let's create a file named tsTest.ts and put the above function in it, along with a call of the function, such as console.log(myFunc('hello'));. Our file should look like this:

1

2

3

4

5

function myFunc(msg: string): string {

return "I like to repeat what you said, therefore: " + msg;

}

console.log(myFunc("hello"));

ts

Then run this file through the TypeScript compiler like this:

tsc tsTest.ts

Now you should see a tsTest.js file beside your tsTest.ts file and now you can run it with node like this:

node tsTest.js

As an output, you should get I like to repeat what you said, therefore: hello.

## **Conclusion**

In this guide, you learned how to get started with using Angular 2 and the components needed to create an application for searching Giphy's gifs by using their API.

In the following guide, [Building a Giphy Search Application in Angular 2](https://app.pluralsight.com/guides/building-giphy-search-application-angular-2) we'll go into more detail about building our application in Angular 2.