Setup for upgrading from AngularJS

> **Audience:** Use this guide **only** in the context of Upgrading from AngularJS or Upgrading for Performance. Those Upgrade guides refer to this Setup guide for information about using the deprecated QuickStart GitHub repository, which was created prior to the current Angular CLI.
>
> **For all other scenarios,** see the current instructions in Setting up the Local Environment and Workspace.

This guide describes how to develop locally on your own machine. Setting up a new project on your machine is quick and easy with the QuickStart seed on github.

Prerequisite: Make sure you have Node.js® and npm installed.

Clone

Perform the *clone-to-launch* steps with these terminal commands.

```
git clone https://github.com/angular/quickstart.git quickstart
cd quickstart
npm install
npm start
```

> `npm start` fails in *Bash for Windows* in versions earlier than the Creator's Update (April 2017).

Download

Download the QuickStart seed and unzip it into your project folder. Then perform the remaining steps with these terminal commands.

```
cd quickstart
npm install
npm start
```

> `npm start` fails in *Bash for Windows* in versions earlier than the Creator's Update (April 2017).

Delete *non-essential* files (optional)

You can quickly delete the *non-essential* files that concern testing and QuickStart repository maintenance (***including all git-related artifacts*** such as the `.git` folder and `.gitignore`!).

> Do this only in the beginning to avoid accidentally deleting your own tests and git setup!

Open a terminal window in the project folder and enter the following commands for your environment:

OS/X (bash)

```
xargs rm -rf < non-essential-files.osx.txt
rm src/app/*.spec*.ts
rm non-essential-files.osx.txt
```

Windows

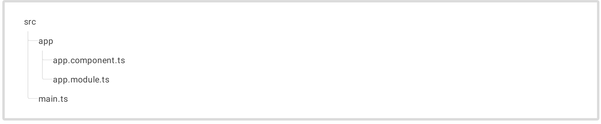
```
for /f %i in (non-essential-files.txt) do del %i /F /S /Q
rd .git /s /q
rd e2e /s /q
```

What's in the QuickStart seed?

The **QuickStart seed** provides a basic QuickStart playground application and other files necessary for local development. Consequently, there are many files in the project folder on your machine, most of which you can learn about later.

> **Reminder:** The "QuickStart seed" example was created prior to the Angular CLI, so there are some differences between what is described here and an Angular CLI application.

Focus on the following three TypeScript (`.ts`) files in the `/src` folder.



src/app/app.component.ts | src/app/app.module.ts | src/main.ts

```
import { Component } from '@angular/core';

@Component({
  selector: 'app-root',
  template: `<h1>Hello {{name}}</h1>`
})
export class AppComponent { name = 'Angular'; }
```

All guides and cookbooks have *at least these core files*. Each file has a distinct purpose and evolves independently as the application grows.

Files outside `src/` concern building, deploying, and testing your app. They include configuration files and external dependencies.

Files inside `src/` "belong" to your app. Add new Typescript, HTML and CSS files inside the `src/` directory, most of them inside `src/app`, unless told to do otherwise.

The following are all in `src/`

File	Purpose
`app/app.component.ts`	Defines the same `AppComponent` as the one in the QuickStart playground. It is the **root** component of what will become a tree of nested components as the application evolves.
`app/app.module.ts`	Defines `AppModule`, the root module that tells Angular how to assemble the application. When initially created, it declares only the `AppComponent`. Over time, you add more components to declare.
`main.ts`	Compiles the application with the JIT compiler and bootstraps the application's main module (`AppModule`) to run in the browser. The JIT compiler is a reasonable choice during the development of most projects and it's the only viable choice for a sample running in a *live-coding* environment such as Stackblitz. Alternative compilation, build, and deployment options are available.

Appendix: Develop locally with IE

If you develop angular locally with `ng serve`, a `websocket` connection is set up automatically between browser and local dev server, so when your code changes, the browser can automatically refresh.

In Windows, by default, one application can only have 6 websocket connections, MSDN WebSocket Settings. So when IE is refreshed (manually or automatically by `ng serve`), sometimes the websocket does not close properly. When websocket connections exceed the limitations, a `SecurityError` will be thrown. This error will not affect the angular application, you can just restart IE to clear this error, or modify the windows registry to update the limitations.

Appendix: Test using `fakeAsync()/async()`

If you use the `fakeAsync()/async()` helper function to run unit tests (for details, read the Testing guide), you need to import `zone.js/dist/zone-testing` in your test setup file.

> If you create project with `Angular/CLI`, it is already imported in `src/test.ts`.

And in the earlier versions of `Angular`, the following files were imported or added in your html file:

```
import 'zone.js/dist/long-stack-trace-zone';
import 'zone.js/dist/proxy';
import 'zone.js/dist/sync-test';
import 'zone.js/dist/jasmine-patch';
import 'zone.js/dist/async-test';
import 'zone.js/dist/fake-async-test';
```

You can still load those files separately, but the order is important, you must import `proxy` before `sync-test`, `async-test`, `fake-async-test` and `jasmine-patch`. And you also need to import `sync-test` before `jasmine-patch`, so it is recommended to just import `zone-testing` instead of loading those separated files.