Angular 2 is the great new framework that will provide us a great deal of flexibility and power when building our apps.

Goals

* Create an Angular 2 starter project with min requirements
* Understand what goes into setting up Angular 2

Let's take a high level view of what we'll be doing. The steps we'll take are as follows:

1. Setup our base project
2. Setup lite-server
3. Setup TypeScript
4. Grab packages and setup System.js
5. Build with Angular 2

The directory structure for our app will look like this:

|- app/

|- app.component.ts // main app component

|- app.module.ts // main app module

|- main.ts // bootstrap our app

|- index.html

|- package.json

|- tsconfig.json

|- systemjs.config.json

**NPM PACKAGES**

* Angular applications and Angular itself depend upon features and functionality provided by a variety of third-party packages. These packages are maintained and installed with the Node Package Manager ([npm](https://docs.npmjs.com/" \t "_blank)).
* All projects (which are also packages) need to be configured to work with NPM
* The command **npm init** is used to configure a project
* It will ask a series of questions, all of which have default answers, that are used to create and initialize a **package.json** file
* The **package.json** file contains metadata about the project, as well as, a list of application and development dependencies
* When NPM packages are installed, NPM will register them with the **package.json** file

Saving Package Dependencies

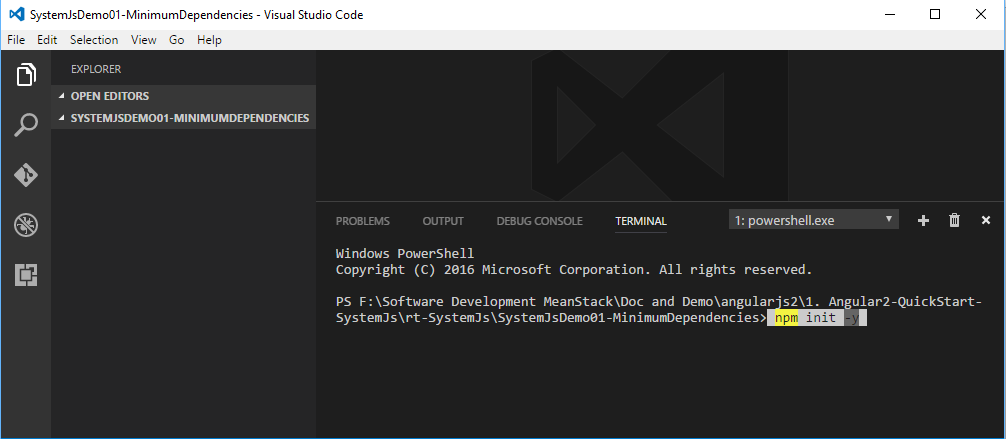
* Simply installing packages do not save the dependency in the **package.json** file
* In addition to installing, additional flags need to be specified:
  + **--save** or **-S** will save the package as an application dependency
  + **--save-dev** or **-D** will save the package a development dependency
* Application dependencies are used by the Node.js program when executing (common example would be Express)
* Development dependencies are used to develop the Node.js program (common example would be Grunt)

Terminal Commands

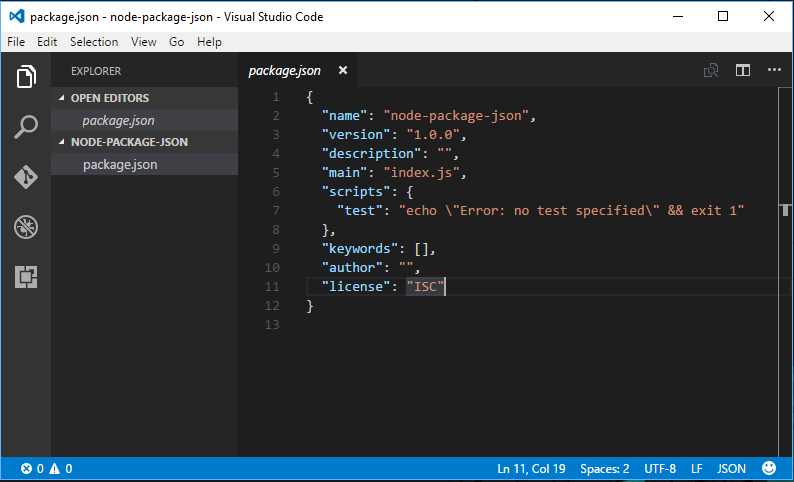
npm init –y

**Setting Up From Scratch**

* Create an empty folder
* Open it with visual studio code.
* Open integrated terminal
* Crate package.json



* The terminal commands, will produce a **package.json** file



{

"name": "node-package-json",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"test": "echo \"Error: no test specified\" && exit 1"

},

"keywords": [],

"author": "",

"license": "ISC"

}

* The file is a JSON file, and can be edited by hand
* Name is the name of the package
* Version follows the SEMVER scheme
* The version of each dependency is tracked as well
* Main is the main file imported when requiring the module

**Install lite-server**

[*lite-server*](https://www.npmjs.com/package/lite-server) : A light-weight, static file server, by [John Papa](http://johnpapa.net/) with excellent support for Angular apps that use routing.

Terminal Commands

npm install --save-dev lite-server

* It will create node\_modules folder.
* It will also update package.json file
* Create an npm script in package.json to start lite-server

"scripts": {

"lite": "lite-server"

}

Now after install lite-server and Create an npm script the updated package.json  file is

{

"name": "SystemJsDemo01-MinimumDependencies",

"version": "1.0.0",

"description": "",

"main": "index.js",

"scripts": {

"lite": "lite-server"

},

"keywords": [],

"author": "",

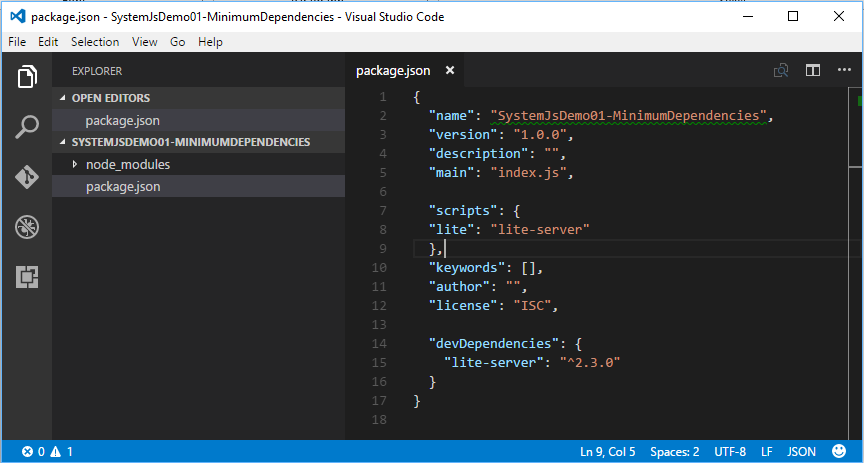
"license": "ISC",

"devDependencies": {

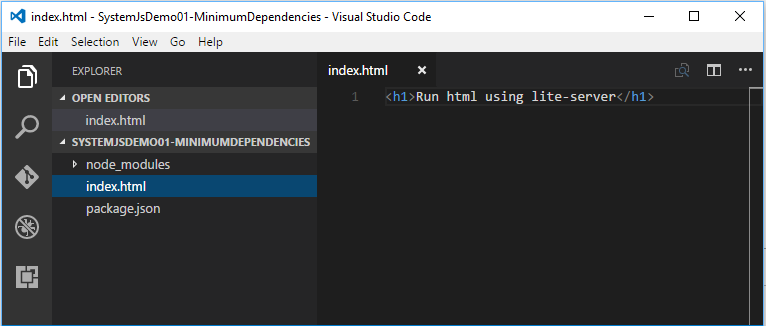
"lite-server": "^2.3.0"

}

}



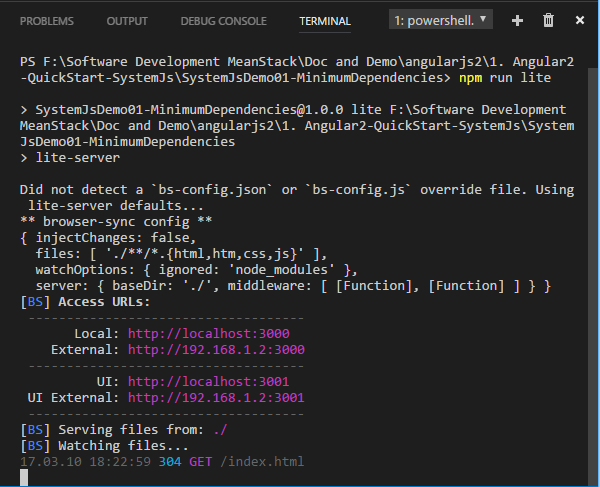
**Use npm command to run index.html file**



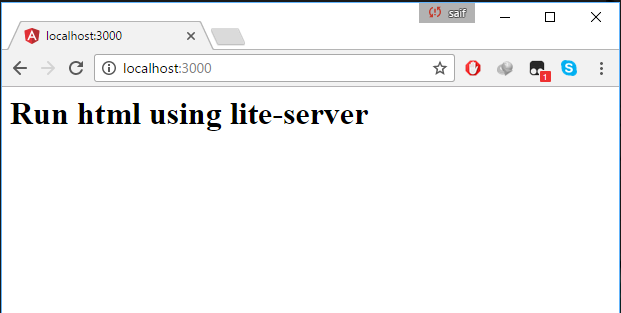
Create a new file named index.html

Terminal Commands to run lite server

npm run lite



You'll see your browser open with your index.html file.



Try to update and save the index.html and watch your browser reload like it's magic!

