**Angular**

**Getting Started**

**Components**

**Controller**

**Directive**

**Factory**

**Service**

**Filter**

**Angular**

<https://angularjs.org/>

Angular JS provides great functionality to a website by extending the functionality html can provide. it addresses the problem with html, which is making a webpage dynamic, and can do this by loading the required JavaScript files into the browser, this allows for less requests to the server, and makes use of the local browser evaluating the JavaScript creating a dynamic webpage.

Single page apps

<https://blog.codecentric.de/en/2014/08/angularjs-browserify/>

layout and module loading with browserify

load templates with browserify

<https://www.npmjs.com/package/grunt-angular-templates>

ng route

<https://docs.angularjs.org/error/$injector/modulerr?p0=myapp&p1=Error:%20%5B$injector:nomod%5D%20Module%20%27myapp%27%20is%20not%20available!%20You%20either%20misspelled%20the%20module%20name%20or%20forgot%20to%20load%20it.%20If%20registering%20a%20module%20ensure%20that%20you%20specify%20the%20dependencies%20as%20the%20second%20argument.%0Ahttp:%2F%2Ferrors.angularjs.org%2F1.6.1%2F$injector%2Fnomod%3Fp0%3Dmyapp%0A%20%20%20%20at%20http:%2F%2Flocalhost:3333%2Flibs.js:69:12%0A%20%20%20%20at%20http:%2F%2Flocalhost:3333%2Flibs.js:2184:17%0A%20%20%20%20at%20ensure%20(http:%2F%2Flocalhost:3333%2Flibs.js:2108:38)%0A%20%20%20%20at%20module%20(http:%2F%2Flocalhost:3333%2Flibs.js:2182:14)%0A%20%20%20%20at%20http:%2F%2Flocalhost:3333%2Flibs.js:4737:22%0A%20%20%20%20at%20forEach%20(http:%2F%2Flocalhost:3333%2Flibs.js:358:20)%0A%20%20%20%20at%20loadModules%20(http:%2F%2Flocalhost:3333%2Flibs.js:4721:5)%0A%20%20%20%20at%20createInjector%20(http:%2F%2Flocalhost:3333%2Flibs.js:4643:19)%0A%20%20%20%20at%20doBootstrap%20(http:%2F%2Flocalhost:3333%2Flibs.js:1839:20)%0A%20%20%20%20at%20bootstrap%20(http:%2F%2Flocalhost:3333%2Flibs.js:1860:12>

<https://docs.angularjs.org/api/ngRoute>

Node is server side JavaScript, and takes advantage of the browser doing some of the bulk of the work, instead of passing requests back and forward to the server. The JavaScript will be loaded along with the page and the browser will evaluate the functions and expressions. This allows for some powerful features, for example 2 way data binding and dynamic web pages.

**Get started:**

To start out we must include angular in our app

We can do this by installing it through Npm. Although Npm is the package manager for node, there are additional package managers, we are going to use bower, bower is a package manager for client side dependencies. <https://github.com/bower/bower>

We can install this by npm install bower -g

We are installing this module globally so we can use command line with it.



Then we must initialize bower, using the “bower init” command, this will create a bower.json, similar to the way we initialized npm and it created the package.json file for us.



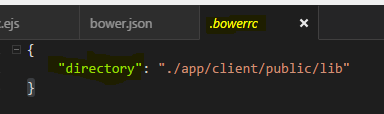
Again follow through with the setup, answering yes to them all will suffice. You will end up with something like so.



You may configure your bower with an additional file “.bowerrc”, here I am specifying what directory I wish for my dependencies to be installed into.

<https://bower.io/docs/config/>

I am installing my bower dependencies into my public folder in my app directory, as later we will use this folder to store all our files we wish to be made accessible in the browser.

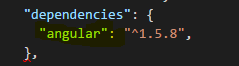


Now we are ready to start installing our client side dependencies.

Install Angular: “bower install angular –save”



This will result in adding angular to your bower.json, just like installing a dependacy with Npm (remember here NPM is for server side dependencies, while Bower is for Client side).



We must now add this angular.js file into index.egs, so when we launch our app it has angular loaded and we can use it then.

Open your index.ejs file and add this line of code. Insert this in the <head> section of our .egs file.



Now we are set up to do some angular work.

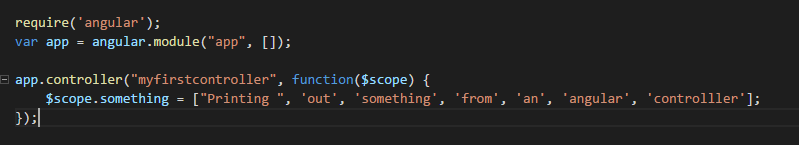
**Angular**

Creating a simple angular app and controller,

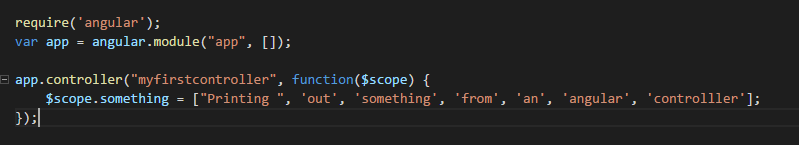
Creating the angular app module is simple and easy and can be done in 2 lines, although it doesn’t do anything, it is the entry point of the app.

Create an app.js file within the app/client/ folder

We must require angular, when this page is loaded into the browser, we have already specified in our index.egs that angular must be loaded with it, but in this JavaScript file we are telling it to look for the angular file.



Next are going to create a simple controller, where we will hard code in some values and in the browser we will use angular to display this array.



Next go back to the index.egs file and now we can add this file to be loaded

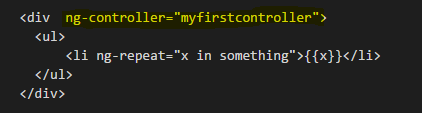


Now we have an index.js file that loads our app file and angular file we can now start doing things wit angular now in the. egs file

First we must declare what app we want, we called our app module “app” earlier now we are telling our body tag that everything inside it will be able to access this module.



In another tag we can call on our angular controller and inside the {{ }} we can write angular expressions. This will iterate through the array that the angular controller has initialized and print them out in the browser.



Angular

**Directives**

Templates with functions built in

**Controller / Service**

Busisness logic & fucntions

**services**

[**https://ilikekillnerds.com/2014/11/angularjs-call-controller-another-controller/**](https://ilikekillnerds.com/2014/11/angularjs-call-controller-another-controller/)

[**https://daveceddia.com/sharing-data-between-controllers-best-practice-use-a-service/**](https://daveceddia.com/sharing-data-between-controllers-best-practice-use-a-service/)[**https://docs.angularjs.org/guide/services**](https://docs.angularjs.org/guide/services)

**Factory**

Fetch data from server, creating data

**Filter**

Provide some sort of filtering functionality

Eg turn all the words first letter into uppercase or filter by some term.