Lab Setup Guilde

Intro

This guide will help you to install and configure your system to support a working lab environment for the AngularJS course. You will install multiple things; a web server, a software development tool. We will also be installing a few packages for the web server ahead of time. Throughout the week we will be installing more things via the node package manager and bower web package manager.

Software Installation

Some features of this course require the use of the Node.js web server. This server is programmed using JavaScript, making it particularly suited to a course that involves writing client side JavaScript. You will not be expected to write any supporting software for it; what you need will be provided. However, should you choose to do so, you will know the language that is needed.

In addition, this course assumes that you will use the Sublime Text 2 editor to write your web applications. This is not mandatory, but if you choose to use another environment, you must be able to support yourself in both configuring it and using it. The configuration is not complex however, so you should read these notes first, and then decide on the best course of action.

Installing Sublime Text 2

Assuming you will use Sublime Text as your exercise environment, but do not already have the software installed on your system, you will first need to download the tool. Sublime Text is available from http://www.sublimetext.com.

Click the **download** link at the top of the page. There you will see download information about Sublime Text 2 and Sublime Text 3 beta. This class will be utilizing Sublime Text 2. Click on the appropriate operating system link and your download will start. After it has downloaded simply run its installation process.

Node.js Installation

Normal Installation: Alternate instructions below

Next download a current distribution of Node.js, suitable for your host computer. Node.js is available from http://nodejs.org/. The homepage has a green **Install** button which will automagically determine the host platform you are running on and install the correct packages in most cases. After the download has finished simply follow the installers instructions. Make sure you have node added to the Windows Path.

Alternative Installation: If you have OSX follow below:

Homebrew allows you to easily install and manage packages with OSX. First visit their website http://brew.sh. Open up your terminal and run:

```
ruby -e "$(curl -fsSL https://raw.github.com/Homebrew/homebrew/go/install)"
```

OSX might be prompted to install **command line developer tools**. Go ahead and install them. It will take a couple minutes for that to complete. After the completion the terminal will prompt you to press any key to continue installing Homebrew. This will take

a couple more minutes. The terminal should indicate that **Installation successful!** After installation in the terminal run:

brew doctor

After running, the terminal should indicate **Your system is ready to brew**. Now you can proceed to install Node.js. It is always a good idea to run **brew update** before installing any packages, because you want to make sure you will be grabbing the latest and greatest package. Now you need to tell Homebrew to install node. In the terminal run:

brew install node

This will take a couple minutes as node installs. After the terminal opens up type **node -v** which will let you know that Node.js is installed and what version is installed.

Node.js Support Packages

Node Package Manager (npm) is used to install packages within Node.js. It is automatically installed when Node.js is installed to your machine. Package information can be found @ https://www.npmjs.org. Within this class we will utilize a couple of packages. With the installation of Node.js we have npm available to us. We will need to install support packages. In your console run the following command.

npm install express body-parser

* Possible windows error on running npm. You might get an error that begins “Error: ENOENT, stat …” This error means npm is expecting an “npm” directory to be present that is not in your file system. To fix the error, navigate to the the directory where “npm” should be (e.g. you will know by what is printed in the error) and create an empty “npm” directory. After creating the directory you will be able to add node modules.

Start Node.js and Verify the Installation

Finally, you should start Node.js and verify that the installation was successful. To start Node.js, use a command-line tool (cmd on windows, or a terminal in a Unix-like platform). In the accompanying **LabSetup** directory you will find a **server.js** file and a **WebContent** directory. The **server.js** is configuration for the Node.js server and the **WebContent** directory is where we are going to place our labs. Issue the command within the directory where the **server.js** file is located:

node server.js

You should see output something like this:

```
C:\application\express>node server.js
Node server started @ http://localhost:8080
Serving static files from C:\application\express
Press Ctrl + c for server termination
```

If you see the expected output, start your web browser and point it at:

http://localhost:8080/

If you see **node is working!!**, then your installation is successful as it is rendering the index.html page within the WebContent directory.

Bower Installation

Bower is a package manager for the web. It will allow us to fetch JavaScript libraries we will be using within our development.

npm install -g bower

* In order to use Bower we will need git installed:

If you are on a Mac it should already be installed for you or you can run:

brew update

brew install git

If you are on Windows you will need to download it. Go to http://git-scm.com and download the Windows version. Once downloaded run the installer. Use all the defaults except the defaults on the “Adjusting your PATH environment.” Change that to “Use Git from the Windows Command Prompt” or “Use Git and optional Unix tools from the Windows Command Prompt.” If you like using Unix style commands then choose the “optional Unix tools”. If you don’t care just set it to “Use Git from the Windows Command Prompt.” Then click through and finish the installer. Close your command prompt and then reopen it.

Jasmine Installation

Jasmine will be used for writing our Unit Tests. Install it via npm.

npm install jasmine

Karma Installation

Karma will be used for running our Jasmine unit tests. Install it via npm. We will grab packages for it during class for the specific project we will be working on. However, ahead of time you can download the Karma Command Line Interface (CLI)

npm install -g karma-cli

Protractor Installation

Protractor will be used for our End-to-End (e2e) testing of our Angular application. We will install it via npm.

npm install -g protractor

You can tell that it is installed properly by running:

protractor --version

Selenium Web Driver Installation

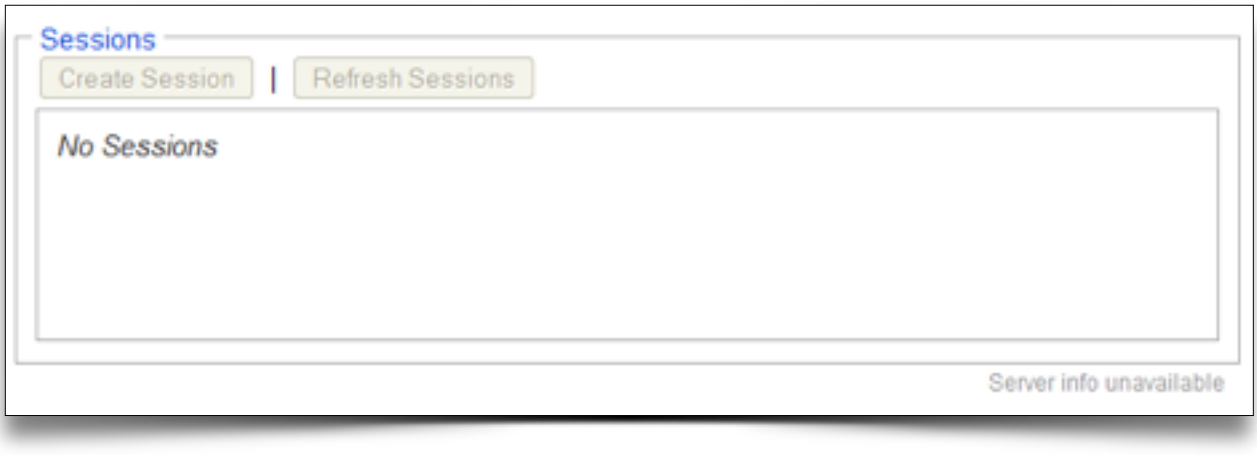
Protractor runs on top of Selenium Web Driver. We will need to make sure Web Driver is all set up and ready to go.

webdriver-manager update

update gets Web Driver all set up and **start** will make sure it runs appropriately.

webdriver-manager start

* You will need to have Java installed to run Selenium Web Driver. Go to http://java.com/en/ to download Java for your machine. Click “Free Java Download” and follow the instructions. After installing Java then you can run **webdriver-manager start**
* To make sure Web Driver is running correctly visit http://localhost:4444/wd/hub and you should see something that looks like the following:



Grunt Installation

Grunt is a task runner we will be utilizing. Install it via npm:

```
npm install -g grunt
```

Yeoman Installation

Yeoman is a web scaffolding tool. Install it via npm:

```
npm install -g yo
```