Introduction to MEAN.JS (Mark Scott)

# Setup: Download npm, nodejs and yeoman, MongoDB, grunt

## NVM and NodeJs

* Install ***nvm*** (Node Version Manager - <https://github.com/coreybutler/nvm-windows> )
* Install nodejs **0.10.36** 64 x64 via nvm
  + **nvm install** **0.10.36 *64*** (this installs the x64 version of nodejs)
  + **nvm use** 0.10.36 (this uses the specific nodejs version)
  + node version 0.10.36 64 is suggested as per Pluralsight “Getting set up” module (<https://nodejs.org/dist/v0.10.36/x64/> )

## SASS Install

When I grunt the app, it complains that SASS and Ruby are not installed.

I used the –force option to run the app (dir ***C:\Bob\dev\pluralsight\volunteer\volunteer*** ):

>**grunt --force**

npm install node-sass

## MondoDB 2.6.7

<https://www.mongodb.com/download-center#community>

<https://www.mongodb.org/dl/win32/x86_64> (**VERSION** **2.6.7** AS PER COURSE)

## Version Checking

npm –version

node –version

nvm list

yo –version

grunt –version

npm info generator-meanjs

## Bower/grunt/yo

***Install Bower:*** *-g**flag is global install*

npm install –g bower

**Install grunt** (or gulp, whichever you want)

npm install -g grunt-cli

**Install Yeoman:**

npm install -g yo

# MEAN - MongoDB/Express/Angular/Node

## Boilerplate Framework

* Started with newspaper matts, and boiling lead was poured into the matt to create a plate
* A block of code that is used over and over again
* MEAN is also an application scaffolding tool

## Code Generation from boilerplate (Advantages)

* Consistency
* Fewer Bugs
* Best Practices (supported by the community)
* Support (esp. for the bigger frameworks like MEAN)
* Saves time in starting your dev
* Uses ***Yeoman*** scaffolding tool

# Scaffolding the site

## Yo Meanjs Generator – see yeoman.io/generators

Go to c:\bob\dev\pluralsight\volunteer, and install the ***yo meanjs generator***

npm install -g generator-meanjs

## Generate application

yo meanjs

## Start MongoDB

* Start Mongo with default DB dir:

**mongod –dbpath** C:\Bob\dev\pluralsight\volunteer\volunteer\db

## Start the site

When I grunt the app, it complains that SASS and Ruby are not installed. I used the ***–force*** option anyway.

Cd C:\Bob\dev\pluralsight\volunteer\volunteer>

* **grunt debug --force**

# Tour the Scaffold Site

## Server-side controllers

..\Volunteer\volunteer\app

## Clients-side Angular controllers

C:\Bob\dev\pluralsight\volunteer\volunteer\**modules**\

…articles\client

…**core\client** – contains the ***‘home’ route*** in \config\core.client.routes.js

…\users\client

***User auth*** (a bit confusing. Why two versions ??? )

* C:\Bob\dev\pluralsight\volunteer\volunteer\node\_modules\grunt-protractor-coverage\public\modules\users\services
* C:\Bob\dev\pluralsight\volunteer\volunteer\modules\users\client\services\

### Menu module

..\volunteer\modules\core\client\services\menus.client.service.js

### Config and routes files: \volunteer\modules\articles\client\config\

* Articles.client.config.js sets up the nav menu items
* Uses ui-router

### Articles module

$scope.create / remove / update / find

C:\Bob\dev\pluralsight\volunteer\volunteer\modules\articles\client\controllers

C:\Bob\dev\pluralsight\volunteer\volunteer\modules\articles\client\services

Articles.client.service.js contains the ***Api end point***: .factory(**'Articles'**

***Register*** the ‘articles’ module 🡪 articles\client\articles.client.module.js

\*\*\* THIS IS INTERESTING \*\*\*

Modules are registered in the core section. See ‘***var ApplicationConfiguration’*** in ..\modules\core\client\app\**config.js**

### Main angular init file

C:\Bob\dev\pluralsight\volunteer\volunteer\modules\core\client\app\**init.js**

## Application Configuration

* **..\config\config.js** 
  + the app config files are loaded here
  + see **process.env.NODE\_ENV** checks to use either dev or prod
  + ‘development’ is chosen if no env is found - **process**.**env**.**NODE\_ENV** = **'development'**;
  + First, module dependencies are loaded (***var \_ = require***)
  + At the bottom of file, module.exports is invoked
    - see ***var initGlobalConfig*** to load assets and and config files.
    - See ..\config\assets\ and ..\config\env\
* See ***config\env\default.js***
  + Contains our app title, description, etc. that happened when we scaffolded the site with yo
* See **config\env\development.js**
  + Mongodb conn string is set here
* See **config\env\production.js**
  + Settings for our prod env module.exports = *initGlobalConfig*();
* Volunteer\**server.js** file kicks the whole thing off
  + Var server = app.start(); // see ..\config\lib\***app.js***
  + App.js 🡪
    - module.exports.start 🡪 **listen on <port>**
    - module.exports.init 🡪 ***Initialize Express***
  + Init models via ***module.exports.loadModels***
  + Routes are loaded in ***lib\express.js***

# Yo generator

## Generate a module (using yo sub-generator)

**ex/ generate a CRUD module**

yo meanjs:crud-module <module-name>

**ex/ generate a one-off express model**

yo meanjs:express-controller (ctrl-name)

**ex/ express route**

yo meanjs:express-route <route-name>

## Generate Angular modules

* yo meanjs:angular-module <module-name>
  + the sub-generator prompts you for details
* yo meanjs:angular-route <route-name>
  + it prompts you for the module it belongs to
* yo meanjs:angular-view <view-name>

# Demo (from Code Generation pluralsight module)

## Adding a new CRUD module to the volunteer app

yo meanjs:crud-module needs

* New module created **under ..\volunteer\module\needs**
  + client: contains menu config and routes
  + server: contains models, etc.
  + tests

# From Scaffold to Application

## Styling your application

* Layout server view 🡪 ***layout.server.view.html***
  + See ..\volunteer\modules\core\client\views
  + The ***SWIG engine*** loops thru css files in this file {% for css in cssfiles %}
* Css files collection found in ***..\config\assets\default.js***
  + Lib\***express.js*** pulls in the css files
    - app.**locals**.**cssFiles** = **config**.**files**.**client**.**css**;
  + the ..\config\config.js sets the globbed css files
    - config.files.client.css = ***getGlobbedPath***(assets.clientlib.css, …)
* Add new “Solid” Bootstrap theme from <https://www.bootstrapzero.com/bootstrap-template/solid>
  + Added new “***theme***” folder under ..\modules\
  + Added paths to config/assets/default.js
    - See the two css: sections (and the BM: comments)

## View and Models – notes

# Testing

## Grunt test tasks

See ..\volunteer\gruntfile.js 🡪 grunt.registerTask(**'test'**, [ … ] );

* 2nd argument is the array of tasks that we run
* i.e. grunt.registerTask(**'test:e2e'**, [**'env:test'**, **'lint'**, **'dropdb'**, **'server'**, **'protractor'**]);

See module.exports for the test definitions:

* see ***grunt.initConfig()***
  + also see the watch: section under here

See ..\config\assets\**test.js** and ..\config\env\test.js where the ‘tests’ paths are exported

Run tests from cmd prompt: ***grunt test --force***

Run production build: ***grunt prod –force***

* ***in grunt.js,*** see grunt.registerTask(**'prod' )**

aa