Gap Analysis

Ember.js

Nicole MacDougall – April 22nd, 2015

Summary:

I have learned a lot about Ember.js. For the most part, Ember was meant to be used to develop very large-scale web applications while using less code. It uses convention over configuration, which means that it's already assumed that you will be using JavaScript best practices and has allowed you to implement them in fewer lines of code. As this was explained in the tutorial, the speaker also gave reasons why Angular and Backbone were very different. There was a lot of emphasis given on the Ember core concepts, as this seemed to be where the learning curve hit a lot of people. These are templates, router, components, models, route, and controllers. While these were tricky, I feel that from what I have learned from the tutorial, I have a fairly good handle on Ember and would be able to create a site with it given the spare time.

The completed Lynda.com tutorial is located at https://github.com/nmacd85/EmberDemo Also on my github in the extras folder are my notes from the presentation and other tutorials.

Evidence:

I started with the todo tutorial TutsPlus (this basically copied the official ember.js tutorial) before deciding it was too in depth to use to teach the class. Below are my notes while working on that tutorial.

Installing Ember

Must have Node and NPM installed

1. npm install -g ember-cli
2. navigate to the dir you want ember new my-app
3. cd my-app
4. ember server
5. open in text editor
6. open app dir -> index.html
7. note the {{ … }} tags
    a. ember uses handlebars for templating http://handlebarsjs.com/
8. ember isn't a traditional MVC. You have templates that talk to your model. These know the other exists which is why ember will auto update the template when the model is modified.

- **Expressions**, like {{firstName}}, which take information from the template's model and put it into HTML.
- **Outlets**, which are placeholders for other templates. As users move around your app, different templates can be plugged into the outlet by the router. You can put outlets into your template using the {{outlet}} helper.
- **Components**, custom HTML elements that you can use to clean up repetitive templates or create reusable controls.

http://guides.emberjs.com/v1.11.0/concepts/core-concepts/

9. the router is basically your controller.

The **router** translates a URL into a series of nested templates, each backed by a model. As the templates or models being shown to the user change, Ember automatically keeps the URL in the browser's address bar up-to-date.

This means that, at any point, users are able to share the URL of your app. When someone clicks the link, they reliably see the same content as the original user.

A route is an object that tells the template which model it should display.

10. as angular has custom directives, ember has Components, custom HTML elements that you can use to clean up repetitive templates or create reusable

controls.

11. Install ember inspector in chrome, click allow access to file URLs
12. http://guides.emberjs.com/v1.10.0/getting-started/
13. todo app time
14. -----

http://code.tutsplus.com/tutorials/getting-into-ember-js--net-30709

dl ember starter kit

use a json api to get ember-data

naming conventions:

```
1    App.Router.map( function() {
2      this.resource( 'employees' );
3    });
```

I would then name my components, like this:

- **Route object:** *App.EmployeesRoute*
- **Controller:** *App.EmployeesController*
- **Model:** *App.Employee*
- **View:** *App.EmployeesView*
- **Template:** *employees*

After deciding this was too complex for what we needed, I decided to complete the Lynda.com tutorial on Ember.js since it would give me the knowledge I needed, as well as a tutorial I could go through with the class. Here are my notes, which are random but will be put together for a class sheet.

- JS Framework
    - discourse
    - square
    - yahoo ad managment built with ember

- convention over configuration
    - it was intended to build large apps but with less code
    - ember assumes it knows what you will need based on JS best practices (this is the convention part), so they have these blocks of best practices built for you, so you run those with less lines of JS code (like abstration)
    - ember is an opinionated framework, so they chose what you may need
- a little code, to create complex apps

- uses MVC-ish
    - also has router to list components in app
    - many refer to ember as MVC + R
    - when route is built, auto generates MVC that you need for later use.

- Ember generally uses SPAs
    - all "pages" contained in one file
    - when ember loads in browser file may be slow on 1st download but ember caches whole site this time so after this the site is quick

- Ember vs others (Angular, Backbone)
    - ember updates more often

    - Backbone is a library
        - configuration over convention (opposite of ember)
        - because of this backbone is more flexible to make your own configurations

    - Angular
        - similar in convention over config
        - have core concepts like ember which help build the app
        - Angular may be easier because of its directives

    - ember
        - more documentation to understand its abstration

Ember guides
    - read about routers and models

Ember API
    - review often
    - covers ember classes
    - a lot of info
    - start with components if self learning
    - so in-depth, its why people say its a steep learning curve

Ember Core concepts
    - Routing
    - Templates
    - Components
    - Models
    - Controllers

    - not using Ember.View, Handlebar templates mainly used because they're so powerful

Ember Data

Ember Changes
    - moving to HTMLBars

Steps for walkthrough:

- Based on ember starter kit from their site

- if you install ember inspector, make sure to allow access to URLs in the settings

- {{outlet}} is where other pages load

- if passing to html src, need to bind it

-metamorph if ember
    - uses to morph html from ember to what a browser can understand

-ember components based on web components
    - allow custom html elements
    - {{single-collection}} was our custom
    - web-components would be <single-collection>
    - ember working with w3c and developing web-components

https://css-tricks.com/modular-future-web-components/

- In MVC controller sends model data to View
- In Ember, controller sends data to templates
- Models not aware of templates
    - Makes more efficient
- ObjectController can send 1 piece of data or an ArrayController will send an array
- Nested routes are sub-routes
    - link within a route would be the link to a sub-route

- Will use getJSON() with jQuery's .then() to return promises
    - Async JS operations
    - .then will wait for JSON to load before anything else

- after adding route for single route, refresh. everythings gone, only happens on refresh or from bookmark
    - need to add a route for single exhibits to fix

- Ember Data used when you want to save to Local storage, like cookies
    - DS = Data Store

- DS.LSAdapter - Data Store Local Storage...

References:

- Up and Running with Ember.js (Lynda.com) – Kai Gittens
- Writing an Ember.js app from scratch – Drew Schrauf
- Official Ember.js site
    - Core Concepts
- Ember Inspector
- Handlebars.js
- Getting Into Ember.js (Rey Bango) – TutsPlus