Cutting Edge Web Dev

IT Academy, November 7th, 2013
College of Information Science and Technology

- Welcome!
- Who am I?

Zac Fowler
Director, IT Outreach
College of IS&T
University of Nebraska at Omaha
6001 Dodge St. PKI 383
Omaha, NE 68182
Office: (402) 554-6060
Email: zfowler@unomaha.edu

- Who are you?

What will we cover today?

Javascript

Look at 2 Frameworks

- Knockout.js
- AngularJS

Peek at other frameworks on the web…

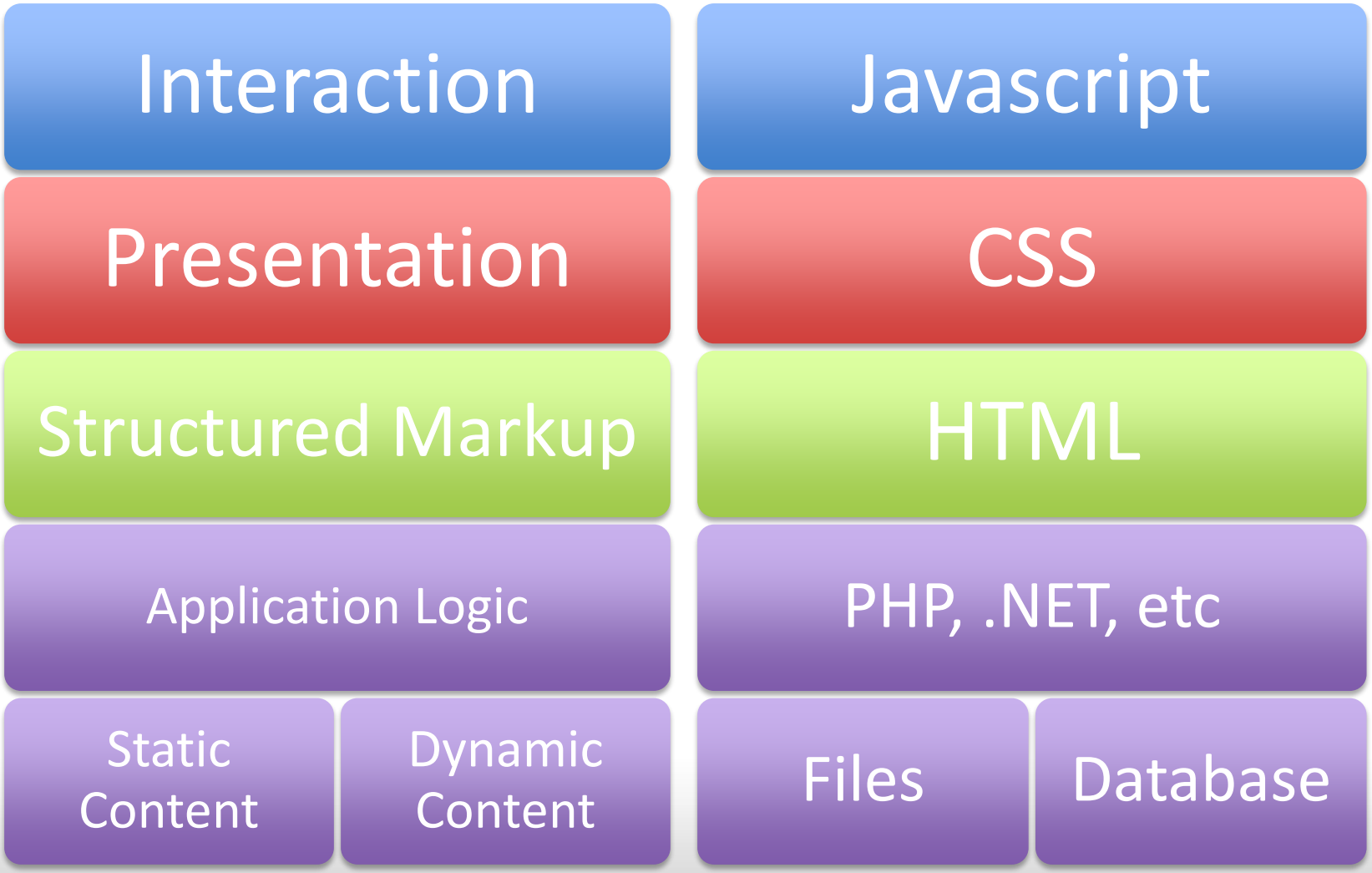
- What you should know:
  - HTML.
    - You know tags. You write tags. You *own those tags*.
  - Basic Javascript Syntax
    - Similar to java/php/c++
    - Variables aren't typed
    - Everything is an object
    - Lambda functions
  - Difference between server and client side code

- Object Literal Notation / JSON

```
var x = {
        someVar: 'Value',
        someVar2: 'Test',
        someArray: [0,1,3,5,10]
        someArrayOfObjects : [
                { id: 1, name: 'Zac' },
                { id: 2, name: 'Julie' }
        ]
}

x.someVar == 'Value'
x.someArray[2] == 3
x.someArrayOfObjects[1].name == 'Julie'
```

The Typical Web Stack



- Model
  - Objects that deal with business logic
  - Handles all persistent storage for data
  - Data layer
- View
  - Presentation of content to user
  - May handle limited interaction on page
  - Presentation/Interaction layer
- Controller
  - Application Logic
  - Connects user action with models and views
  - In request / response cycle, a controller typically will send a view as a result of each action

A Data Binding Framework

KNOCKOUT.JS

- Download our working files from

- https://github.com/zfowler-uno/it-academy-cutting-edge

- Knockout.js
  - A library that lets you *bind* objects in a view, or web page, to objects in Javascript
  - Model View ViewModel pattern
    - ViewModel handles model's representation on view
    - Updates to model reflect in view
    - Updates to view reflect in model
  - Does not replace other parts of application
    - Not a jQuery replacement
    - HTML via handlebars.js templates
    - Does not replace server-side scripting for data access or storage

- Download from knockoutjs.com
- Include in your web page

```
<script src="path/to/knockout.js"></script>
```

- Define a ViewModel
- Connect tags with observables

- Uses data-bind HTML5 attribute

- An observable is a ViewModel variable
- It connects a Javascript object to the UI

- Connect an observable via data-bind attribute on a tag
  - <h1 data-bind="text: title"></h1>

- One step deeper than *binding*
  - These are watched by Knockout.

Knockout.js – Hello World

- knockout / index.html

```
<h1>Hello World</h1>


<p>First name: <input data-bind="value: firstName" /></p>
<p>Last name: <input data-bind="value: lastName" /></p>
<h2>Hello, <span data-bind="text: fullName"> </span>!</h2>

<script src="lib/knockout-3.0.0.js"></script>
<script>
    // Here's my data model
    var ViewModel = function(first, last) {
        this.firstName = ko.observable(first);
        this.lastName = ko.observable(last);

        this.fullName = ko.computed(function() {
// Knockout tracks dependencies automatically. It knows that fullName depends
on firstName and lastName, because these get called when evaluating fullName.
            return this.firstName() + " " + this.lastName();
        }, this);
    };

    ko.applyBindings(new ViewModel("Planet", "Earth")); // This makes
Knockout get to work
</script>
```

- Build a template to display a Movie
- Create the ViewModel
- Assign observables
- Update with a form

Knockout.js - ViewModel

```
<!DOCTYPE html>
<html>
<head>
    <title>Knockout - Movie</title>
</head>
<body>
<h1>Movie</h1>
<div class="title" data-bind="text: title"></div>

<script src="lib/knockout-3.0.0.js"></script>
<script>
    // Here's my data model
    var MovieViewModel = function() {
        this.title = ko.observable("Karate Kid");
   };

    vm = new MovieViewModel();
    ko.applyBindings(vm); // This makes Knockout get to work
</script>
</body>
</html>
```

- That's a lot of work for each variable.
- Thankfully, there's a mapping tool
- Grab mapping tool from:
  - http://knockoutjs.com/documentation/plugins-mapping.html
- Load data from a JSON file, apply to template!

- In script tag

```
$.getJSON("karate-kid.json", function(data) {

        // Subsequently, set up the VM
   var viewModel = ko.mapping.fromJS(data);
   viewModel.loadUserData = function() {
       $.getJSON("karate-kid.json", function(data) {
           ko.mapping.fromJS(data, viewModel);
       });
   }
   ko.applyBindings(viewModel);
 });
```

CHECKPOINT

Another Framework

ANGULARJS

- From your friends at Google
- Another Javascript library
- Brings entire MVC/MVVM/MVW to your application
  - MVW – Model-View-"Whatever"
- Uses ng-* attributes

- Basic Angular Document

```
<!doctype html>
<html ng-app>
<head>
<script
src="https://ajax.googleapis.com/ajax/libs/angularjs/1.0.8/angular
.min.js"></script>
</head>
    <body>
    <div>
        <label>Name:</label>
        <input type="text" ng-model="yourName"
            placeholder="Enter a name here">
        <hr>
        <h1>Hello {{yourName}}!</h1>
    </div>
</body>
</html>
```

- It's not much of an application if it's all in one like that…
  - Let's structure one by dividing out some controller code.

- Create an app.js file.

```
var app = angular.module('MyTutorialApp',[]);

app.controller("MainController",
function($scope){
    $scope.inputValue = "";
});
```

- Create a basic controller

```
app.controller("MainController", function($scope){
    $scope.selectedPerson = 0;
    $scope.selectedGenre = null;
    $scope.people = []; // See file on github for data
});
```

- Modify the View

```
<div id='content' ng-app='MyTutorialApp' ng-
controller='MainController'>

<select ng-model='selectedPerson'
    ng-options='obj.name for obj in people'></select>

<select ng-model='selectedGenre'>
<option ng-repeat='label in
    people[selectedPerson.id].music'>
    {{label}}
</option>
</select>
</div>
```

- How about we add a filter for searching?
- New template
- Input box to bind the model
- Unordered List for the results
- Uses a pipe | and filter in ng-repeat

- Angular provides a bootstrap document to get started
- Includes the angular files and modules
- Includes a common folder structure for apps
- Includes unit and end to end testing tools

- Git clone, or download
  - https://github.com/angular/angular-seed
- Node.js
  - It comes with a node-ready webserver for testing
    - node scripts/web-server.js

- Let's add a controller file.
  - controllers.js

```
angular.module('myApp.controllers', []).
  controller('MovieList', ['$scope',
      function MovieList($scope) {
        $scope.variable = "test";
   }]);
```

- Link it to the app
  - app.js

```
angular.module('myApp', [
'myApp.controllers'
])
```

- Now add the variable to the view

  {{variable}}

Pretty cool!

- Basic Angular Document for our Movies

```
<!doctype html>
<html lang="en" ng-app="myApp">
<head>
  <meta charset="utf-8">
  <title>Google Phone Gallery</title>
  <link rel="stylesheet" href="css/app.css">
  <link rel="stylesheet" href="css/bootstrap.css">
  <script src="lib/angular/angular.js"></script>
  <script src="js/controllers.js"></script>
</head>
<body ng-controller="MovieList">

  <ul>
    <li ng-repeat="movie in movies">
      {{movie.movie_title}}
      <p>{{movie.synopsis}}</p>
    </li>
  </ul>

</body>
</html>
```

- Work time.
- Let's look at the seed app
  - Views.
  - Routing.
  - Partials.
- Update the detail view to include more content.

- Now let's add some AJAX/JSON requests to replace the movie

- $http is a new element we can access within the controller.
  - $http.get().success(function(data){});

- It requires an additional parameter in the function definition.

CHECKPOINT

Evaluating

FRAMEWORKS

Framework to Evaluate

- Determine what part of the stack the framework covers
  - Full stack?
  - Presentation?
  - Binding?
- Template Engine
  - Handlebars seems to be very popular
- Integration with your own code
  - Many of these frameworks are GREAT
  - Few are "drop in" replacements

- Let's take a look at a few.
  - ExpressJS
  - Ember.js
  - Silk.js
  - Node.js
  - Meteor*

QUESTIONS?

Thank you!

THE END