Modern Web by Design 2016-1

Bill Wolff

Agility Systems

Setup

# Tasks

* Install Web Essentials
* Configure two web sites

# Steps

1. Open Visual Studio 2015 (2013 or older should work)
2. File > New > Web Site
3. ASP.NET Empty Web Site
4. **Modern Web**
5. Insert your USB drive
6. Copy the **Modern Web 2016-1** folder to your documents\visual studio 2015\websites
7. Open a second instance of Visual Studio 2015
8. File > Open > Web Site
9. **Modern Web 2016-1**
10. Keep both sites open so you can use the finished version to copy and paste
11. Run some of the pages in the final version to see what we will build
    1. Right click the HTML file and select Set as Start Page
    2. Ctrl-F5 to run in your browser
    3. Or, open a page and do **Crtl-Shift-W**
    4. Try the pages with different browsers

Inning 1: HTML5 Semantics

# Key Concepts

* Document type
* Character encoding
* Semantic elements
* SVG and canvas
* Multimedia

# Tasks

* Create an HTML4 page
* Convert to HTML5
* Add semantic elements
* Add a scalable graphic
* Add some media

# Steps

1. Right click the **Modern Web** site and select Add > Add HTML Page
2. Enter the name **1-Html**
3. The page created uses the HTML5 doctype
4. Copy the markup from the demo **1-Html**.html
5. Study the HTML 4 markup at the top
6. Run this in several browsers
7. Use the browser developer tools (**F12**)to look at the markup, JS console, and timeline
8. Close the browser tab (**Ctrl-F4**)
9. Right click the **Modern Web** site and select Add > Add HTML Page
10. Enter the name **1-Html5**
11. Copy the markup from the demo **1-Html5**.html
12. Study the element names…this is called semantic layout
13. Run this in several browsers and use your developer tools
14. Right click the **Modern Web** site and select Add > HTML Page
15. Enter the name **1-Html5-SVG-Media**
16. Copy the markup from the demo **1-Html5-SVG-Media**.html
17. Review the canvas code, SVG markup, and video tag
18. Run this in several browsers

Inning 2: Tables + Floats + Flex

# Key Concepts

* ZenCoding (Emmet)
* Table elements
* Floating DIVs
* Fun with Flexbox

# Tasks

* Manual team list with tables
* Manual team list with floats
* Manual team list with flex
* Semantic page layout with flex

# Steps

1. Web Essentials must be installed for this to work
2. Add an HTML page called **2-team-table**.html
3. Select all markup and delete
4. Type **html:5** and click tab
5. Position after <body> and type **table>tr#idc$\*30>(td>img[onclick][height="60"][width="60"])+td+td+td** on a new line
6. Make a copy of this line, select the top one, highlight and make it a comment
7. Go to the end of the second line and click tab to expand the ZenCoding table
8. Copy the data from the data\team.json file to the <td> elements
9. Run in several browsers
10. Add an HTML page called **2-team-float**.html
11. Position after <body> and type **ul#team>li#id$\*30>div.league+div.division+(div.code>img[onclick][height="60"][width="60"])+div.location+div.team+div.stadium** on a new line
12. Make a copy of this line, select the top one, highlight and make it a comment
13. Go to the end of the second line and click tab to expand the ZenCoding float layout
14. Copy the data from the data\team.json file to the <div> elements
15. Run in several browsers…not very useful
16. Copy the style block form the finished site
17. Run in several browsers
18. Add an HTML page called **2-team-flex**.html
19. Replace the body tag with the structure from **2-team-float**.html
20. Copy the **grid layout** style block form the demo site **2-team-flex**.html
21. Run to see how flex can work with tabular data
22. Uncomment some of the CSS to see special effects
23. Copy the header, footer, nav, section, and content elements from the demo site
24. Copy the **page layout** style block form the demo site
25. Run in several browsers to see an application style layout with content scrolling

Inning 3: JavaScript + jQuery

# Key Concepts

* SEAF
* Geolocation
* JavaScript for current location
* Call function to get distance
* jQuery framework
* JSON data
* Datatables plug-in

# Tasks

* Explain Self Executing Anonymous Functions
* Add GeoLocation
* Add jQuery to a page
* Review JSON player data
* Load JSON via jQuery
* Bind JSON array to datatable

# Steps

1. Copy the data folder from the demo site to the **Modern Web** site
2. Refresh the Solution Explorer to see the files
3. Open each of the files to see the data
4. Add an HTML page called **3-player**.html
5. Copy the SEAF script block from the demo page
6. Copy the header inside the body from the demo page
7. Copy the style block from the demo page header
8. Run in several browsers, you will be prompted to accept the location call
9. Copy the link and script elements from the demo page header
10. Make sure the files in <link> and <script> exist in your project
11. Copy the player data table definition from the demo page
12. Copy the jQuery script block from the demo page
    1. HTML markup defines the header and footer columns
    2. The id is used by jQuery to place the data table
    3. Document ready runs once the DOM is built
    4. getJSON reads the data file from disk and makes an array in memory
13. Run this in several browsers
    1. Verify that jQuery and datatables are loaded
14. Experiment with sorting, paging, and searching

Inning 4: CSS3 + Responsive Design

# Key Concepts

* Stylesheets
* Selectors
* Local storage
* CDN
* Viewport
* Grids
* Media queries
* CSS Frameworks

# Tasks

* Alternate colors on team rows with css selectors
* Save favorite team to Local Storage
* Highlight favorite team using static style rule
* Highlight favorite team dynamically using jQuery
* Add w3.css for basic responsive design
* Add Bootstrap for a popular responsive design framework
* Team list in semi-responsive grid

# Steps

1. Copy the **2-team-flex**.html to **4-team-css**.html
2. Set the ID of each <li> to the code for the team
3. Hand edit the <style> rule for your favorite team and test
4. Run in several browsers
5. Comment out the favorite team style rule
6. Add the fave style block from the demo page
7. Add the setFave function to the <script> area
8. Edit the click event of each image to call setFave(‘team code’)
9. Run and test with the Console active (F12 in IE, F12 or Ctrl-Shift-J in Chrome)
10. Look at the local storage to see the saved value (replaces cookies)
11. Add an HTML page called **4-w3**.html
12. Copy the markup from the demo site
    1. This loads W3.css from a CDN
    2. Optionally replace the <link> with a local copy
    3. Review the viewport and row/column classes
13. Run and test in various browser sizes
14. Add an HTML page called **4-bootstrap**.html
15. Copy the markup from the demo site
    1. This loads Bootstrap.css from a CDN
    2. Optionally replace the <link> with a local copy
    3. Review the viewport and row/column classes
16. Run and test in various browser sizes
17. Open style/bootstrap.css
    1. Search for @media to see the media queries
18. Copy **3-player**.html to **4-player-bootstrap**.html
19. Change the link and script tags in the header to use local copies
20. Copy the nav section from the demo page
21. Copy the fonts folder form the demo site
22. Run and test in various browser sizes
    1. BootStrap uses Gylphicon fonts for the sorting indicators

Inning 5: Bootstrap

# Key Concepts

* Nav elements
* Burger menus
* Drop downs
* Glyphs
* Carousel
* Modal dialog
* Code free interactivity

# Tasks

* Create menu for several pages
* Use mobility burger approach
* Use glyphs for menu and news items
* News items in a rotating carousel
* User registration in a modal dialog

# Steps

1. Copy the **4-player-bootstrap**.html to **5-player-bootstrap**.html
2. Edit the navbar menu so the <a> tag hrefs point to the 3 new pages form this inning
3. Copy the **5-player-bootstrap**.html to **5-team-bootstrap**.html
4. Edit the jumbotron <h1> to MLB Teams
5. Copy the **5-player-bootstrap**.html to **5-news-bootstrap**.html
6. Edit the jumbotron <h1> to MLB News
7. Fix it so content scrolls underneath by using navbar-fixed-top in the navbar <div>
8. Test in various browsers and sizes
   1. URL will change, but content is still the same in each page
   2. Content should scroll under the menu
   3. Very small browser size will make the burger menu appear
9. Edit each file to load the correct table
   1. Copy from previous work!
10. For 5-news-bootstrap.html, copy the container <div> from the demo page
    1. The <b> tags host the glyph font symbols
    2. Rows and columns are created using Bootstrap class names
11. Copy the glyph style block from the demo page
    1. Glyphs are sized and colored like any other font
12. Make news the default page
13. Run and test in different browsers and sizes
14. Add the carousel div from the demo page
    1. Items are placed in a series of wrapping containers
    2. There are navigation bullets in the <ol> tag
    3. There are slider controls in the <a> tags
15. Copy the carousel style block from the demo page
16. Run and test in browsers
17. Add the modal form <div> from the demo page
    1. Modals have a header, body, and footer
    2. The registration menu item calls the modal without code
    3. For extra credit, copy the modal and registration link to the other pages
18. Run and test in browsers

Inning 6: AngularJS

# Key Concepts

* Directives
* Controllers
* Data binding
* Service
* JavaScript Promises
* ng-repeat
* ng-click

# Tasks

* Start a basic Angular team page
* Bind the JSON team list in a responsive grid
* Use ng-click to toggle views
* Bind the JSON player list in a grid with an Angular directive
* Graph the JSON team list with an Angular directive

# Steps

1. Copy the **4-team-css**.html to **6-angular**.html
2. Remove the jQuery script tag in the header
3. Add an angular script tag
4. Comment out or remove the code in the <script> section at the bottom of the page
5. Run and test in browsers
   1. Verify that Angular is loaded
6. Remove all but one of the <li> tags in the team list
7. Add ng-app and ng-controller attributes to the body tag
8. Run and test in browsers
   1. Check for errors in the developer console
9. Add a function to define the app and controller in the script block
   1. Create one variable so we can display some data
10. Run and test in browsers
    1. This is the angular entry point, it is loaded and ready for code and markup changes
11. Experiment with some markup changes in the <content> tag above the team <ul>
    1. Data binding is done with the {{}} syntax and applies to many types of tags
    2. **{{**now**}}**
    3. **{{**now | date:'medium'**}}**
    4. **{{**now | date:'fullDate'**}}**
    5. <input ng-model="name"/> **{{**name**}}**
12. Load some JSON data using angular $http injection
    1. **{{**teams.length**}}**
    2. **{{**teams | json**}}**
    3. console.log(JSON.stringify($scope.teams));
13. Add ng-repeat to the <li> tag and use data binding
    1. ng-repeat="team **in** teams"
    2. **{{**team.Lg**}}**
14. Copy the **6-angular**.html to **6-angular-view**.html
15. Add a $scope variable to track the current view
16. Make the <content> tag switchable and add a <div> for each view
    1. ng-switch="view"
    2. ng-switch-when="teams"
17. Edit the <nav> buttons and call a function called setView()
18. Add the setView function in the <script> block
19. Run and test in browsers
    1. Views switch without a page load
20. Add the angular-datatables links to the header, copy from the demo
21. Add an http get to the <script> block for the player data, copy from the demo
22. Add the players view from the demo
    1. This loads the same datatables utility but in Angular fashion
    2. The :: means one way binding which greatly increases speed
23. Run and test in browsers
24. Add the angular-chart links to the header, copy from the demo
25. Add the capacity view from the demo
26. Copy the remaining script block code from the demo
    1. This loads the chart data and labels in a for loop
27. Run and test in browsers

Inning 7: AngularJS + CSS3 + Fonts

# Key Concepts

* ng-cloak
* ng-class
* ng-disabled
* Font Awesome badges

# Tasks

* Add favorite team to controller
* Select favorite team from the team list
* Create class functions to highlight favorite team
* Add a player list page and highlight the favorite team
* Add angular-fontawesome for likes

# Steps

1. Copy the **6-angular-view**.html to **7-angular-css**.html
2. Add ng-click to the repeating team <li>
3. Call and create the routine setFave()
   1. ng-click="setFave(team)"
4. Add a pointer cursor to the <li> style
5. Remove the cursor pointer for #team div.code
6. Add code to track the fave in localStorage
   1. Refer to **4-team-css**.html
7. Run and test in browsers
   1. Confirm that localstorage is persisting your selection
8. Highlight the favorite team with ng-class
   1. ng-class="{'fave':team.Code==fave}"
9. Default the favorite to localstorage
   1. $scope.fave = localStorage.getItem('fave');
10. Add a new menu item and view for Pick Players
    1. Also disable the current button
    2. ng-disabled="view=='myteam'"
11. Copy the team loop and change the variables
12. Copy some styles in the <style> block
13. Add a header block for the list
14. Run and test in browsers
    1. Confirm that favorite team is highlighted for each player
15. Add angular-fontawesome support to the header
16. Add picardy-fontawesome to the module injection
17. Add a conditional column in the team list
    1. <fa ng-show="team.Code==fave" name="thumbs-o-up" size="2"></fa>
18. Run and test in browsers

Inning 8: AngularJS Functions + Modules

# Key Concepts

* Angular functions
* Directives
* Filters
* Services
* Modules

# Tasks

* Move data calls to a service
* Build MyTeam page as directive
* Add function for distance to team
* Add filters to MyTeam for positions
* Separate code into multiple files

# Steps

1. Copy the **7-angular-css**.html to **8-angular-function**.html
2. Create a directive called **myteam** inthe script block
   1. This only allows E (element) mode
   2. Add this directive to the myteam view
3. Add myteam styles to the <style> layout section
4. Add an HTML page called **8-myteam**.html
5. Copy the markup from the demo page
6. Add localstorage for the favorite team
7. Add geolocation calls for the latitude and longitude
8. Add a distance function to compute miles to a given stadium
9. Run and test in browsers
10. Add a service for data access
    1. $http is injected in to the service
    2. Promises are used for asynchronous calls via $q and defer, resolve, and reject
11. Change the controller to use the service for data access
12. Add player repeat loops with filters
    1. Some use ng-if
    2. Some use filter fields
    3. One uses a filter function
13. Run and test in browsers
14. Copy the **8-angular-function**.html to **8-angular-module**.html
15. Create a JavaScript file called **8-app**.js
16. Create a JavaScript file called **8-controllers**.js
17. Move the <script> block code into these files
18. Add references at the top of the page pointing to the new JS files
19. Create a stylesheet called **8-site**.css
20. Move the <style> sections to the new file
    1. Edit the comments to proper CSS format
21. Add references at the top of the page pointing to the new CSS file
22. Run and test in browsers

Inning 9: AngularJS SPA + Templates

# Key Concepts

* Single Page Application
* Route
* Templates

# Tasks

* Move views into templates
* Create routes
* Bind route to multiple views
* Create multiple controllers

# Steps

1. Copy the **8-angular-module**.html to **9-angular-route**.html
2. Copy the code from the demo copy
   1. ng-view replaces ng-switch
   2. <nav> buttons are replaced by <a> links with hash urls
   3. Remove the views
3. Copy the **8-site**.css to **9-site**.css
4. Copy the code from the demo copy
5. Copy the **8-app**.js to **9-app**.js
6. Copy the route code from the demo copy
7. Copy the **8-controllers**.js to **9-controllers**.js
8. Copy the code from the demo copy
9. Copy the **8-site**.css to **9-site**.css
10. Create an Html page called **9-myteam**.html
11. Copy the code from the demo copy
12. Create an Html page called **9-teams**.html
13. Copy the code from the demo copy
14. Create an Html page called **9-players**.html
15. Copy the code from the demo copy
16. Create an Html page called **9-capacity**.html
17. Copy the code from the demo copy
18. Create an Html page called **9-pick**.html
19. Copy the code from the demo copy
20. Run and test in browsers
    1. Url now changes with each view
    2. Browser back button and history work
    3. Search engines can search the links
    4. Links can be shared