.NET Core: Developing Cross-Platform Web Apps with ASP.NET Core – Workshop*PLUS*

Wael Kdouh - @waelkdouh

Senior Consultant

v3.0

Conditions and Terms of Use

Microsoft Confidential

This training package is proprietary and confidential, and is intended only for uses described in the training materials. Content and software is provided to you under a Non-Disclosure Agreement and cannot be distributed. Copying or disclosing all or any portion of the content and/or software included in such packages is strictly prohibited.

The contents of this package are for informational and training purposes only and are provided "as is" without warranty of any kind, whether express or implied, including but not limited to the implied warranties of merchantability, fitness for a particular purpose, and non-infringement.

Training package content, including URLs and other Internet Web site references, is subject to change without notice. Because Microsoft must respond to changing market conditions, the content should not be interpreted to be a commitment on the part of Microsoft, and Microsoft cannot guarantee the accuracy of any information presented after the date of publication. Unless otherwise noted, the companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted herein are fictitious, and no association with any real company, organization, product, domain name, e-mail address, logo, person, place, or event is intended or should be inferred.

Copyright and Trademarks

Microsoft may have patents, patent applications, trademarks, copyrights, or other intellectual property rights covering subject matter in this document. Except as expressly provided in written license agreement from Microsoft, the furnishing of this document does not give you any license to these patents, trademarks, copyrights, or other intellectual property.

Complying with all applicable copyright laws is the responsibility of the user. Without limiting the rights under copyright, no part of this document may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means (electronic, mechanical, photocopying, recording, or otherwise), or for any purpose, without the express written permission of Microsoft Corporation.

For more information, see Use of Microsoft Copyrighted Content at

http://www.microsoft.com/en-us/legal/intellectualproperty/permissions/default.aspx

ActiveX, Internet Explorer, Microsoft, Microsoft Corporate Logo, SQL Server, Silverlight, and Visual Studio are either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. Other Microsoft products mentioned herein may be either registered trademarks or trademarks of Microsoft Corporation in the United States and/or other countries. All other trademarks are property of their respective owners.

How to View This Presentation

- To switch to **Notes Page** view:
  - On the ribbon, click the **View** tab, and then click **Notes Page**

- To navigate through notes, use the Page Up and Page Down keys
  - Zoom in or zoom out, if required

- In the **Notes Page** view, you can:
  - Read any supporting text—now or after the delivery
  - Add notes to your copy of the presentation, if required

- Take the presentation files home with you

Module 6: Client-Side Development

Module Overview

Module 6: Client-Side Development

Section 1: ASP.NET MVC and JavaScript

Lesson: Project Templates

Why Use JavaScript in ASP.NET MVC?

- Combines server-side and client-side processing
- More responsive web applications
- JavaScript comes in many forms:
  - AJAX – Partial page update and refresh
  - Jquery – Elegantly and Efficiently find and manipulate HTML DOM elements
  - MooTools – Modular JavaScript and code reuse
  - Prototype – Simplify development of dynamic web application
  - Node.js – Developing high performance JavaScript using multithreading model
  - Industry standard for modern web development, etc.

Project Template – wwwroot Folder

- All static files should be located in this folder (JavaScripts, CSS, images or HTML files)
- wwwroot folder is the root of the website
  - http://hostname/ points to wwwroot
  - All static content should be relative to the folder
- Code files should be placed outside of wwwroot (C# or Razor)
- Static files created through compilation or pre-processing should be copied into wwwroot
- Can be renamed by changing “webroot” setting in the *project.json* file

Project Template: JavaScript Libraries Part 1

- _references.js
  - Contains JavaScript reference in the form of comments
  - Allows Microsoft Visual Studio to augment IntelliSense support for JavaScript
  - The autosync flag set to true:
    - JavaScript files are automatically added
    - Automatic update when a referenced file is moved
  - Manual update is possible by right-clicking the file
- Bootstrap.js and bootstrap.min.js
  - HTML, CSS, and JavaScript-based design templates for creating responsive websites

Note: Always use the .min version of a JavaScript file to improve load time

Project Template: JavaScript Libraries Part 2

- Bootstrap-touch-carousel and hammer.js
  - A slideshow component for cycling through elements, like a carousel
  - Enable gestures on touch devices using hammer.js, a JavaScript library for multi-touch gestures

Project Template: JavaScript Libraries Part 3

- Jquery-version.intellisense.js
  - Extending IntelliSense for jQuery library
- Jquery-version.js and jquery-version.min.js
  - Main JQuery version
- Jquery-version.min.map
  - Allows to map to the un-minified version of JQuery for troubleshooting purposes
- Jquery.validate.js (Jquery.validate.min.js)
  - Provide client side validation using jQuery
  - Multilanguage support
- Jquery.validate.unobtrusive.js and jquery.validate.unobtrusive.min.js

Client-Side Development Configuration Files

- gruntfile.js
  - JavaScript configuration of Grunt tasks
- gulpfile.js
  - JavaScript configuration of Gulp tasks
- Project.json
  - Main project file. NuGet package dependencies are listed here
- Package.json
  - Lists npm packages
- Bower.json
  - Lists Bower packages

Demo: Default JavaScript Libraries

Module 6: Client-Side Development

Section 2: VS Client-side Dev Tooling

Lesson: Bower and Gulp

Why Use Gulp (or Grunt) and Bower?

- Modern web applications incorporate various and rich client-side libraries, such as jQuery, TypeScript, Bootstrap etc.
- Easy management of client-side packages
- Automating build tasks such as scripts compilation, bundling, minification or unit testing
- Use the existing tools from the web development community

What Is Bower?

- “A package manager for the web” (http://bower.io)
- Installs and restores client-side libraries
- Keeps track of all the packages in a manifest file, bower.json
- Improves page load

What is Gulp?

- “The JavaScript Task Runner” (http://gulpjs.com)
- An application to automate routine client-side development tasks (compilation, bundling, minification, unit testing, etc.)
- gulpfile.js contains Gulp tasks with JavaScript-like configuration
- Grunt is another task runner
    - Gulpfile.js uses JSON-like syntax for configuration

Bundling and Minification

- Bundling reduces the number of requests to the server
  - Combining multiple files into a single file
  - Create CSS, JavaScript and other bundles
  - Use the “Include” or “IncludeDirectory” of the Bundle Class
- Minification reduces the size of the requested assets (CSS and JavaScript)
- ASP.NET Core MVC uses Gulp or Grunt to achieve bundling and minification

Demo: Bower and Gulp

Module 6: Client-Side Development

Section 3: Development Techniques

Lesson: JavaScript and jQuery

Using JavaScript in MVC

- JavaScript scripts can be defined inside a View using the html script tag like in a html page
- Use the MVC @section tag to organize JavaScript scripts
  - The @RenderSection is used to inject JavaScript at a desired location inside the View
- For best practices, declare JavaScript scripts inside a .js file
- Use a minification tool in Visual Studio for optimization
- IntelliSense support for JavaScript in Visual Studio

jQuery and Microsoft

- Lightweight open source JavaScript library
- Deprecated Microsoft.Ajax libraries in favor of jQuery
- Distributed jQuery library with Visual Studio projects since 2008
- Extended Microsoft product support for jQuery
  - Enterprises can open jQuery support cases 24x7 with Microsoft Support
- Integrated Client template support
- Default templates use jQuery

jQuery

- Reduces client-side coding
- CSS 3-based syntax for traversing and manipulating DOM
- Concise wrappers for Ajax calls
- Abstracted to eliminate cross-browser differences
- Unobtrusive client validation
- XPath selectors to access elements in the DOM
- Elements are retrieved in the form of jQuery objects
- Start with jquery(), jquery., $() or $. to use jQuery

jQuery: Selectors

- Execute commands on a single or multiple selected DOM elements
- Basic types of selectors:
  - Based on HTML elements IDs. For example, $("#main")
  - Based on cascading style sheets (CSS). For example, $(".header")
  - Based on element tags. For example, $("div")
  - Based on element attributes. For example, $("[type = 'button']")
- Build more complex selectors through combination
  - $("#main p.quote") ⇔ Select paragraphs with a "quote" CSS class located inside elements with IDs equal to "main"
- $(this) operator

jQuery: Selectors (continued)

- Specific operators are used to expand selection options
  - A white space selects all elements that are descendants of the given ancestor. For example, $(“div p”) ⇔ $(“div”).find(“p”)
  - The > operator selects direct child elements of the given ancestor. For example, $(“div > p”) ⇔ $(“div”).children(“p”)
  - The + operator selects adjacent elements. For example, $(“div + p”) ⇔ $(“div”).next(“p”)
  - The ~ operator selects all siblings elements. For example, $(“div ~ p”) ⇔ $(“div”).nextall(“p”)
  - The comma operator selects all the specified elements. For example, $(div, p, a)

jQuery: Filters

- Used with Selectors, or alone
  - **Positional filters** :first, :even, :eq(index), :gt(index), :not (selector) etc.
  - **Child filters** :nth-child(expression), :first-child, :only-child
  - **Content filters** :contains(text), has(selector), :parent, :empty
  - **Form filter** :visible, :hidden, :button, :input, :selected

- Can be chained by appending with colon (:) *

* Examples in the notes section:

jQuery: Methods

- Class/Style Methods
  - Used to apply CSS styles to the result of a selector
  - .addClass(), .css(), .height(), .position()
- DOM Methods
  - .before(), .insertBefore(), .append(), .empty(), .attr()

jQuery: Events

- jQuery simplifies events implementation
- Events are triggered by the page or end user's interaction
- Events are often used to attach a callback function
- To bind an event:
  - Use of function to bind a event directly. For example, .click()
  - Use .on() For example, .on("click", …)
  - Use .bind() For example, .bind("click", …)

Unobtrusive JavaScript

- Traditional use of JavaScript
  - <input type="button" value="Click me" onclick="handleClick()" />
- For cleaner HTML page, remove inline JavaScript references
- Use jQuery to attach handlers to DOM elements

```
<script type="text/javascript">
    $(document).ready(function () {
            $(“:button").bind("click", function () {
                    alert(“Hello World!”);
            });
    });
</script>
<div>
    <input type="button" value="Click me" />
<div>
```

Demo: jQuery selectors and events

Module 6: Client-Side Development

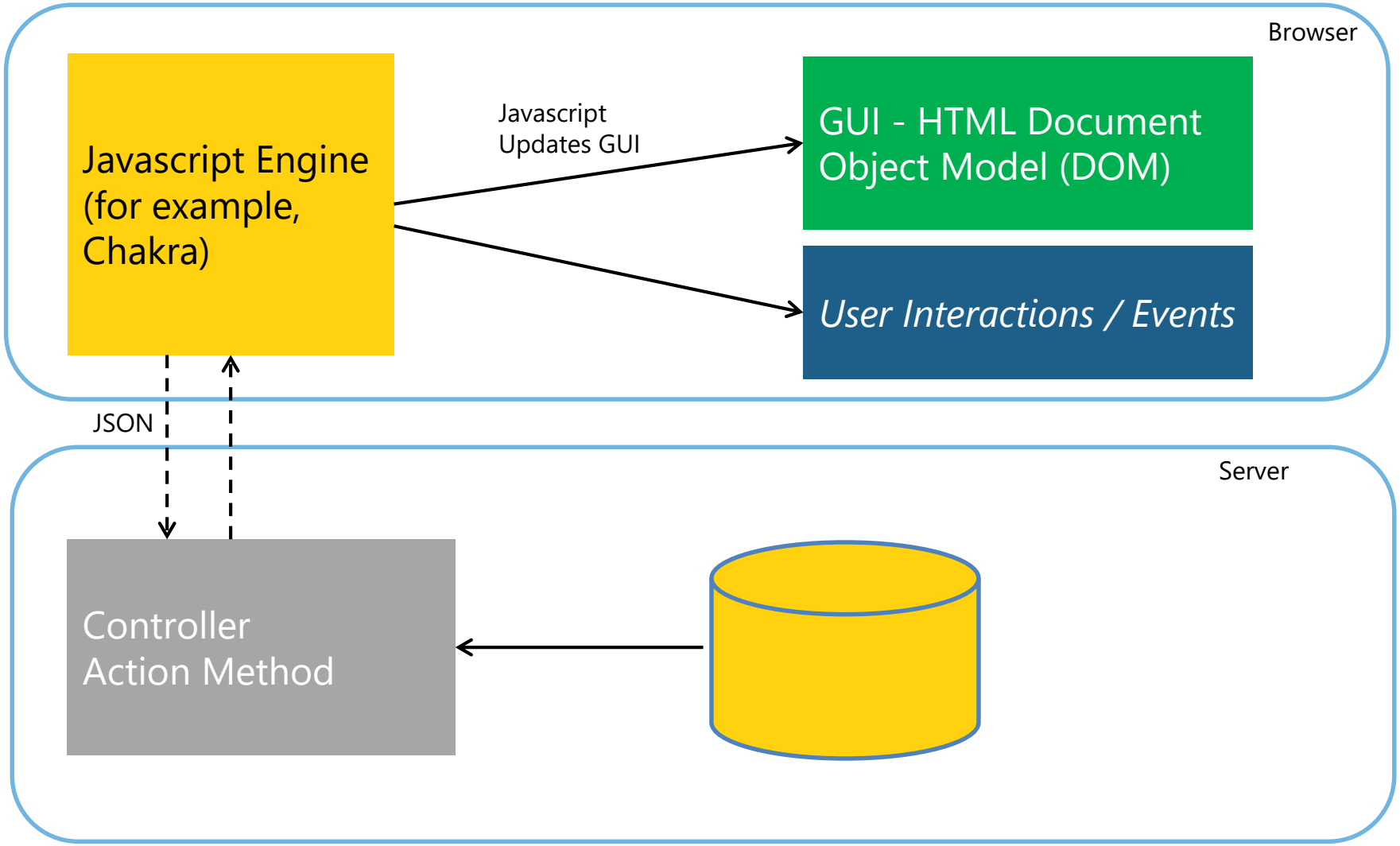
Section 3: Development Techniques

Lesson: AJAX

AJAX: (Asynchronous JavaScript and XML)

- Send and receive data from a server asynchronously (in the background) – better performance
- Uses XMLHttpRequest object
- JSON typically used instead of XML
- Back button and bookmarking challenges
- Webcrawlers do not index pages created by Ajax
- Does not work across domains by default

AJAX Engine


Browser
Javascript Engine (for example, Chakra)
Javascript Updates GUI
GUI - HTML Document Object Model (DOM)
User Interactions / Events
JSON
Server
Controller Action Method

AJAX in Jquery

- Simplifies Ajax implementation in JavaScript
- The full-feature .ajax() method performs an asynchronous HTTP (Ajax) request
- Shortcuts: .get(), .getScript(), .getJSON(), .post(), .load()

Ajax in Actions

- Create a new ActionMethodSelectorAttribute

```
[AttributeUsage(AttributeTargets.Class | AttributeTargets.Method)]
public class AcceptAjaxAttribute : ActionMethodSelectorAttribute
{
    public override bool IsValidForRequest(
    ControllerContext controllerContext, MethodInfo methodInfo)
    {
        return controllerContext.HttpContext
        .Request.IsAjaxRequest();
    }
}
```

```
    [AcceptAjax]
    public ActionResult Details(int id)
    {
        var employee = _repository.FindEmployee(id);
        return View(employee);
    }
```

Demo: AJAX

Module Summary

- In this module, you learnt about:
  - ASP.NET Core MVC support for Client-side Technologies
  - Visual Studio Client-side Dev Tooling
  - JavaScript and jQuery
  - AJAX

Lab: Client-Side Development


Microsoft