Developing ASP.NET MVC Web Applications

Session: 8

Responsive Pages

Objectives

- Define and describe how to use JavaScript
- Define and describe how to use jQuery
- Define and describe AJAX
- Explain and describe how work with AJAX

Using JavaScript 1-3

JavaScript:

- Is a client-side scripting language that allows you to develop dynamic and interactive Web applications.
- Allows your Web applications to respond to user requests without interacting with a Web server.
- Reduces the response time to deliver Web pages faster.

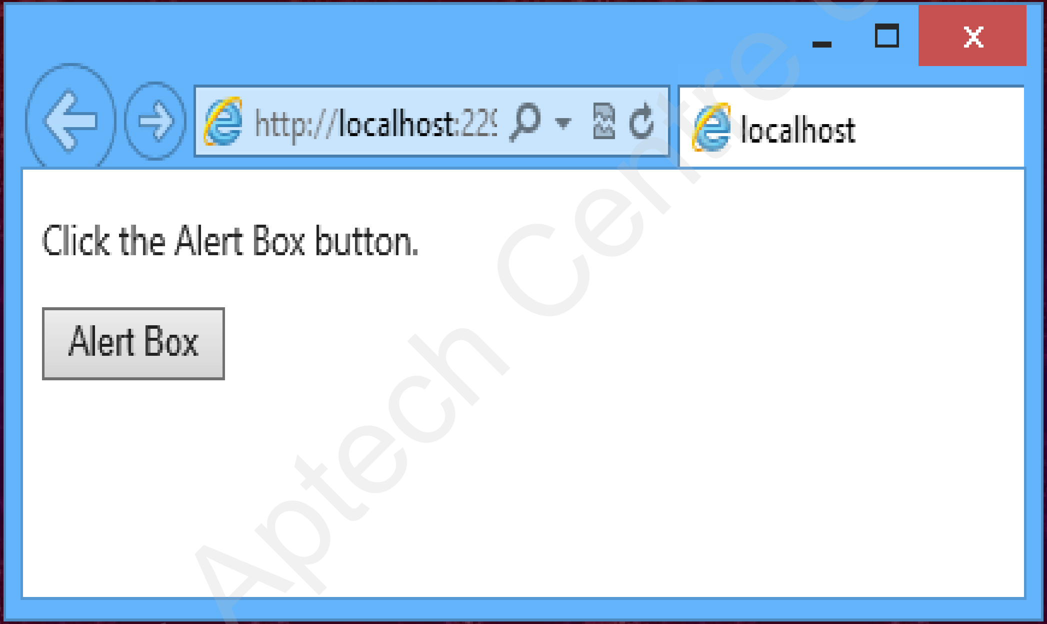
- Following code snippet shows a JavaScript in an HTML page:

Snippet

```
<!DOCTYPE html>
<html>
<body>
<p> Click the Alert Box button.</p>
<button onclick="scriptFunction()"> Alert Box </button>
<script>
functionscriptFunction()
        { alert("Welcome user")
        }
</script>
</body>
</html>
```

Using JavaScript 2-3

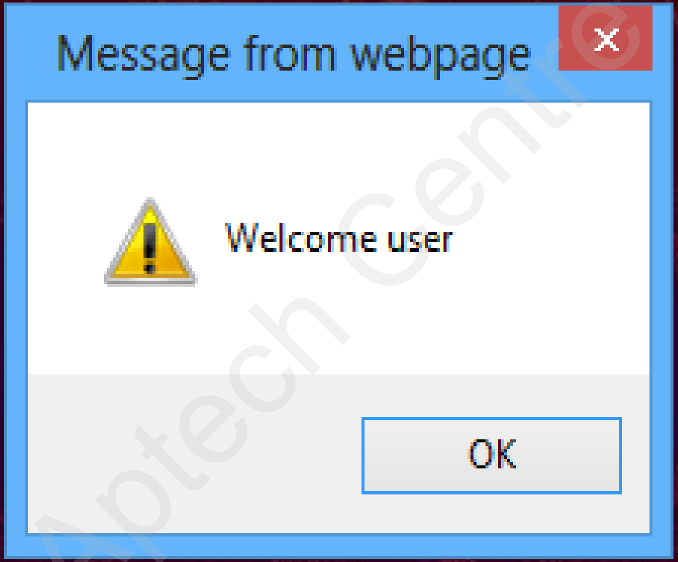
- The preceding code uses JavaScript to display a view that contains an Alert Box button.
- Following figure shows the Index.cshtml view that displays the Alert Box button:



Click the Alert Box button.

Using JavaScript 3-3

- When a user clicks the Alert Box button, the JavaScript code will execute on the browser and display an alert box.
- Following figure shows the alert box that appears when a user clicks the Alert Box button:


Message from webpage

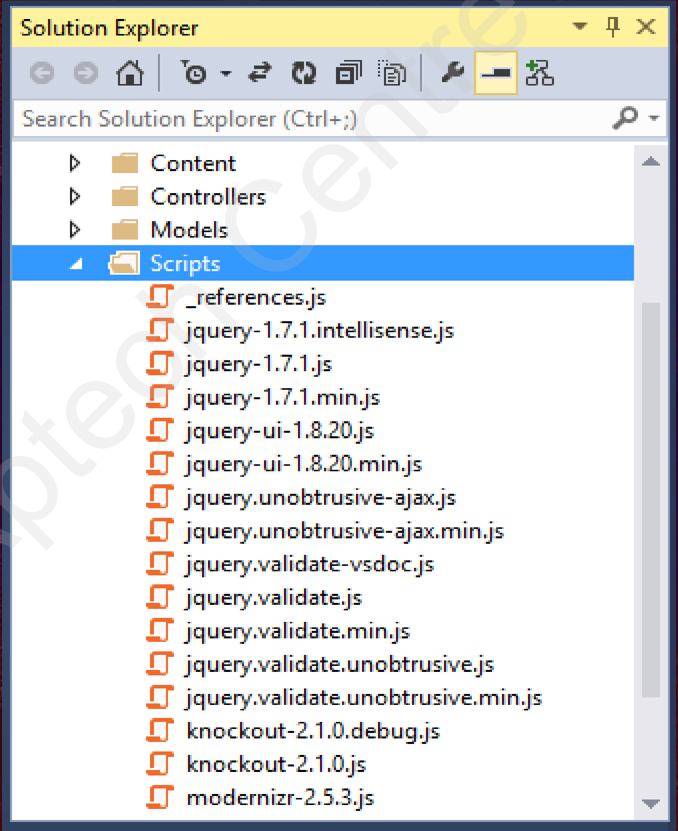
Welcome user

Unobtrusive JavaScript 1-3

- Traditionally in a Web application, JavaScript is used in the same file with HTML code.
- However, using JavaScript with HTML code is not recommended because, in a Web page, you should always separate the presentation from the business logic code.
- As a solution, you can use unobtrusive JavaScript to separate the JavaScript code from HTML markups.
- Is an approach to use JavaScript separately from HTML markups in Web applications.
- Code is stored in a file with the .js extension.

Unobtrusive JavaScript 2-3

- When you use Visual Studio 2013 to create an ASP.NET MVC application, it automatically creates a Scripts folder that contains various JavaScript files.
- Following figure shows the Solution Explorer window that displays the .js files in the Scripts folder:



Solution Explorer

Search Solution Explorer (Ctrl+;)

- Content
- Controllers
- Models
- Scripts
  - _references.js
  - jquery-1.7.1.intellisense.js
  - jquery-1.7.1.js
  - jquery-1.7.1.min.js
  - jquery-ui-1.8.20.js
  - jquery-ui-1.8.20.min.js
  - jquery.unobtrusive-ajax.js
  - jquery.unobtrusive-ajax.min.js
  - jquery.validate-vsdoc.js
  - jquery.validate.js
  - jquery.validate.min.js
  - jquery.validate.unobtrusive.js
  - jquery.validate.unobtrusive.min.js
  - knockout-2.1.0.debug.js
  - knockout-2.1.0.js
  - modernizr-2.5.3.js

Unobtrusive JavaScript 3-3

- Once you have a JavaScript file with .js file extension, you can refer it in a view using the <script> element.
- The general syntax of using the <script> element in a view is:

```
<script type= <script_type>src=<script_source>></script>
```

- where,
  - script_type: Specifies the type of scripting that has to be used, for example, "text/javascript"
  - script_source: Specifies the source of the jQuery library that has to be used, for example, "@Url.Content("~/Scripts/TestScripts.js")"
- Following code snippet shows using the <script> element:

Snippet

```
<script src= "@Url.Content("~/Scripts/Testscript.js")"
type="text/javascript">
</script>
```

- This code uses the <script> element to refer a JavaScript file named TestScripts.js.

JavaScript Libraries 1-2

- When you create an ASP.NET MVC application in Visual Studio 2013, a Script folder is automatically created with several JavaScript libraries.
- You can refer one of those libraries in a view and then, use the functionalities provided by the library.
- The default JavaScript library files that Visual Studio 2013 provides two versions:
  - One version is the main JavaScript library and another version is the minified version of the main JavaScript library.
  - The minified version contains the word min in their name.
  - The minified versions are smaller in size compared to their corresponding main version as they do not contain unnecessary characters, such as white spaces, new lines, and comments.

JavaScript Libraries 2-2

- The minified versions provide the same functionalities as their corresponding main versions.
- The advantage of using the minified versions is that it helps reduce the amount of data to be packaged in an application.
- The Visual Studio 2013 automatically generates the following characteristics:
    - Files with the vsdoc word in their name, are annotated to help Visual Studio to provide better intellisense.
    - Files with the modernizr word in their name enable you to take the advantages of the evolving Web technologies.
    - Files with the knockout word in their name are responsible for providing data binding capabilities.

Introducing jQuery 1-5

jQuery:

- Is a cross-browser JavaScript library that you can use to perform various functionalities, such as finding, traversing, and manipulating HTML elements.
- Enables you to animate HTML elements, handle events to make views of an application more dynamic and interactive.
- You use the jQuery library in a view of your application by using the <script> element in the head section of the view.
- Following code snippet shows using the <script> element:

Snippet

```
<script type= "text/javascript"
src="@Url.Content("~/Scripts/jquery-1.7.1.js")">
</script>
```

- This code uses the <script> element to use jQuery library in a view.

Introducing jQuery 2-5

- jQuery provides the document.ready() function to ensure that a view of the Web application has been fully loaded before executing the jQuery code.
- The general syntax of using the document.ready() function is :$(document).ready(function(){
// jQuery code...
});

  where,

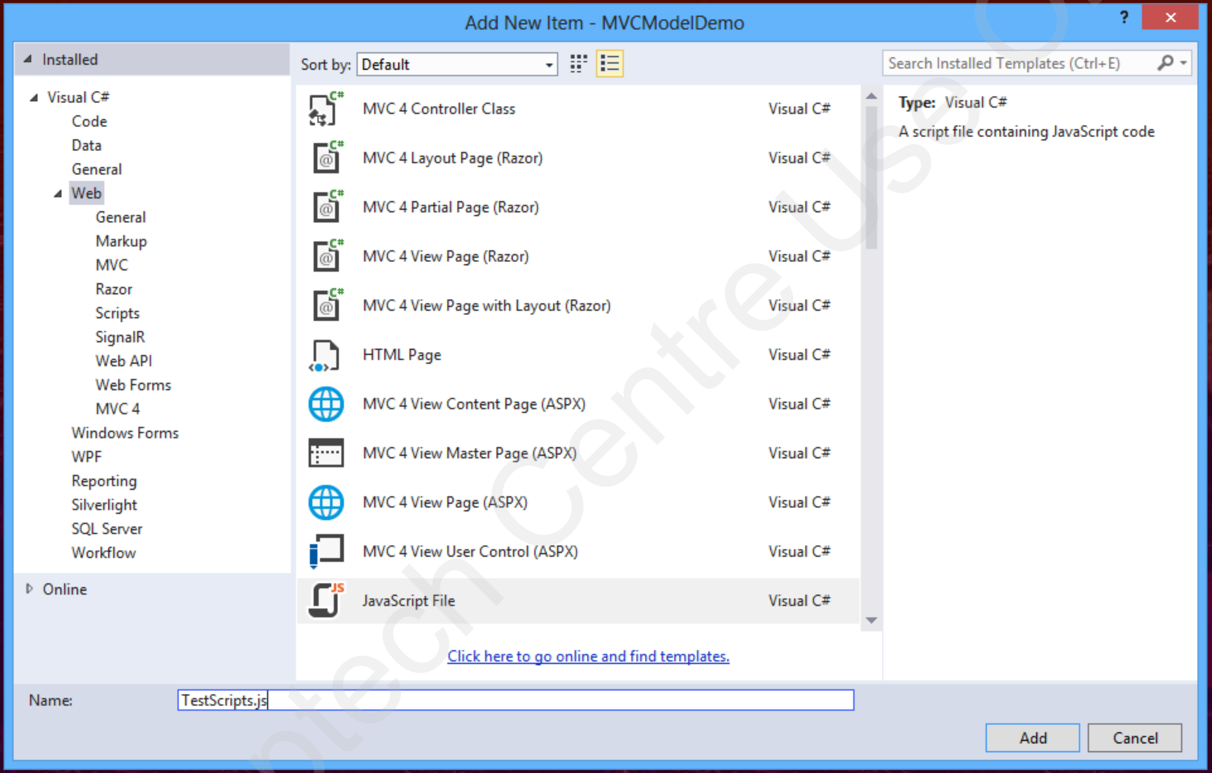
  - ($): Represents the start of a jQuery code.
  - (document).ready(): Checks the readiness of the actions performed on an HTML element.

Introducing jQuery 3-5

- Sometimes, you might require adding customized client-side functionality in your application.
- In such situation, you can add your jQuery code in a file with .js extension and then, add it to the Scripts folder of the application directory.
- To create a JavaScript file in Visual Studio 2013, you need to perform the following tasks:
  - Right-click the Scripts folder in the Solution Explorer window.
  - Select Add → New Item from the context menu that appears. The Add New Item dialog box appears.
  - Select JavaScript File template in the Add New Item dialog box.
  - Type TestScripts in the Name text field.

Introducing jQuery 4-5

- Following figure shows the Add New Item dialog box:



Add New Item - MVCModelDemo

Sort by: Default

Search Installed Templates (Ctrl+E)

Installed
- Visual C#
  - Code
  - Data
  - General
  - Web
    - General
    - Markup
    - MVC
    - Razor
    - Scripts
    - SignalR
    - Web API
    - Web Forms
    - MVC 4
  - Windows Forms
  - WPF
  - Reporting
  - Silverlight
  - SQL Server
  - Workflow
- Online

MVC 4 Controller Class	Visual C#
MVC 4 Layout Page (Razor)	Visual C#
MVC 4 Partial Page (Razor)	Visual C#
MVC 4 View Page (Razor)	Visual C#
MVC 4 View Page with Layout (Razor)	Visual C#
HTML Page	Visual C#
MVC 4 View Content Page (ASPX)	Visual C#
MVC 4 View Master Page (ASPX)	Visual C#
MVC 4 View Page (ASPX)	Visual C#
MVC 4 View User Control (ASPX)	Visual C#
JavaScript File	Visual C#

Click here to go online and find templates.

Type: Visual C#

A script file containing JavaScript code

Name: TestScripts.js

- Click Add. This will add the newly created TestScripts.js file in the Scripts folder.

Introducing jQuery 5-5

- Once you have created and added the jQuery code in a .js file, you need to invoke it by using the <script> element.
- Following code snippet shows referring to a customized TestScripts.js file:

Snippet

```
<script type= "text/javascript" src="~/Scripts/TestScripts.js"></script>
```

- This code uses the <script> element that refers to the TestScripts.js file.
- Apart from this, you can also use jQuery to select and manipulate HTML elements, handle events related to HTML elements, and add effects to HTML elements.

Selecting and Manipulating HTML Elements 1-2

- To select HTML elements and then manipulating them, jQuery allows to use some specific syntax.
- The general syntax to select HTML elements and then perform the required actions on them is:

```
$(selector).action()
```

where,

  - $: Defines or access jQuery.
  - selector: Finds an HTML element or a group of HTML elements based on their name, id, class, and attribute.
  - action(): Specifies the action to be performed on the HTML element.

Selecting and Manipulating HTML Elements 2-2

- You can use one of the following jQuery selectors, as per your requirements:
  - Element selector: Allows searching an HTML element or a group of HTML elements on the basis of their names. For example, to search all the h2 elements, you can use $("h2").
  - ID selector: Allows searching elements on the basis of their ids. For example, to search an element with the id, green, you can use $("#green").
  - Class selector: Allows searching elements on the basis of their class names. For example, to search all elements with the class, named red you can use $(".red").

Handling Events 1-6

- In jQuery, to handle events you can use functions or built-in event methods.
- You can use an event method in order to select an event and then, trigger a function when that event occurs.
- jQuery provides various event methods that you can use to handle events on HTML element.
- Following table describes some of the event methods of jQuery:

Event Method	Description
$(document).ready(function)	Allows executing a function once a document completely loads.
$(selector).click(function)	Allows executing a function on the click event of the selected elements.

Handling Events 2-6

- Following table describes some of the event methods of jQuery:

Event Method	Description
$(selector).dblclick(function)	Allows executing a function on the double-click event of the selected elements.
$(selector).mouseover(function)	Allows executing a function when the mouse pointer is moved over the selected elements.
$(selector).mouseout(function)	Allows executing a function when the mouse pointer is moved out of the selected elements.
$(selector).keydown(function)	Allows executing a function when a key is pressed on the selected elements.

Handling Events 3-6

- Consider a scenario, where you want to use a jQuery code to change the text inside the <h1> element.
- When a user clicks the Click to see the changes button the text inside the <h1> element should be changed in the view.
- Following code snippet shows the code of the Index.cshtml view:

Snippet

```
<html>
<head>
<title>Using jQuery</title>
<script type="text/javascript" src="@Url.Content("~/Scripts/jquery-
1.7.1.js")"></script>
<script src="@Url.Content("~/Scripts/Testscripts.js")"
type="text/javascript"></script>
</head>
<body>
<h1>This is a header.</h1>
<hr />
<input type="button" id="change_header" value='Click to see the changes'
/>
</body>
</html>
```

Handling Events 4-6

- The preceding code creates an Index.cshtml view containing a button named, Click to see the changes, and added a reference to the script file, named Testscripts.js.
- After creating the Index.cshtml view, you need to add the jQuery code in the TestScripts.js file to change the text inside the <h1> element.
- Following code snippet shows the content of the TestScripts.js file:

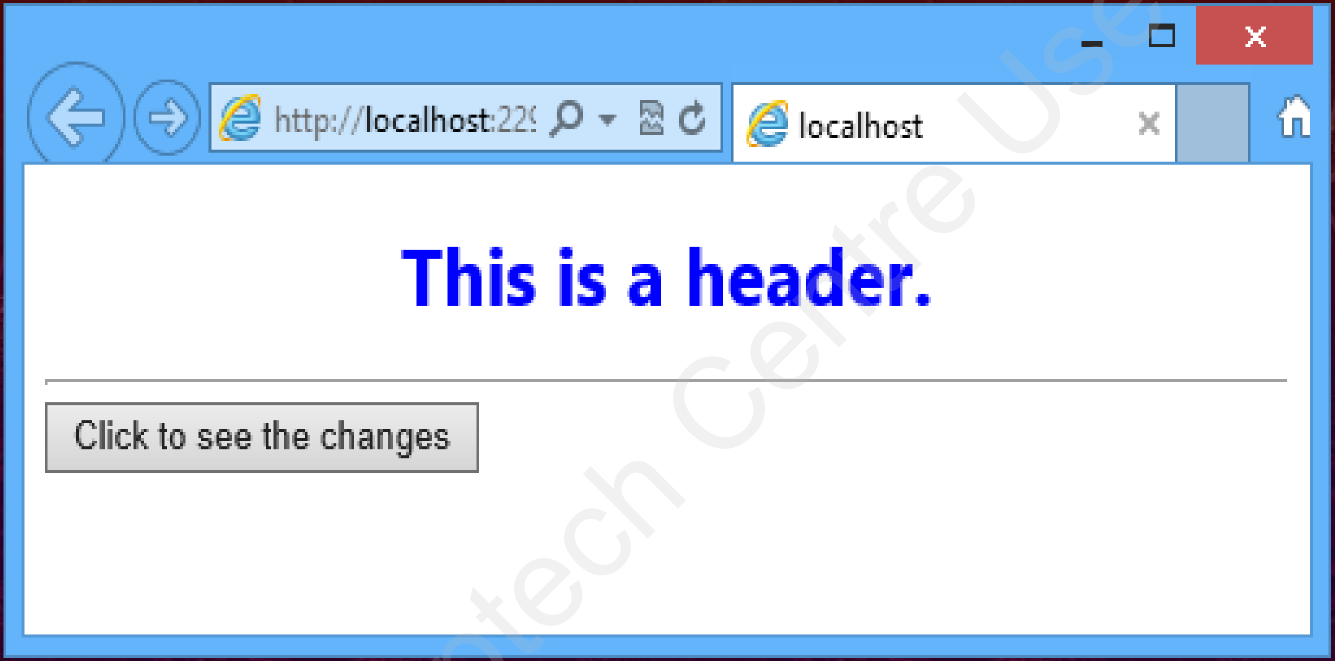
Snippet

```
$(document).ready(function () {
    $("#change_header").click(function () {
        $("h1").html("This header has changed.");
    });
});
```

- In this code, jQuery identifies a click event when the user clicks the Click to see the changes button and the function associated with the click event of the button is executed and the text inside the <h1> element changes.

Handling Events 5-6

- Following figure shows the Index.cshtml view, when you access the application:



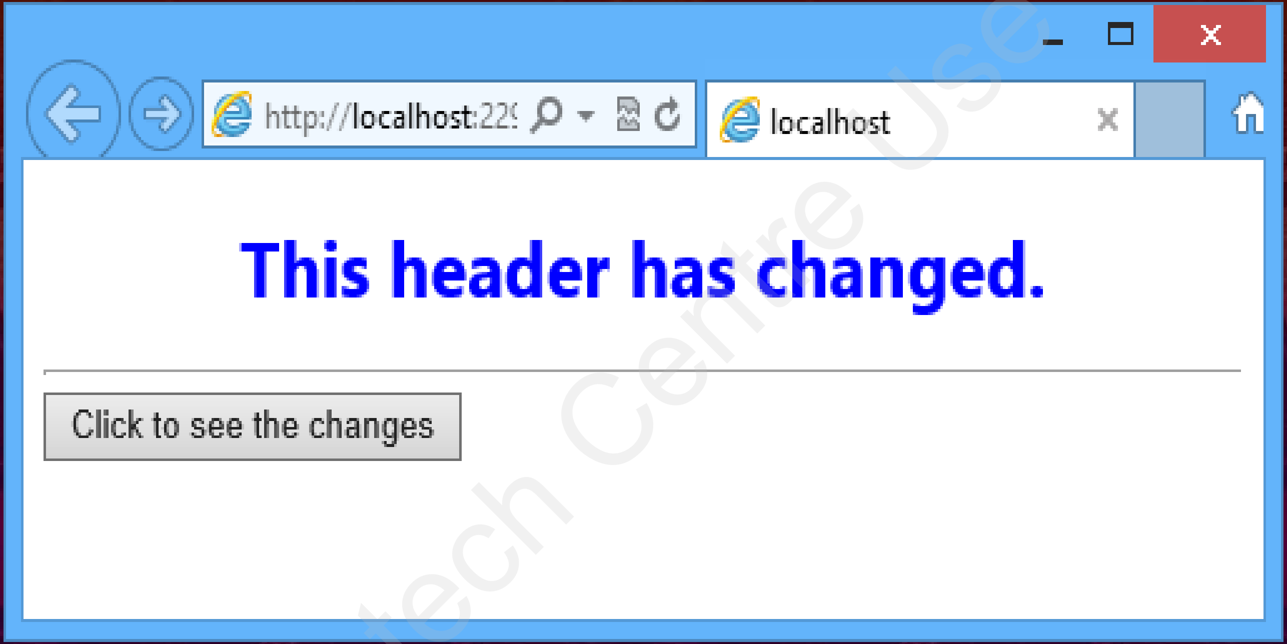
This is a header.

Click to see the changes

- When you click the Click to see the changes button the text inside the <h1> element that is “This is a header.” should be changed.

Handling Events 6-6

- Following figure shows the changes of the text inside the <h1> element:



This header has changed.

Adding Different Effects 1-7

- Using jQuery, you can perform various functionalities on HTML elements.
- For this, jQuery provides various built-in actions, such as hide and fade effect which can be applied to the HTML elements.
- These effects enable you to enrich the browsing experience of your user.
- The general syntax of using the hide() function to hide the selected elements.

```
$(selector).hide(speed)
```

where,

speed: Specifies the speed at which an element disappears. You can use one of the following values for the speed attribute:

- slow
- fast
- duration in milliseconds

Adding Different Effects 2-7

- When you specify the values for the speed attribute as duration in milliseconds, you should ensure that this value should not enclosed within quotes.
- On the other hand, when you specify values for the speed attribute as slow or fast, this value should be within the quotes.
- Consider a scenario, where you want to slowly hide a text when a user clicks a button in a view.
- In such scenario, you can use the hide() function.

Adding Different Effects 3-7

- Following code snippet shows the Index.cshtml view that contains a Click to hide the text button:

Snippet

```
<html>
<head>
<title> Hide effect </title>
<script type="text/javascript" src="@Url.Content("~/Scripts/jquery-
1.7.1.js")"></script>
<script src="@Url.Content("~/Scripts/TestScripts.js")"
type="text/javascript"></script>
</head>
<body>
<h1>Text to be disappeared</h1>
<button>Click to hide the text</button>
</body>
</html>
```

- This code contains the text, where the hide effect is to be applied, and a reference to the file that contains the jQuery code with hide effects is specified.
- Now, to hide the text, when a user clicks the button you need to use the hide()function in the .js file.

Adding Different Effects 4-7

- Following code snippet shows the jQuery code in the TestScripts.js file to hide a text:

Snippet

```
$(document).ready(function(){
        $(" button").click(function(){
            $(" h1").hide("slow");
        });
    });
```

- In this code, the text “Text to be disappeared” is displayed above the Click to hide the text button. When the user clicks this button, the text “Text to be disappeared” slowly disappears from the browser window.

Adding Different Effects 5-7

- You can use the fade effect on a selected element using jQuery.
- This effect enables you to gradually reduce the opacity of the selected elements.
- Some of the common fade function that you can use on selected elements are fadeOut()and fadeIn().
- Consider a scenario, where you want to fade out the text “Text to be faded out” when a user clicks over it.
- Thereafter, you want to fade in and display the disappeared text when a user clicks the Restore Text button. In such scenario, you can use the fade effect of jQuery.

Adding Different Effects 6-7

- Following code snippet shows the Index.cshtml file that contains the text where the fade effect is to be applied and a Restore Text button:

Snippet

```
<html>
<head>
<title>Hide Effect</title>
<script type="text/javascript" src="@Url.Content("~/Scripts/jquery-
1.7.1.js")"></script>
<script src="@Url.Content("~/Scripts/TestScripts.js")"
type="text/javascript"></script>
</head>
<body>
<p>Text to be faded out!</p>
<br />
<button>Restore Text</button>
</body>
</html>
```

- This code contains the text, where the fade effect is to be applied, and a reference to the script file that contains the jQuery code that defines fade effects.

Adding Different Effects 7-7

- Once you have created the view, you can define the jQuery code to implement fade effect in the TestScripts.js file.
- Following code snippet shows the jQuery code to perform fade in and fade out effects:

Snippet

```
$(document).ready(function () {
    $("p").click(function () {
        $(this).fadeOut(1400);
    });
    $("button").click(function () {
        $("p").fadeIn(1000);
    });
});
```

- In this code:
  - The <p> element is referred using its name.
  - The this keyword in the click event handler of the <p> element refers to the current HTML element, which is the <p> element.
  - When the user clicks the text “Text to be faded out” the text fades away. When the user clicks the Restore Text button, the text “Text to be faded out” reappears.

JQuery UI

- jQuery UI:
  - Is a set of features that allows you to develop highly interactive Web applications.
  - Contains interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript library.
  - Library provides a set of widgets that you can use to design UIs and apply various effects to the UI elements.
  - Library can be used in a view by including a reference to the jQuery UI script file in the view.
- Following code snippet shows adding reference to the jQuery UI script file in a view:

Snippet

```
<script type= "text/javascript" src="@Url.Content("~/Scripts/jquery-ui-
1.8.20.js")"></script>
```

Applying jQuery UI Effects 1-4

- While applying style animations, jQuery UI provides a technique that you use to add, remove, or toggle class for the HTML elements.
- Following code snippet shows the Index.cshtml view that contains three buttons:

Snippet

```
<html>
<head>
<title>Using jQuery UI</title>
<link rel="stylesheet" href="@Url.Content("~/Content/main.css")">
<script type="text/javascript" src="@Url.Content("~/Scripts/jquery-
1.7.1.js")"></script>
<script type="text/javascript" src="@Url.Content("~/Scripts/jquery-ui-
1.8.20.js")"></script>
<script src="@Url.Content("~/Scripts/TestScripts.js")"
type="text/javascript"></script>
</head>
<body>
<div id="div1"></div>
<button id="button1">Add new color</button>
<button id="button2">Remove a color</button>
<button id="button3">Toggle color</button>
</body>
</html>
```

Applying jQuery UI Effects 2-4

- The preceding code creates a view that contains a <div> element and three buttons, named, Add new color, Remove a color, and Toggle color.
- The view uses the default a stylesheet of the application named Site.css.
- Following code snippet shows the defined styles in the Site.css file:

Snippet

```
div {
width:350px;
height: 250px;
background-color:blue;
}
 .myclass {
width : 350px;
height: 250px;
background-color:purple;
}
```

- This code defines style to the <div> and myclass elements.

Applying jQuery UI Effects 3-4

- Once you define the style in the Site.css file, you can define jQuery code to perform these effects in the script file named, TestScripts.js.
- Following code snippet shows the TestScripts.js that contains jQuery code to perform these effects:

Snippet

```
$(document).ready(function () {
    $("#button1").click(function () {

        $("#div1").addClass("myclass", 550);

    });
    $("#button2").click(function () {

        $("#div1").removeClass("myclass", 550);
    });
    $("#button3").click(function () {
        $("#div1").toggleClass("myclass", 550);
    });
});
```

Applying jQuery UI Effects 4-4

- In the preceding code:
  - Applies different effects, such as when a user clicks the Add new color button, the myclass class defined in the Site.css file will be applied to the <div> element.
  - Secondly, when a user clicks the Remove a color button, the myclass class that is applied to the <div> element will be removed.
  - Finally, when the user clicks the Toggle colors button, the myclass class will be applied to the <div> element if it is not already applied otherwise the class will be removed from the <div> element.

Introducing AJAX

- AJAX:
  - Is a Web development technique that allows you to make requests to the server in the background using client-side code.
  - Enables you to update a view without reloading it completely.
  - Improves the performance of your application and the user experience.
  - To understand the concept of AJAX, you should understand about asynchronous communication.
- Asynchronous communication:
  - Is the ability of a Web application to send multiple requests and receive responses from the server simultaneously.
  - Enables you to work on the application without being affected by the responses received from the server.

AJAX Helpers 1-2

- ASP.NET MVC application also uses a set of AJAX helpers.
- AJAX helpers enable you to create forms and links that point to controller actions.
- The only difference is that AJAX helpers behave asynchronously.
- While using AJAX helpers, you should not explicitly write JavaScript code make the asynchrony work.
- To make use of it, you should ensure that the jquery.unobtrusive-ajax script file is present in the Scripts folder of your application directory.
- However, if you want to include the file manually, you need to use the <script> element.

AJAX Helpers 2-2

- Following code snippet shows adding reference to the jquery.unobtrusive-ajax script file manually in the default _Layout.cshtml file:

Snippet

```
<script src="~Scripts/jquery-1.7.1.min.js")>
</script>
<script src="~/Scripts/Scripts/jquery.unobtrusive-ajax.min.js")>
</script>
@RenderSection("scripts", required:false);
```

- This code adds reference to the jquery.unobtrusive-ajax script file in the default _Layout.cshtml file.

Unobtrusive AJAX 1-3

- The ASP.NET MVC Framework provides built-in support for unobtrusive AJAX.
- It enables you to use AJAX helpers to define AJAX features instead of writing code in your views.
- To enable unobtrusive AJAX feature in an application, you need to configure the Web.config file.
- To enable unobtrusive AJAX feature in the Web.config file, you need to set the UnobtrusiveJavaScriptEnabled property to true under the <appSettings> element of the Web.config.

Unobtrusive AJAX 2-3

- Following code snippet shows enabling the unobtrusive AJAX feature in the Web.config file:

Snippet

```
<appSettings>
<add key="webpages:Enabled" value="false" />
<add key="UnobtrusiveJavaScriptEnabled" value="true" />
</appSettings>
```

- This code sets the value of the UnobtrusiveJavaScriptEnabled property to true to enable the unobtrusive AJAX feature.
- In addition to enabling the unobtrusive AJAX feature in the Web.config file, you need to add references to the jQuery JavaScript libraries that implement the unobtrusive AJAX functionality.

Unobtrusive AJAX 3-3

- For that you need to add reference of the jQuery JavaScript libraries in the views that need to use it.
- Following code snippet shows adding references to jQuery JavaScript libraries in a view:

Snippet

```
<script type= "text/javascript" src="@Url.Content("~/Scripts/ jquery-
1.7.1.min.js")">

</script>

<script src="@Url.Content("~/Scripts/ jquery.unobtrusive-
ajax.min.js")"></script>
```

- In this code, the jquery-1.7.1.min.js file contains the core jQuery library, and the jquery.unobtrusive-ajax.min.js file contains the AJAX library.

Working with AJAX

- Traditionally, to update the content of a view the full page refresh technique was used.
- In this technique, to update the content of a view an element in the page triggers a request to the server.
- When the server finishes processing the request, a new page is sent back to the browser.
- However, AJAX provides an approach to improve the process of updating views.
- In an ASP.NET MVC application, you can implement AJAX using AJAX.ActionLInk() and AJAX Forms.

AJAX ActionLinks 1-2

- Similar to the HTML helper methods, most of the methods of AJAX property are extension methods.
- The Ajax.ActionLink() helper method enables you to create an anchor tag with asynchronous behavior.
- Following code snippet shows using the Ajax.ActionLink() method in the Index.cshtml view:

Snippet

```
<div id="attachmentImage">
@Ajax.ActionLink("Attachment", "AttachmentImage",
new AjaxOptions
{
HttpMethod = "Post",
InsertionMode = InsertionMode.Replace,
LoadingElementId = "loadingImage",
UpdateTargetId = "attachmentImage"
})
</div>
<div id="loadingImage" style="display:none">
Loading attachment...
</div>
```

AJAX ActionLinks 2-2

- In the preceding code:
  - The first parameter of the Ajax.ActionLink() method specifies the link text.
  - The second parameter specifies the name of the action method that needs to be invoked asynchronously.

AJAX Forms 1-4

- Consider a scenario, where you want to enable a user to search for product.
- When a user types a text to search for a product, the details of the matching products should be retrieved from the server and displayed on the same page, without having to post the page back to the server.
- In such scenario, you should use an asynchronous form element on the view.

AJAX Forms 2-4

- Following code snippet shows using an asynchronous form element:

Snippet

```
@using (Ajax.BeginForm("ProductSearch", "Product",
new AjaxOptions {
InsertionMode=InsertionMode.Replace,
HttpMethod="GET",
OnFailure="searchFailed",
LoadingElementId="ajax-loader",
UpdateTargetId="searchresults",
}))
{
<input type="text" name="q" />
<input type="Submit" value="search" />
<img id="ajax-loader"
src="@~/Content/Images/ajax-loader.gif"
style="display:none"/>
}
```

AJAX Forms 3-4

- In the preceding code:
  - Creates an AJAX form that contains a text box and a Submit button and contains a <div> element with the id, search results on the view.
  - The first parameter of the Ajax.BeginForm() method specifies the ProductSearch() action method that handles the AJAX request on the server.
  - The second parameter specifies the Product controller class that contains the action method.
  - The third parameter specifies various options for the AJAX request.

AJAX Forms 4-4

- When the user clicks the Submit button, the browser sends an asynchronous GET request to the ProductSearch() action method of the Product controller.
- Following code snipept shows defining the ProductSearch() action method:

Snippet

```
public ActionResultProductSearch(string q)
 {
ProductDBContextdb = new ProductDBContext();
var products = db.Products.Where (a=>a.name.Contains(q));
return PartialView("_searchresults", products);
     }
```

- In this code:
  - Retrieves product records that match the user specified search string from the database.
  - Then it returns a partial view named _searchresults that contains the retrieved records. This partial view is then rendered in the searchresults element on the view.

Summary

- JavaScript is a client-side scripting language that allows you to develop dynamic and interactive Web applications.
- Unobtrusive JavaScript is an approach to use JavaScript separately from HTML markups in Web applications.
- jQuery is a cross-browser JavaScript library that you can use to perform various functionalities, such as finding, traversing, and manipulating HTML elements.
- jQuery UI library provides a set of widgets that you can use to design UIs and apply various effects to the UI elements.
- AJAX is a Web development technique that allows you to make requests to the server in the background using client-side code.
- ASP.NET MVC application uses a set of AJAX helpers that enable you to create forms and links that point to controller actions.
- The ASP.NET MVC Framework provides built-in support for unobtrusive AJAX that enables you to use AJAX helpers to define AJAX features instead of writing code in your views.