

# **Developing ASP.NET MVC Applications**

# **Developing ASP.NET MVC Applications**

## **Trainer Guide**

© 2015 Aptech Limited

All rights reserved.

No part of this book may be reproduced or copied in any form or by any means – graphic, electronic or mechanical, including photocopying, recording, taping, or storing in information retrieval system or sent or transferred without the prior written permission of copyright owner Aptech Limited.

All trademarks acknowledged.

**APTECH LIMITED**

Contact E-mail: [ov-support@onlinevarsity.com](mailto:ov-support@onlinevarsity.com)

Edition 1 - 2015



## Dear Learner,

We congratulate you on your decision to pursue an Aptech course.

Aptech Ltd. designs its courses using a sound instructional design model – from conceptualization to execution, incorporating the following key aspects:

- Scanning the user system and needs assessment

Needs assessment is carried out to find the educational and training needs of the learner

Technology trends are regularly scanned and tracked by core teams at Aptech Ltd. TAG\* analyzes these on a monthly basis to understand the emerging technology training needs for the Industry.

An annual Industry Recruitment Profile Survey is conducted during August - October to understand the technologies that Industries would be adapting in the next 2 to 3 years. An analysis of these trends & recruitment needs is then carried out to understand the skill requirements for different roles & career opportunities.

The skill requirements are then mapped with the learner profile (user system) to derive the Learning objectives for the different roles.

- Needs analysis and design of curriculum

The Learning objectives are then analyzed and translated into learning tasks. Each learning task or activity is analyzed in terms of knowledge, skills and attitudes that are required to perform that task. Teachers and domain experts do this jointly. These are then grouped in clusters to form the subjects to be covered by the curriculum.

In addition, the society, the teachers, and the industry expect certain knowledge and skills that are related to abilities such as *learning-to-learn, thinking, adaptability, problem solving, positive attitude etc.* These competencies would cover both cognitive and affective domains.

**A precedence diagram for the subjects is drawn where the prerequisites for each subject are graphically illustrated. The number of levels in this diagram is determined by the duration of the course in terms of number of semesters etc. Using the precedence diagram and the time duration for each subject, the curriculum is organized.**

- Design & development of instructional materials

The content outlines are developed by including additional topics that are required for the completion of the domain and for the logical development of the competencies identified. Evaluation strategy and scheme is developed for the subject. The topics are arranged/organized in a meaningful sequence.

The detailed instructional material – Training aids, Learner material, reference material, project guidelines, etc.- are then developed. Rigorous quality checks are conducted at every stage.

➤ Strategies for delivery of instruction

Careful consideration is given for the integral development of abilities like thinking, problem solving, learning-to-learn etc. by selecting appropriate instructional strategies (training methodology), instructional activities and instructional materials.

The area of IT is fast changing and nebulous. Hence considerable flexibility is provided in the instructional process by specially including creative activities with group interaction between the students and the trainer. The positive aspects of Web based learning –acquiring information, organizing information and acting on the basis of insufficient information are some of the aspects, which are incorporated, in the instructional process.

➤ Assessment of learning

The learning is assessed through different modes – tests, assignments & projects. The assessment system is designed to evaluate the level of knowledge & skills as defined by the learning objectives.

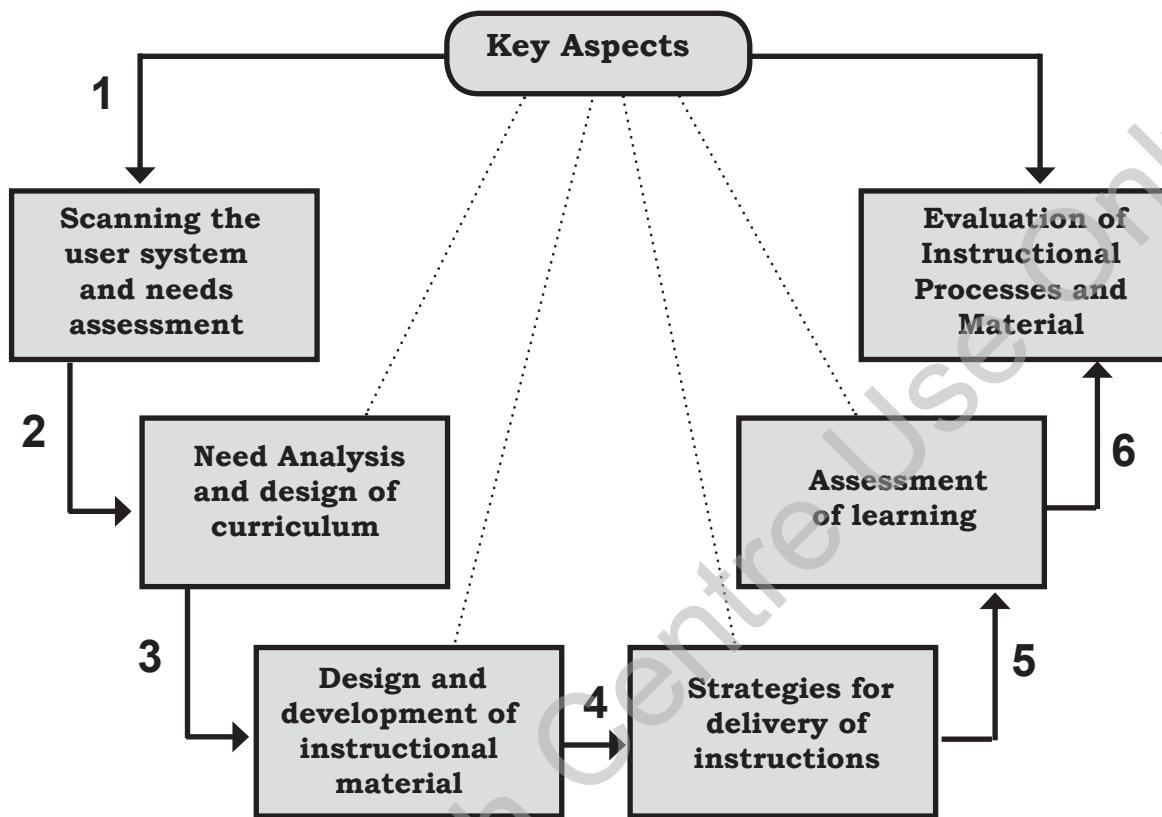
➤ Evaluation of instructional process and instructional materials

The instructional process is backed by an elaborate monitoring system to evaluate - on-time delivery, understanding of a subject module, ability of the instructor to impart learning. As an integral part of this process, we request you to kindly send us your feedback in the reply pre-paid form appended at the end of each module.

\*TAG – Technology & Academics Group comprises of members from Aptech Ltd., professors from reputed Academic Institutions, Senior Managers from Industry, Technical gurus from Software Majors & representatives from regulatory organizations/forums.

Technology heads of Aptech Ltd. meet on a monthly basis to share and evaluate the technology trends. The group interfaces with the representatives of the TAG thrice a year to review and validate the technology and academic directions and endeavors of Aptech Ltd.

### Aptech New Products Design Model



“ Any expansion is life  
all contraction is death ”

For Aptech Centre Use Only

---

## Preface

---

The Trainer's Guide for **Developing ASP.NET MVC Applications**, begins with Introduction to ASP.NET MVC which covers the basic knowledge about how to create and consume the ASP.NET MVC Applications. After that it covers the topics on Controllers, Views and Models. Then it covers the advanced topics such as Data Validations and Annotations, Data Access, Styles and Layouts, Responsive Pages, State Management, Optimization, Authentication and Authorization, Security, Globalization, Debugging and Monitoring, Advanced concepts in ASP.NET MVC and finally it covers the topic on Testing and Deploying ASP.NET MVC Applications.

The faculty/trainer should teach the concepts in the theory class using the slides. This Trainer's Guide will provide guidance on the flow of the session and also provide tips, notes, in-class questions, references and additional examples wherever necessary. The trainer can ask questions to make the session interactive and also to test the understanding of the students.

This book is the result of a concentrated effort of the Design Team, which is continuously striving to bring you the best and the latest in Information Technology. The process of design has been a part of the ISO 9001 certification for Aptech-IT Division, Education Support Services. As part of Aptech's quality drive, this team does intensive research and curriculum enrichment to keep it in line with industry trends.

We will be glad to receive your suggestions.

Design Team

**“Learning how to learn is  
life's most important skill”**

For Aptech Centre Use Only

## Table of Contents

### Sessions

1. Introduction to ASP.NET MVC
2. Controllers in ASP.NET MVC
3. Views in ASP.NET MVC
4. Models in ASP.NET MVC
5. Data Validation and Annotation
6. Data Access
7. Consistent Styles and Layouts
8. Responsive Pages
9. State Management and Optimization
10. Authentication and Authorization
11. Security
12. Globalization
13. Debugging and Monitoring
14. Advanced Concepts of ASP.NET MVC
15. Testing and Deploying

**Knowing is not enough  
we must apply;  
Willing is not enough,  
we must do**

For Aptech Centre Use Only

## Session 1

### Introduction to ASP.NET MVC

---

#### **1.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of this session in-depth. Here, you can ask students the key topics they are already familiar with or you can discuss their previous experience with similar topics. Mostly, the student allowed in this course is familiar with creating basic application programs in some other programming languages. Since, this is the first session of ASP.NET MVC, it is important to recall all the concepts that are necessary in order to grab and perceive the knowledge in the current session. Prepare a question or two, which will be key points to relate the current session objectives.

#### **1.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe the layers of Web application
- Explain the structure of an ASP.NET MVC application
- Explain the evaluation of Web application
- Explain and describe how to create Web application in Visual Studio 2013

#### **1.1.2 Teaching Skills**

To teach this session, you should be well versed with Windows App Architecture. You should have expertise with the basic concepts and important features of Extensible Application Markup Language (XAML) language. You should have sound knowledge on Windows Store components, packaging and deployment and uploading an app to the Windows Store to explain the students in an efficient way.

For teaching in the class, you are expected to use slides and LCD projectors.

**Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

**In-Class Activities**

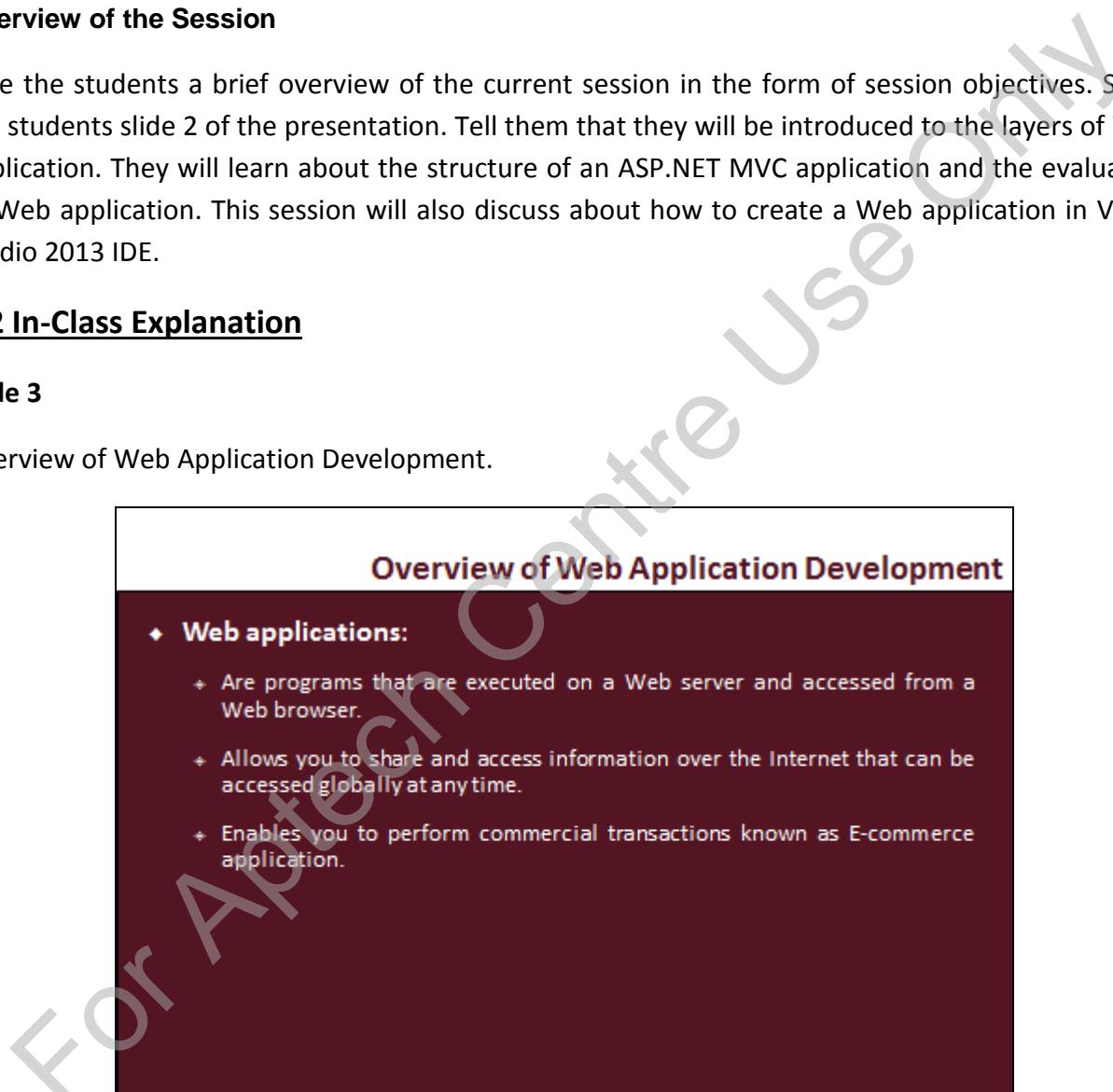
Follow the order given here during In-Class activities.

**Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the layers of Web application. They will learn about the structure of an ASP.NET MVC application and the evaluation of Web application. This session will also discuss about how to create a Web application in Visual Studio 2013 IDE.

**1.2 In-Class Explanation****Slide 3**

Overview of Web Application Development.



**Overview of Web Application Development**

- ◆ **Web applications:**
  - ◆ Are programs that are executed on a Web server and accessed from a Web browser.
  - ◆ Allows you to share and access information over the Internet that can be accessed globally at any time.
  - ◆ Enables you to perform commercial transactions known as E-commerce application.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 3

Use slide 3 to explain that Web applications are programs that are executed on a Web server and accessed from a Web browser. It allows you to share and access information over the Internet that can be accessed globally at any time.

Tell them that Web application enables them to perform commercial transactions known as E-commerce application. E-Commerce is very popular and best use of Internet technology. Using E-commerce the user can sell goods and services to any person in any corner of the world.

#### Slide 4

Let us understand Web application layers.

The slide has a dark red background with white text. At the top, it says 'Web Application Layers 1-2'. Below that is a bulleted list:

- ◆ Web applications are typically divided into three layers:
  - ◆ **Presentation layer:** Enables users to interact with the application.
  - ◆ **Business logic layer:** Enables to control the flow of execution and communication between the presentation layer and data layer.
  - ◆ **Data layer:** Enables to provide the application data stored in databases to the business logic layer.

At the bottom left is the copyright notice '© Aptech Ltd.' and at the bottom right is 'Introduction to ASP.NET MVC/Session 1' and the number '4'.

Using slide 4, explain students about the three layers of Web application.

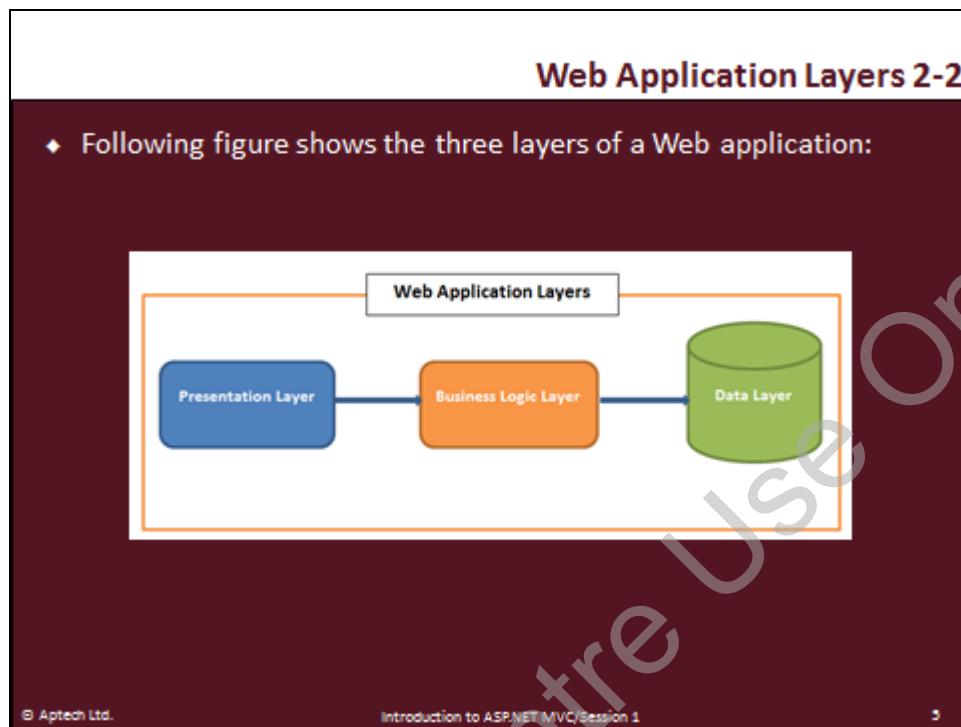
Tell them that the presentation layer enable users to interact with the application. Then, tell them that the business logic layer enables users to control the flow of execution and communication between the presentation layer and data layer. Also, mention that data layer provides the application data stored in databases to the business logic layer.

#### Discussion:

Initiate a discussion on validation of user inputs. Ask the students in which layer the validation logic will run. Carry forward the discussion to conclude that client-side validation will be in the presentation layer while server-side validation will be in the business logic layer.

## Slide 5

Let us discuss the Web Application Layers.



With the help of image given on slide 5 explain to the students how three layers are arranged. Tell them about various functions of each layer. Show them how the three layers are logically separated from each others and have independent functionality.

## Slide 6

Let us understand Web application architectures.

### Web Application Architectures

- ◆ Architecture of an application depends on the system in which the layers of the application are distributed and communicated to each other.
- ◆ An application can be based on one of the following types of architectures:
  - + **Single-tier:** In this architecture, all the three layers are integrated together and installed on a single computer.
  - + **Two-tier:** In this architecture, the three layers are distributed over two tiers, a client and a server.
  - + **Three-tier:** In this architecture, the three layers of the application are distributed across different computers.
  - + **N-tier:** In this architecture, the components of the three-tier are further separated.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1

In slide 6, tell the students that the architecture of an application depends on the system in which the layers of the application are distributed and communicated to each other.

Explain that an application can be based on one of the following types of architectures. Tell them that in a Single-tier architecture, all the three layers are integrated together and installed on a single computer.

Next, tell them that in the Two-tier architecture, the three layers are distributed over two tiers, a client and a server. Also, tell them that in the Three-tier architecture, the three layers of the application are distributed across different computers.

Finally, tell them that in the N-tier architecture, the components of the three-tier are further separated.

#### Information:

Inform students that at the minimum and N-tier architecture are comprised of three tiers, which mean that a Three-tier is a form of N-tier architecture.

## Slide 7

Let us understand the types of Web pages.

**Types of Web Pages**

- ◆ A Web application consists of Web pages.
- ◆ A Web page can be categorized on the following two types:
  - ◆ **Static Web page:** Consists of only Hyper Text Markup Language (HTML) to present content to users.
  - ◆ **Dynamic Web page:** Consists of HTML in combination with server-side and client-side scripts to respond to user actions.

© Aptech Ltd. Introduction to ASP.NET MVC Session 1 7

Use slide 7 to explain students about the two types of Web pages. Tell them that a Web application consists of Web pages.

Then, tell them that a Web page can be categorized on the following two types:

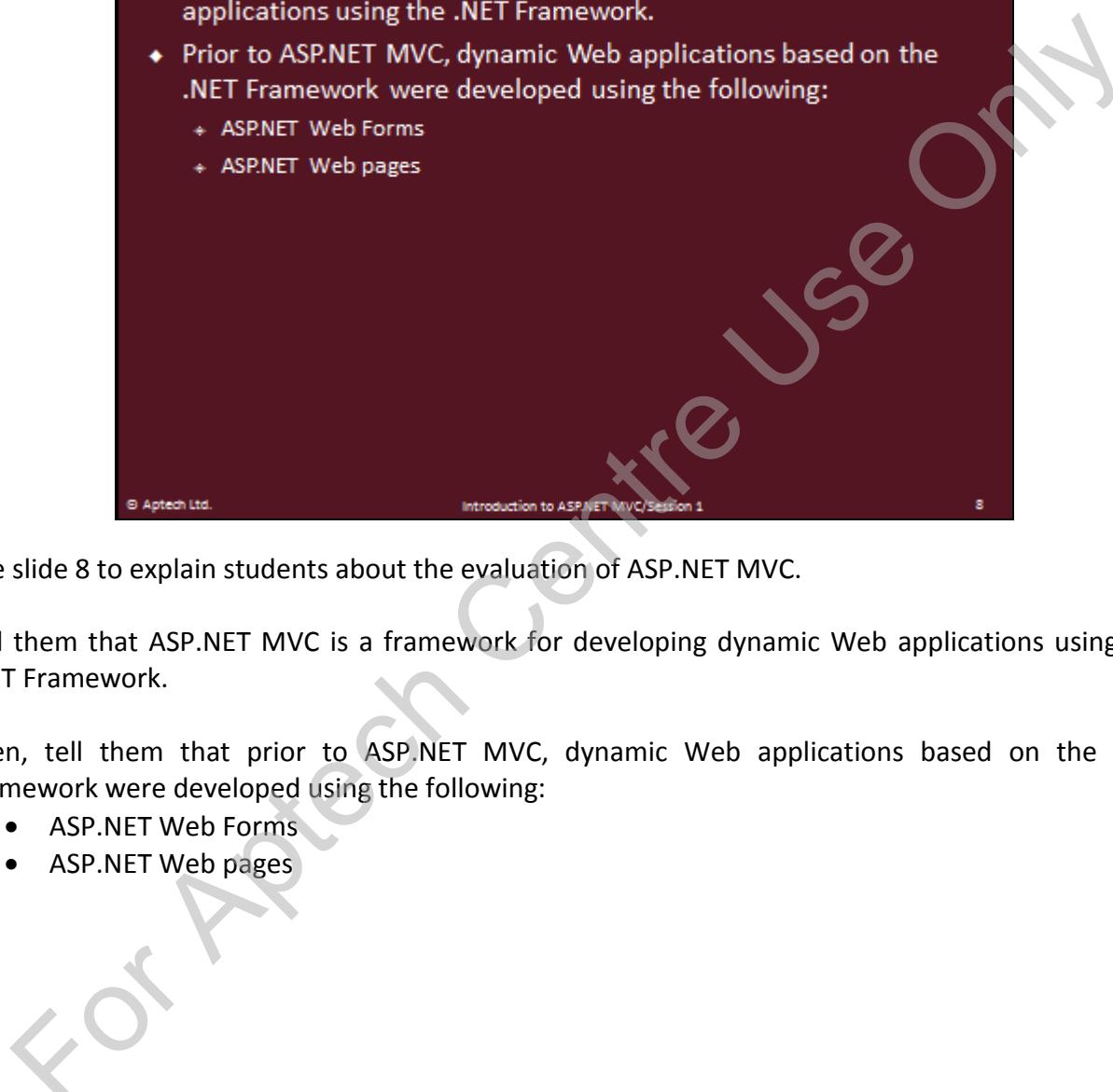
- Static Web page that consists of only Hypertext Markup Language (HTML) to present content to users.
- Dynamic Web page that consists of HTML in combination with server-side and client-side scripts to respond to user actions.

### Discussion:

Ask students to provide examples of static and dynamic Web pages that they encounter while using the Internet. Tell them that Web page displaying read only news is a static Web page as users can only read news on the Web page without further interaction. On the other hand, the home page of Google is a dynamic Web page as it allows users to search content.

## Slide 8

Let us understand the evaluation of ASP.NET MVC.



### Evolution of ASP.NET MVC

- ◆ ASP.NET MVC is a framework for developing dynamic Web applications using the .NET Framework.
- ◆ Prior to ASP.NET MVC, dynamic Web applications based on the .NET Framework were developed using the following:
  - + ASP.NET Web Forms
  - + ASP.NET Web pages

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 8

Use slide 8 to explain students about the evaluation of ASP.NET MVC.

Tell them that ASP.NET MVC is a framework for developing dynamic Web applications using the .NET Framework.

Then, tell them that prior to ASP.NET MVC, dynamic Web applications based on the .NET Framework were developed using the following:

- ASP.NET Web Forms
- ASP.NET Web pages

## Slide 9

Let us understand about ASP.NET applications.

The slide has a dark red header bar with the title 'ASP.NET Applications 1-3' in white. The main content area is dark maroon with white text. A large watermark 'For Aptech Centre Use Only' is diagonally across the slide. At the bottom, there's a thin white footer bar with small text: '© Aptech Ltd.', 'Introduction to ASP.NET MVC/Session 1', and a small number '9'.

◆ **ASP.NET:**

- ◆ Is a server-side technology that enables you to create dynamic Web applications using advanced features.
- ◆ Are comprised of .aspx Web pages that combine both client-side and server-side scripts to build dynamic Websites.
- ◆ Application can be deployed on a Web server such as Internet Information Services (IIS), which is the Web server for the Windows platform.

In slide 9, tell the students that the ASP.NET is a server-side technology that enables them to create dynamic Web applications using advanced features. ASP.NET applications are comprised of .aspx Web pages that combine both client-side and server-side scripts to build dynamic Websites.

Then, explain that an application can be deployed on a Web server such as Internet Information Services (IIS), which is the Web server for the Windows platform.

## Slide 10

Steps involved in Request-Response process.

The slide has a dark red header bar with the title 'ASP.NET Applications 2-3' in white. The main content area is white with a dark red sidebar on the left. The sidebar contains a bulleted list of four steps:

- ◆ The request-response process for an ASP.NET Web page contains the following steps:
  - + Browser sends a request for an ASP.NET Web page.
  - + When the request arrives, the IIS server intercepts the request, loads the requested file, and forwards it to the ASP.NET runtime for processing.
  - + The ASP.NET runtime that contains the ASP.NET script engine processes the requested ASP.NET page and generates the response.
  - + The IIS server sends back the response to the Web server that requested the page.

At the bottom of the slide, there is a footer bar with the following text: '© Aptech Ltd.', 'Introduction to ASP.NET MVC/Session 1', and '10'.

Use slide 10 to explain students that the request-response process for an ASP.NET Web page contains the following four steps:

Tell them that in the first step the browser sends a request for an ASP.NET Web page.

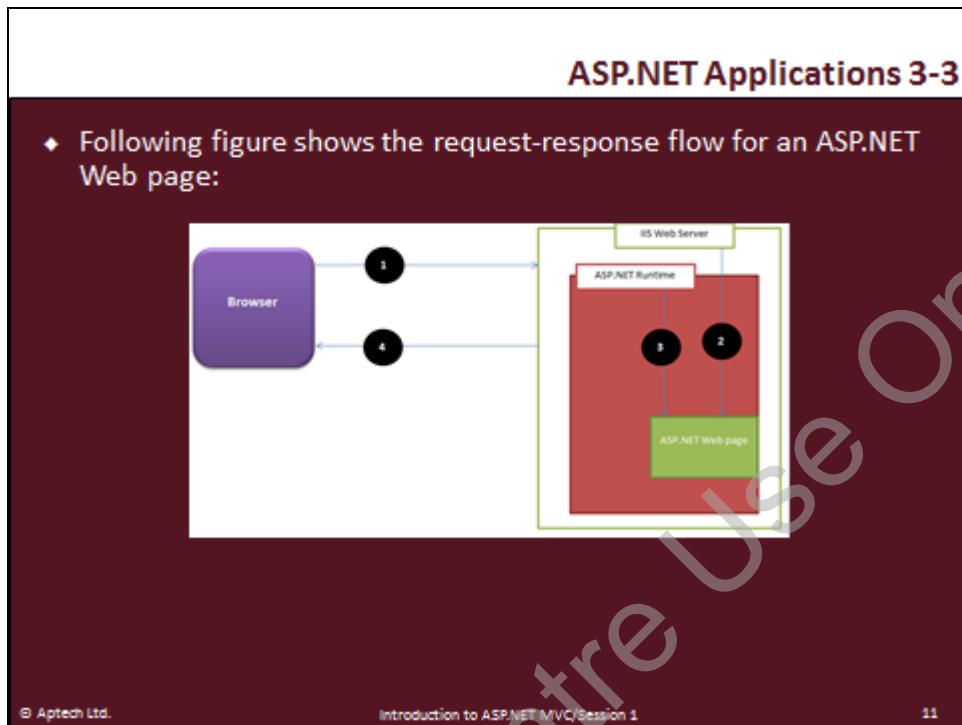
Then, tell them that when the request arrives, the IIS server intercepts the request, loads the requested file, and forwards it to the ASP.NET runtime for processing.

Next, tell them that the ASP.NET runtime that contains the ASP.NET script engine processes the requested ASP.NET page and generates the response.

Then, tell them that finally, the IIS server sends back the response to the Web server that requested the page.

**Slide 11**

Let us understand Request – Response flow.



With the help of slide 11 explain to the students how the response and request model works. Show them how the raw data or form data actually flows from the browser to the server and how the processed data is sent back to the browser by the server. Explain the role of browser and server.

## Slide 12

Let us understand the ASP.NET Web Forms.

The slide has a dark red header bar with the title 'ASP.NET Web Forms' in white. The main content area is white with black text. It contains four bullet points:

- ◆ The traditional ASP.NET Web applications gradually evolved to ASP.NET Web Forms to simplify development of dynamic Web applications.
- ◆ In ASP.NET Web Forms:
  - + You can drag and drop User Interface (UI) controls to design your UI.
  - + You can specify how the form and its control should respond at runtime.
- ◆ ASP.NET Web Forms:
  - + Uses a combination of HTML, server controls, server code, and allows users to request through browsers.
  - + Does not require you to have a hardcore developer background.
  - + Allows you to use CSS, generate semantically correct markup, and handle the development environment created for HTML elements easily.

At the bottom, there is a footer bar with the following text: © Aptech Ltd., Introduction to ASP.NET MVC/Session 1, and the number 12.

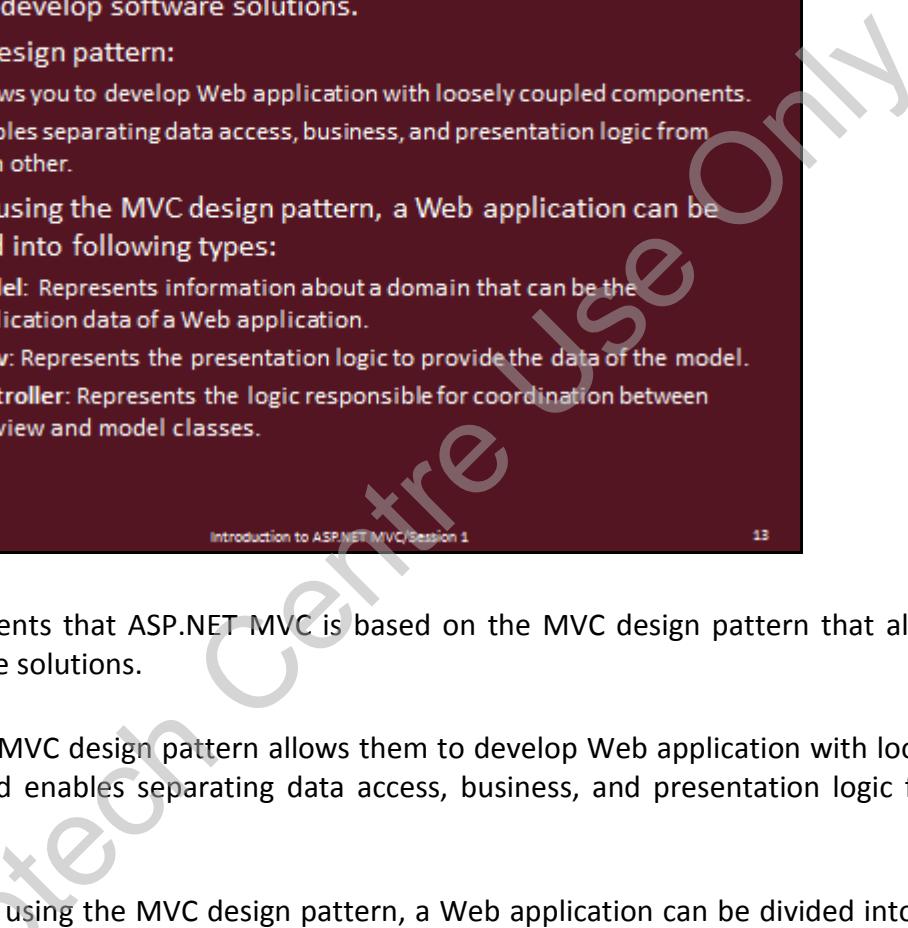
In slide 12, tell the students that the traditional ASP.NET Web applications gradually evolved to ASP.NET Web Forms to simplify development of dynamic Web applications.

Then, explain them that in ASP.NET Web Forms they can drag and drop User Interface (UI) controls to design UI and specify how the form and its control should respond at runtime. Also, mention that ASP.NET Web Forms uses a combination of HTML, server controls, server code, and allows users to request through browsers. ASP.NET Web Forms does not require users to have a hardcore developer background.

In addition, tell them that ASP.NET Web Form allows users to use CSS, generate semantically correct markup, and handle the development environment created for HTML elements easily.

### Slide 13

Let us understand the ASP.NET MVC.



### ASP.NET MVC 1-3

- ◆ ASP.NET MVC is based on the MVC design pattern that allows you to develop software solutions.
- ◆ MVC design pattern:
  - ◆ Allows you to develop Web application with loosely coupled components.
  - ◆ Enables separating data access, business, and presentation logic from each other.
- ◆ While using the MVC design pattern, a Web application can be divided into following types:
  - ◆ **Model:** Represents information about a domain that can be the application data of a Web application.
  - ◆ **View:** Represents the presentation logic to provide the data of the model.
  - ◆ **Controller:** Represents the logic responsible for coordination between the view and model classes.
  - ◆

© Aptech Ltd.      Introduction to ASP.NET MVC/Session 1      13

In slide 13, tell the students that ASP.NET MVC is based on the MVC design pattern that allows them to develop software solutions.

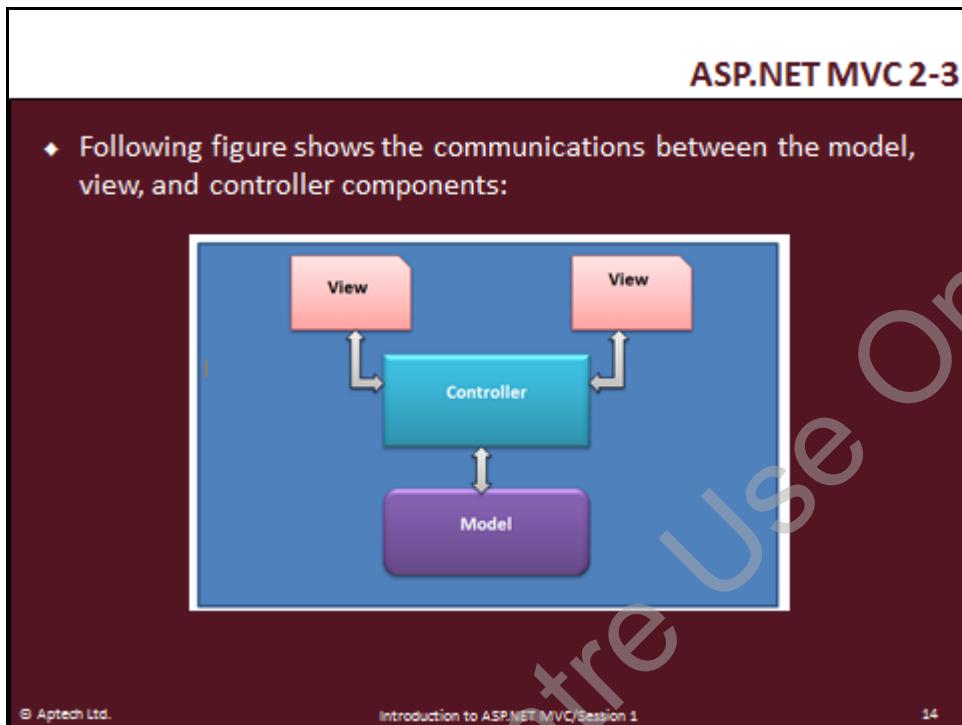
Then, explain them that MVC design pattern allows them to develop Web application with loosely coupled components and enables separating data access, business, and presentation logic from each other.

Also, mention that while using the MVC design pattern, a Web application can be divided into the following three types:

- Model that represents information about a domain that can be the application data of a Web application.
- View that represents the presentation logic to provide the data of the model.
- Controller that represents the logic responsible for coordination between the view and model classes.

**Slide 14**

Communication between Model, View, and Controller.



With the help of slide 14 explain the relationship between Model, View, and Controller. Explain what are the functions handled by each component in the MVC model. Tell them that the View displays the data, Controller resolves what actions is to be taken depending on the request information and Model generally provides the data which is used by the View before it displays anything to the user.

## Slide 15

Let us understand MVC Design Pattern.

The slide has a dark red header bar with the title 'ASP.NET MVC 3-3' in white. The main content area is white with a dark red sidebar on the left. The sidebar contains a bulleted list of benefits:

- ◆ As ASP.NET MVC is based on the MVC design pattern, it provides the following benefits:
  - + **Separation of concerns:** Enables you to ensure that various application concerns into different and independent software components.
  - + **Simplified testing and maintenance:** Enables you to test each component independently. Thus, it allows you to ensure that it is working as per the requirement of the application and then, simplifies the process of testing, maintenance, and troubleshooting procedures
  - + **Extensibility:** Enables the model to include a set of independent components that you can easily modify or replace based on the application requirement.

At the bottom of the slide, there is a footer bar with the following text: © Aptech Ltd., Introduction to ASP.NET MVC/Session 1, and a small number 15.

Use slide 15 to explain the benefits of ASP.NET MVC.

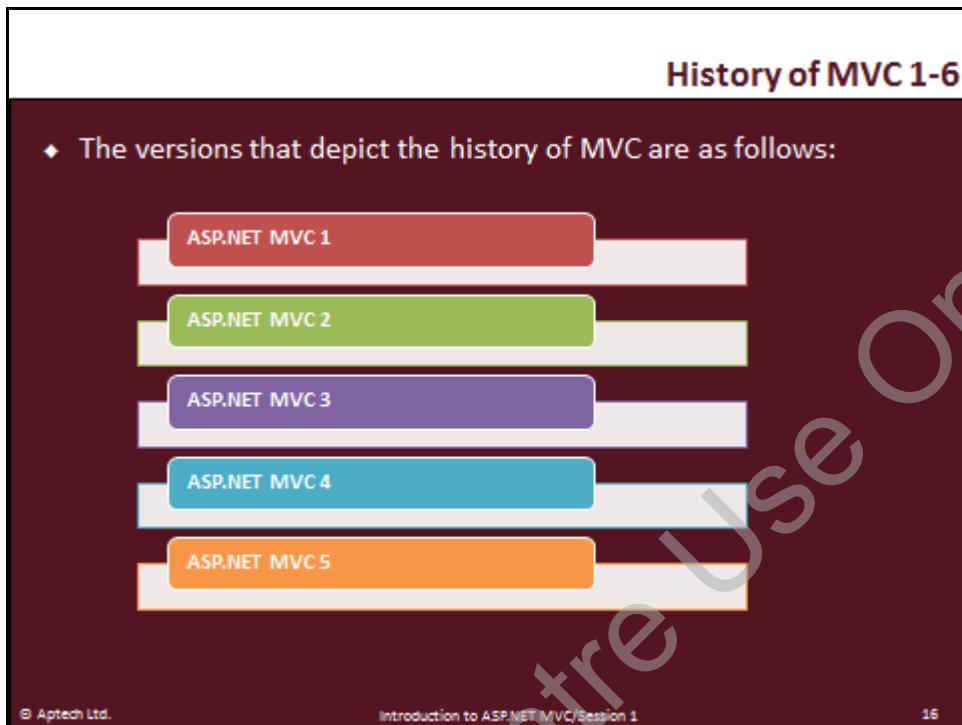
Tell the students that as ASP.NET MVC is based on the MVC design pattern, it provides the various benefits. The first one is separation of concerns. This enables users to ensure that various application concerns into different and independent software components.

The second benefit is simplified testing and maintenance. This enables testing each component independently. Thus, it allows ensuring that it is working as per the requirement of the application and then, simplifies the process of testing, maintenance, and troubleshooting procedures.

The third benefit is the extensibility. This enables the model to include a set of independent components that can be easily modified or replaced based on the application requirement.

## Slide 16

Let us understand the history of MVC.



In slide 16, tell the students that Microsoft has released different versions of ASP.NET MVC to include additional capabilities and functionalities with every newer version.

Explain the different versions of ASP.NET MVC. Tell them the major changes in each version. Chronologically, explain the new features that were added in each version. Also, explain how the evolution took place from MVC 1 to MVC 5. Discuss the need for next versions.

**Slide 17**

Let us understand features in MVC 1.

The slide has a dark red header bar with the title "History of MVC 2-6" in white. Below the header is a white content area with a red rounded rectangle containing the title "ASP.NET MVC 1". To the right of this is a list of bullet points:

- First version of ASP.NET MVC. ASP.NET MVC 1 was released on March 13, 2009.
- Targets .NET Framework 3.5.
- Supports Visual Studio 2008 and Visual Studio 2008 SP1 to develop ASP.NET MVC 1 applications.

At the bottom of the slide, there is a footer bar with the text "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "17". A large, faint watermark reading "For Aptech Centre Use Only" is diagonally across the slide.

In slide 17, tell the students that the ASP.NET MVC 1 is the first version released on March 13, 2009, which targets .NET Framework 3.5.

Tell them that this version supports Visual Studio 2008 and Visual Studio 2008 SP1 to develop ASP.NET MVC 1 applications.

## Slide 18

Let us understand features of MVC 2.

The slide has a dark red header bar with the title "History of MVC 3-6" in white. Below the header is a white content area. A green horizontal bar at the top of the content area contains the text "ASP.NET MVC 2". The main content is a bulleted list:

- Released on March 10, 2010 and targets .NET Framework 3.5 and 4.0.
- Supports Visual Studio 2008 and 2010 to develop ASP.NET MVC 2 applications.
- ASP.NET MVC 2 included the following key features:
  - Strongly typed HTML helpers means
  - Data Annotations Attribute
  - Client-side validation
  - Automatic scaffolding
  - Segregating an application into modules
  - Asynchronous controllers

At the bottom of the slide, there is footer text: "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "18".

In slide 18, tell the students that the ASP.NET MVC 2 is the second version released on March 10, 2010 and targets .NET Framework 3.5 and 4.0.

This version supports Visual Studio 2008 and 2010 to develop ASP.NET MVC 2 applications.

Tell them that ASP.NET MVC 2 included the following key features:

- Strongly typed HTML helpers means
- Data Annotations Attribute
- Client-side validation
- Automatic scaffolding
- Segregating an application into modules
- Asynchronous controllers

## Slide 19

Let us understand features of MVC 3.

The slide has a dark red header bar with the title "History of MVC 4-6" in white. Below the header is a white content area. In the top left corner of the content area, there is a purple rounded rectangle containing the text "ASP.NET MVC 3". To the right of this, a bulleted list details the features of MVC 3:

- This version of ASP.NET MVC was released on Jan 13, 2011 and targets .NET Framework 4.0.
- Supports Visual Studio 2010 to develop ASP.NET MVC 3 applications.
- This version includes the following key features:
  - Supports Razor view engine
  - Improved Support for Data Annotations
  - Dependency Resolver
  - Supports Entity Framework Code First
  - Supports ViewBag to pass data from controller to view
  - Supports Action Filters that can be applied globally
  - Support for Unobtrusive JavaScript
  - Supports jQuery Validation

At the bottom of the slide, there is a footer bar with the text "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and the number "19". A large, faint watermark reading "EduAptech Centre Only" is visible across the slide content.

In slide 19, tell the students that the ASP.NET MVC 3 is the third version released on Jan 13, 2011 and targets .NET Framework 4.0. This version supports Visual Studio 2010 to develop ASP.NET MVC 3 applications.

Tell them that ASP.NET MVC 3 included the following key features:

- Supports Razor view engine
- Improved Support for Data Annotations
- Dependency Resolver
- Supports Entity Framework Code First
- Supports ViewBag to pass data from controller to view
- Supports Action Filters that can be applied globally
- Support for Unobtrusive JavaScript
- Supports jQuery Validation

## Slide 20

Let us understand features of MVC 4.

The slide has a dark red header bar with the title "History of MVC 5-6" in white. Below the header is a blue rectangular callout box containing the text "ASP.NET MVC 4". The main content area is white with a dark red sidebar on the left. The sidebar contains the following text:

- This version of ASP.NET MVC was released on Aug 15, 2012 and targets .NET Framework 4.0 and 4.5.
- Supports Visual Studio 2010 SP1 and Visual Studio 2012 to develop ASP.NET MVC 4 applications.
- This version introduced the following key features:
  - ASP.NET Web API
  - Asynchronous Controllers with Task support
  - Bundling and minification
  - Support for the Windows Azure SDK

At the bottom of the slide, there is a footer bar with the text "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "20". A large watermark reading "For Aptech Centre Use Only" is diagonally across the slide.

In slide 20, tell the students that the ASP.NET MVC 4 is the fourth version released on Aug 15, 2012 and targets .NET Framework 4.0 and 4.5. This version supports Visual Studio 2010 SP1 and Visual Studio 2012 to develop ASP.NET MVC 4 applications.

Tell them that ASP.NET MVC 4 included the following key features:

- ASP.NET Web API
- Asynchronous Controllers with Task support
- Bundling and minification
- Support for the Windows Azure SDK

## Slide 21

Let us understand history and features of MVC 5.

The slide has a dark red header bar with the title "History of MVC 6-6" in white. Below the header is a white content area with an orange rounded rectangle containing the heading "ASP.NET MVC 5". To the right of this is a list of bullet points:

- This version of ASP.NET MVC was released on 17 October 2013 and targets .NET Framework 4.5 and 4.5.1.
- Supports Visual Studio 2013 to develop ASP.NET MVC 5 applications.
- This version introduced the following new features:
  - ASP.NET Identity
  - ASP.NET Web API 2

At the bottom of the slide, there is footer text: "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "21". A large watermark reading "For Aptech Centre Only" is diagonally across the slide.

In slide 21, tell the students that the ASP.NET MVC 5 is the latest version released on 17 October 2013 and targets .NET Framework 4.5 and 4.5.1. This version supports Visual Studio 2013 to develop ASP.NET MVC 5 applications.

Tell them that ASP.NET MVC 5 included the following key features:

- ASP.NET Identity
- ASP.NET Web API 2

**Slide 22**

Let us understand the architecture of ASP.NET MVC application.

### Architecture of ASP.NET MVC Application 1-3

- ◆ The basic architecture of an ASP.NET MVC application involves the following components:
  - ◆ MVC Framework
  - ◆ Route engine
  - ◆ Route configuration
  - ◆ Controller
  - ◆ Model
  - ◆ View engine
  - ◆ View
- ◆ Each of these preceding components communicates to process requests coming to an ASP.NET MVC application.
- ◆ The process of handling an incoming request involves a series of steps that these components perform.

© Aptech Ltd.      Introduction to ASP.NET MVC/Session 1      22

In slide 22, tell the students that the basic architecture of an ASP.NET MVC application involves different components.

Explain to them that the components that the architecture of an ASP.NET MVC application involves are:

- MVC Framework
- Route engine
- Route configuration
- Controller
- Model
- View engine
- View

Tell them that each of these components communicates to process requests coming to an ASP.NET MVC application.

Also, explain that the process of handling an incoming request involves a series of steps that these components perform.

## Slide 23

Let us understand architecture of ASP.NET MVC application.

### Architecture of ASP.NET MVC Application 2-3

- ◆ The steps that the components of the ASP.NET MVC Framework performs while handling an incoming request includes:
  - ◆ The browser sends a request to an ASP.NET MVC application.
  - ◆ The MVC Framework forwards the request to the routing engine.
  - ◆ The route engine checks the route configuration of the application for an appropriate controller to handle the request.
  - ◆ When a controller is found it is invoked.
  - ◆ When a controller is not found the route engine indicates that the controller has not been found and the MVC Framework communicates this as an error to the browser.
  - ◆ The controller communicates with the model.
  - ◆ The controller requests a view engine for a view based on the data of the model.
  - ◆ The view engine returns the result to the controller.
  - ◆ The controller sends back the result as an HTTP response to the browser.

© Aptech Ltd.      Introduction to ASP.NET MVC/Session 1      23

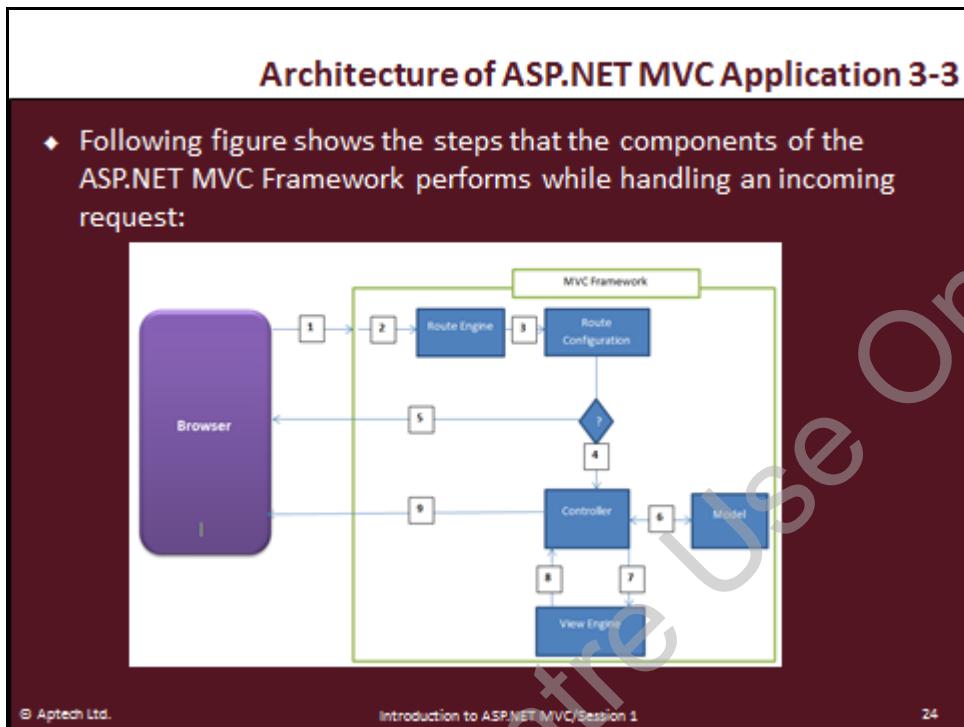
In slide 23, explain to the students that there are several steps that the components of the ASP.NET MVC Framework perform while handling an incoming request.

Tell them that the components of the ASP.NET MVC Framework perform the following steps while handling an incoming request:

- The browser sends a request to an ASP.NET MVC application.
- The MVC Framework forwards the request to the routing engine.
- The route engine checks the route configuration of the application for an appropriate controller to handle the request.
- When a controller is found it is invoked.
- When a controller is not found the route engine indicates that the controller has not been found and the MVC Framework communicates this as an error to the browser.
- The controller communicates with the model.
- The controller requests a view engine for a view based on the data of the model.
- The view engine returns the result to the controller.
- The controller sends back the result as an HTTP response to the browser.

**Slide 24**

Let us learn how to handle incoming request.



Give them detail idea about what goes on behind the screen once a request is received from the browser. With the help of image shown on slide 24, explain various steps involved in the process. Also, explain the role of each step in detail so that students get an idea about the entire MVC processing.

## Slide 25

Let us understand the supporting technologies.

**Supporting Technologies**

- ◆ ASP.NET MVC application supports various technologies to create dynamic and responsive Web application.
- ◆ Some of the supporting technologies that you can use while creating an ASP.NET MVC application are as follows:

- JavaScript
- JQuery
- Asynchronous JavaScript and XML (AJAX)
- IIS
- Windows Azure

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 25

In slide 25, tell the students that ASP.NET MVC application supports various technologies to create dynamic and responsive Web application.

## Slide 26

Let us understand what is JavaScript.

The slide has a dark red header bar with the word 'JavaScript' in white. The main content area is white with a dark red sidebar on the left. A large watermark 'For Aptech Centre Use Only' is diagonally across the slide. The content is as follows:

**JavaScript**

- ◆ Is a client-side scripting language that enables a Web application to respond to user requests without interacting with a Web server.
- ◆ Allows creating responsive Web applications that enhances user experience.
- ◆ Allows implementing functionalities, such as an easy to use UI, quick response to the user's request, and run in all available browsers.
- ◆ Allows your Web applications to respond to user requests without interacting with a Web server.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 25

In slide 26, tell the students that JavaScript is a client-side scripting language that enables a Web application to respond to user requests without interacting with a Web server.

JavaScript allows creating responsive Web applications that enhances user experience. It also allows implementing functionalities, such as an easy to use UI, quick response to the user's request, and run in all available browsers.

Tell them that JavaScript allows Web applications to respond to user requests without interacting with a Web server.

## Slide 27

Let us understand about JQuery.

The slide has a dark red header bar with the title "JQuery 1-3" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following bullet points:

- ◆ JQuery:
  - ◆ Is a JavaScript library that simplifies the client-side scripting of HTML.
  - ◆ Is used in views to make your views responsive and dynamic.
  - ◆ Allows you to perform client-side validation of forms.
  - ◆ Provides several plugins that enable you to enhance the UI elements of a view.
- ◆ When you create an ASP.NET MVC project in Visual Studio 2013, the project automatically includes the jQuery libraries within the **Scripts** folder.

At the bottom of the slide, there is a footer bar with the following text: "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "27".

In slide 27, tell the students that the JQuery is a JavaScript library that simplifies the client-side scripting of HTML. It is used in views to make your views responsive and dynamic.

Tell them that JQuery allows users to perform client-side validation of forms and provides several plugins that enable user to enhance the UI elements of a view.

Explain to them that when they create an ASP.NET MVC project in Visual Studio 2013, the project automatically includes the jQuery libraries within the Scripts folder.

## Slide 28

Let us check where and how the JQuery libraries are shown.

JQuery 2-3

- ◆ Following figure shows the jQuery libraries in the **Scripts** folder of an ASP.NET MVC project:

The screenshot shows the 'Solution Explorer' window in Visual Studio. The 'Scripts' folder is expanded, revealing numerous jQuery files. A red box highlights the 'references.js' file and the list of jQuery files. The files listed are:

- 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

© Aptech Ltd. Introduction to ASP.NET MVC Session 1 28

With the help of slide 28, tell the students what is JQuery and which are the supporting files in MVC model. Tell them the difference between each version of JQuery files and need to include it in the current project. Also, tell them that these files are included by default even if you use the JQuery or don't. Each project will automatically contain these JQuery files the moment the project is created. If the user is not using any of the JQuery function while developing his/her Website then these files can be deleted to reduce the space used by the Website.

## Slide 29

Let us understand the use of JQuery library in the View.

### JQuery 3-3

- ◆ To use a jQuery library in a view, you need to refer the library from the view.
- ◆ You can use the following code snippet to refer to a jQuery library from a view:

**Code Snippet:**

```
<script src="/Scripts/jquery-1.7.1.min.js"
type="text/javascript"></script>
<!-- JQuery Code -->
</script>
```

© Aptech Ltd.      Introduction to ASP.NET MVC Session 1      29

In slide 29, tell students that to use a jQuery library in a view, they need to refer the library from the view.

Explain that they can do this using the code displayed on the slide.

#### Additional Information:

Instead of referring the jQuery library in each view, you can refer the library in a layout file and use the layout file in the views. The views can then automatically use the jQuery libraries.

## Slide 30

Let us understand AJAX.

AJAX

- ◆ **AJAX:**
  - ◆ Is a Web development technique that is used to create interactive applications.
  - ◆ Allows you to asynchronously retrieve data in the background of an ASP.NET MVC application without interfering with the display and behavior of the existing view.
  - ◆ Enables the users to work continuously on the application without being affected by the responses received from the server.
- ◆ JavaScript forms an integral part of the AJAX-enabled Web applications because these applications process most of the requests on the client side, unless there is a need to connect to the Web server.
- ◆ The XMLHttpRequest object is one of the primary objects used by JavaScript because it enables asynchronous communication between the client and the server.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 30

In slide 30, tell the students that AJAX is a Web development technique that is used to create interactive applications. AJAX also allows asynchronously retrieve data in the background of an ASP.NET MVC application without interfering with the display and behavior of the existing view.

AJAX also enables the users to work continuously on the application without being affected by the responses received from the server.

Tell the student that JavaScript forms an integral part of the AJAX-enabled Web applications because these applications process most of the requests on the client side, unless there is a need to connect to the Web server.

Then, tell them that the XMLHttpRequest object is one of the primary objects used by JavaScript because it enables asynchronous communication between the client and the server.

## Slide 31

Let us understand IIS.

The slide has a dark red header bar with the text 'IIS 1-5' in white. The main content area is white with a dark red sidebar on the left. A large watermark 'For Aptech Centre Use Only' is diagonally across the slide. The content is as follows:

- ◆ An ASP.NET MVC application requires a Web server that enables handling HTTP requests and creates responses.
- ◆ When you request for a Web page on a browser, you type a URL on a browser, for example, `http://mvctest.com/index.html`.
- ◆ This URL consists of the following parts:
  - ◆ `http`: A protocol to use for exchanging request and response.
  - ◆ `mvctest.com`: A domain name that maps to a unique IP address
  - ◆ `index.html`: A file that you are requesting.

At the bottom, there is a footer bar with the text '© Aptech Ltd.', 'Introduction to ASP.NET MVC/Session 1', and '31'.

In slide 31, explain to the students that an ASP.NET MVC application requires a Web server that enables handling HTTP requests and creates responses.

When they request for a Web page on a browser, they type a URL on a browser, for example, `http://mvctest.com/index.html`.

Explain the following parts that this URL consists of:

- `http`: A protocol to use for exchanging request and response.
- `mvctest.com`: A domain name that maps to a unique IP address.
- `index.html`: A file that you are requesting.

## Slide 32

Let us understand how the communication between the browser and the server.

IIS 2-5

- ◆ Following figure shows the communication between a browser and the server:

```
graph LR; Browser[Browser] -- "Sending request for a page" --> WebServer[Web Server]; WebServer -- "Sends back the requested page" --> Browser;
```

- ◆ In this figure:
  - ◆ The browser creates a connection with a Web server where the IP address is registered based on the IP address.
  - ◆ Next, the browser uses the HTTP protocol to send a request to the server, asking for the file.
  - ◆ The server, searches for the file and sends back the content of the file as HTML markup to the browser.
  - ◆ The browser parses the HTML markup and displays the output to you.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 32

In slide 32, refer to the figure displayed on the slide and tell them that in this figure the browser creates a connection with a Web server where the IP address is registered based on the IP address.

Next, the browser uses the HTTP protocol to send a request to the server, asking for the file. Then, the server, searches for the file and sends back the content of the file as HTML markup to the browser.

Finally, the browser parses the HTML markup and displays the output.

### Slide 33

Let us understand the main features of IIS Server.

The slide has a dark red header bar with the text 'IIS 3-5' in white. The main content area is white with a dark red sidebar on the left. The sidebar contains a single bullet point: '◆ IIS:' followed by a list of features. At the bottom of the slide, there is a dark red footer bar with the text '© Aptech Ltd.', 'Introduction to ASP.NET MVC/Session 1', and '33'.

- ◆ IIS:
  - + Is a product of Microsoft and is designed to deliver information in high speed and securely.
  - + Acts as a platform that you can target to extend the capabilities of the Internet standards.
  - + Has been designed to be modular that allows selecting only those modules that meet your specific requirements while installing IIS.
  - + Some of the modules that you need to host ASP.NET MVC applications are:
    - Content modules: Enables controlling how content is delivered to client.
    - Security modules: Enables implementing security through various authentication mechanisms, authorization based on URLs, and filtering requests.
    - Compression module: Enables compressing responses sent to client browser using compression standard, such as Gzip.
    - Caching modules: Enables caching content and delivering the cached content in subsequent requests for the same resource.
    - Logging and Diagnostics modules: Enables logging request processing information and response status for diagnostic purposes.

Show slide 33 and tell the students that IIS is a product of Microsoft and is designed to deliver information in high speed and securely. This product acts as a platform that the students can target to extend the capabilities of the Internet standards.

Tell them that IIS has been designed to be modular that allows selecting only those modules that meet user's specific requirements while installing IIS.

Explain the following modules that the students need to host ASP.NET MVC applications:

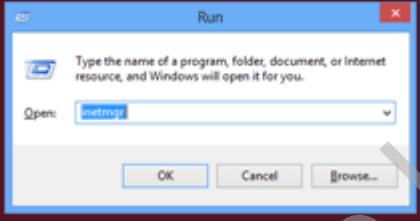
- Content module enables controlling how content is delivered to client.
- Security modules enable implementing security through various authentication mechanisms, authorization based on URLs, and filtering requests.
- Compression module enables compressing responses sent to client browser using compression standard, such as Gzip.
- Caching module enables caching content and delivering the cached content in subsequent requests for the same resource.
- Logging and Diagnostics module enables logging request processing information and response status for diagnostic purposes.

## Slide 34

Let us understand deployment and management of an ASP.NET MVC application.

**IIS 4-5**

- ◆ You can deploy and manage ASP.NET MVC applications in IIS using IIS Manager.
- ◆ To access IIS Manager, you need to perform the following steps:
  1. Ensure that IIS is installed in your computer.
  2. Press the Windows+R key combination. The Run dialog box appears.
  3. Type `inetmgr`, as shown in the following figure:



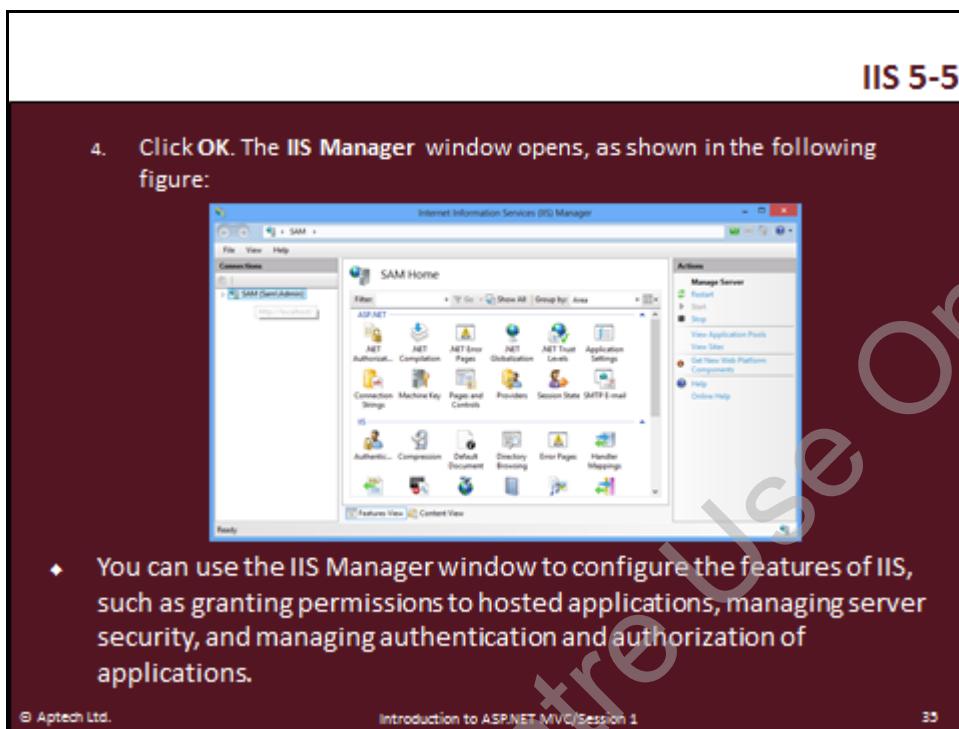
© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 34

In slide 34, tell the students that they can deploy and manage ASP.NET MVC applications in IIS using IIS Manager.

Explain them the steps to access IIS Manager as displayed on the slide.

### Slide 35

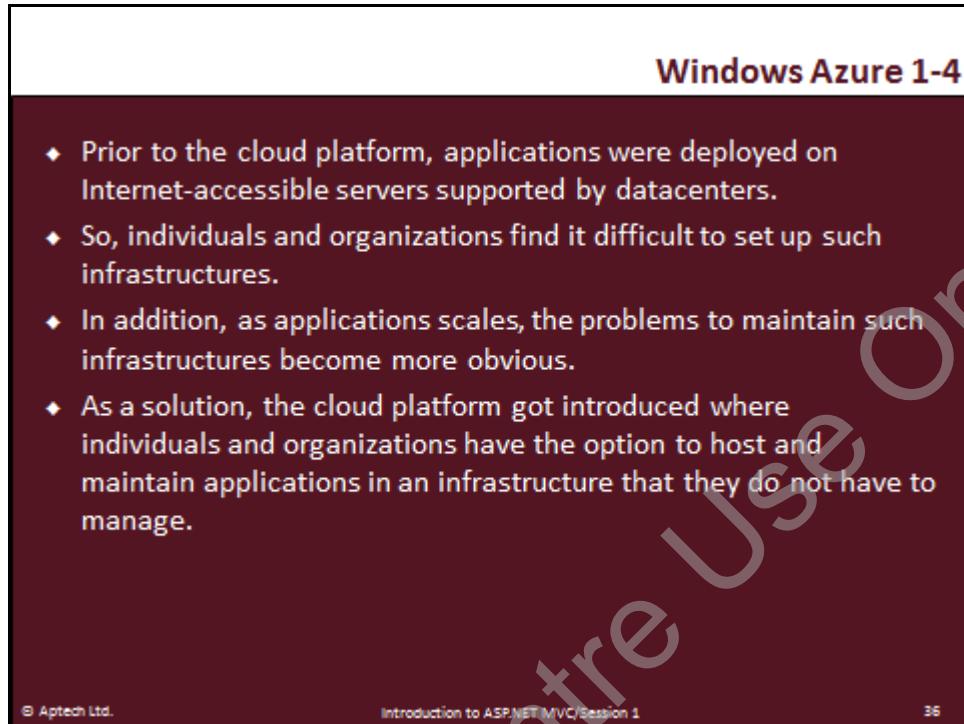
Let us understand management of Websites using IIS Manager.



In slide 35, explain to the students that they can use the IIS Manager window to configure the features of IIS, such as granting permissions to hosted applications, managing server security, and managing authentication and authorization of applications.

## Slide 36

Let us understand Windows Azure.



The slide has a dark red header bar with the text "Windows Azure 1-4" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following text:

- ◆ Prior to the cloud platform, applications were deployed on Internet-accessible servers supported by datacenters.
- ◆ So, individuals and organizations find it difficult to set up such infrastructures.
- ◆ In addition, as applications scales, the problems to maintain such infrastructures become more obvious.
- ◆ As a solution, the cloud platform got introduced where individuals and organizations have the option to host and maintain applications in an infrastructure that they do not have to manage.

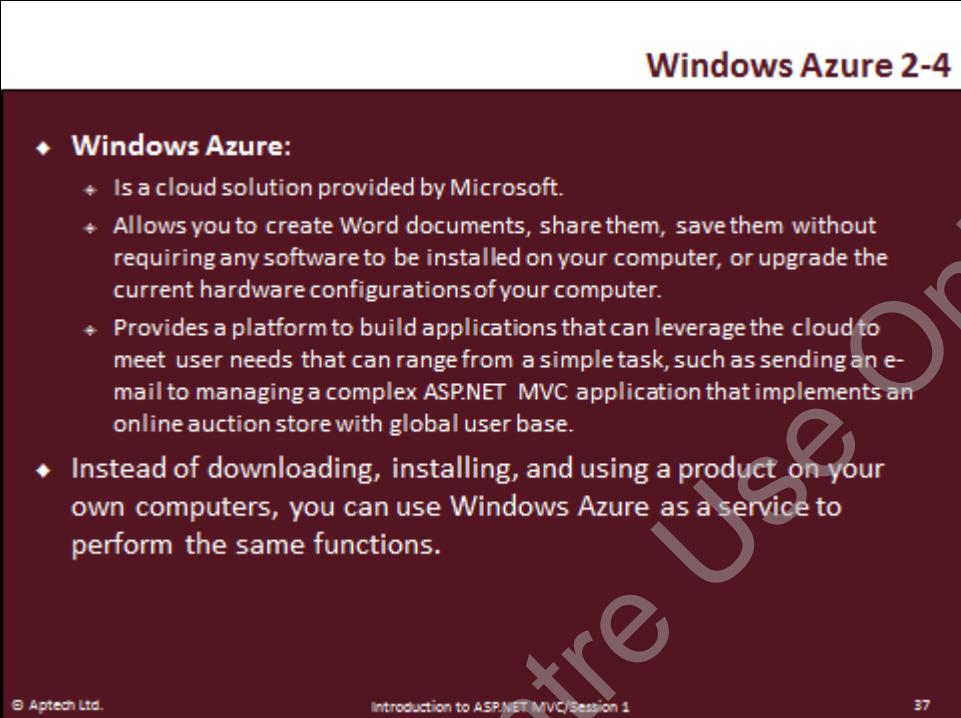
At the bottom of the slide, there is a footer bar with the following text:  
© Aptech Ltd.      Introduction to ASP.NET MVC/Session 1      36

In slide 36, explain to the students that prior to the cloud platform, applications were deployed on Internet-accessible servers supported by datacentres. So, individuals and organizations find it difficult to set up such infrastructures. In addition, as applications scales, the problems to maintain such infrastructures become more obvious.

As a solution, the cloud platform got introduced where individuals and organizations have the option to host and maintain applications in an infrastructure that they do not have to manage.

## Slide 37

Let us understand the features of Windows Azure.



The slide has a dark red header bar with the title "Windows Azure 2-4" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following text:

- ◆ **Windows Azure:**
  - ◆ Is a cloud solution provided by Microsoft.
  - ◆ Allows you to create Word documents, share them, save them without requiring any software to be installed on your computer, or upgrade the current hardware configurations of your computer.
  - ◆ Provides a platform to build applications that can leverage the cloud to meet user needs that can range from a simple task, such as sending an e-mail to managing a complex ASP.NET MVC application that implements an online auction store with global user base.
  - ◆ Instead of downloading, installing, and using a product on your own computers, you can use Windows Azure as a service to perform the same functions.

At the bottom of the slide, there is a footer bar with the following text:  
© Aptech Ltd.      Introduction to ASP.NET MVC/Session 1      37

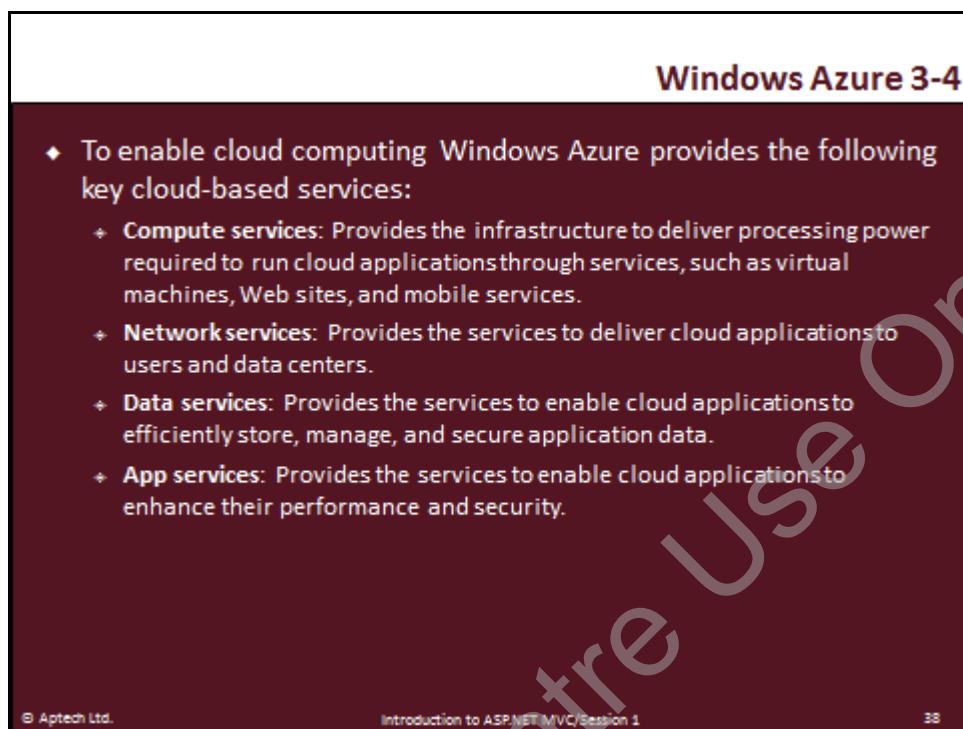
In slide 37, explain to the students that Windows Azure is a cloud solution provided by Microsoft. It allows user to create Word documents, share them, save them without requiring any software to be installed on a computer, or upgrade the current hardware configurations of your computer.

Tell them that Windows Azure provides a platform to build applications that can leverage the cloud to meet user needs that can range from a simple task, such as sending an e-mail to manage a complex ASP.NET MVC application that implements an online auction store with global user base.

Explain the student that instead of downloading, installing, and using a product on computers, user can use Windows Azure as a service to perform the same functions.

## Slide 38

Let us understand Windows Azure services that enable cloud computing.



The slide has a dark red header bar with the title "Windows Azure 3-4" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains a single bullet point: "To enable cloud computing Windows Azure provides the following key cloud-based services:" followed by four sub-points: "Compute services", "Network services", "Data services", and "App services". At the bottom of the slide, there is a footer bar with three items: "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "38". A large, faint watermark reading "For Aptech Centre Use Only" is diagonally across the slide.

- ◆ To enable cloud computing Windows Azure provides the following key cloud-based services:
  - + **Compute services:** Provides the infrastructure to deliver processing power required to run cloud applications through services, such as virtual machines, Web sites, and mobile services.
  - + **Network services:** Provides the services to deliver cloud applications to users and data centers.
  - + **Data services:** Provides the services to enable cloud applications to efficiently store, manage, and secure application data.
  - + **App services:** Provides the services to enable cloud applications to enhance their performance and security.

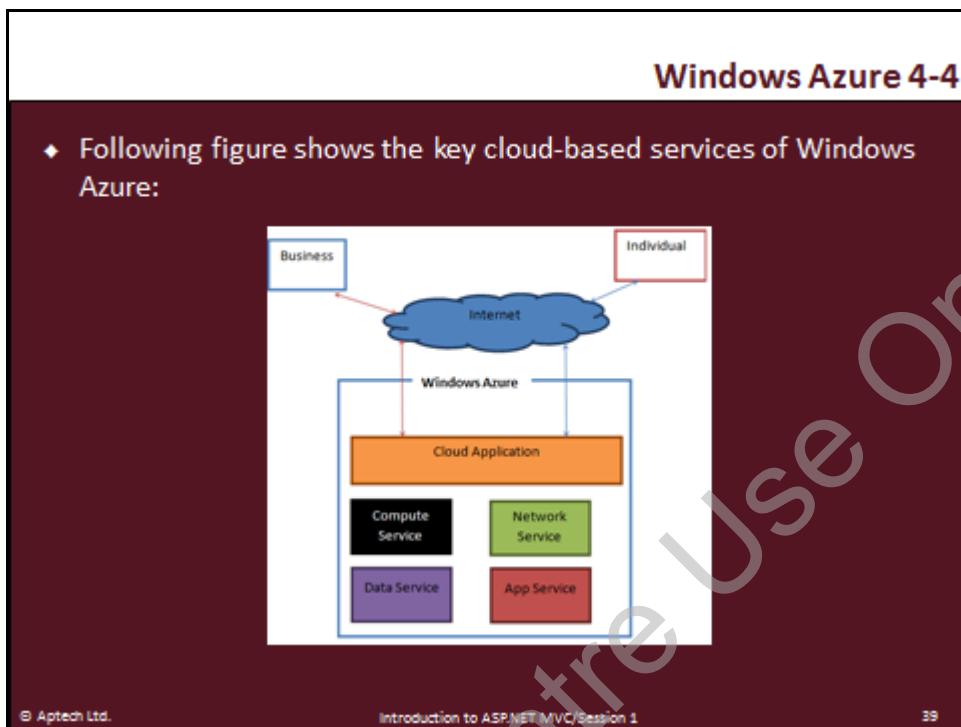
In slide 38, explain to the students that to enable cloud computing Windows Azure provides the various key cloud-based services.

Explain the students about the following cloud-based services that Windows Azure provides:

- Compute services provides the infrastructure to deliver processing power required to run cloud applications through services, such as virtual machines, Web sites, and mobile services.
- Network services provide the services to deliver cloud applications to users and data centers.
- Data services provide the services to enable cloud applications to efficiently store, manage, and secure application data.
- App services provide the services to enable cloud applications to enhance their performance and security.

### Slide 39

Let us understand the Cloud based services in Windows Azure.



In slide 39, the figure shows the cloud based services available in Windows Azure. You can learn from the figure that there are four main services working in the background. The Compute Service, Network Service, Data Service, App Service. The Compute Service providing computing facilities, Network Service provides the functionality that is needed to access the data smoothly. The Data Service provides the functionality that is necessary for accessing databases. And the App Services provides functionality that provides functionality that is necessary for properly managing and configuring the applications.

## Slide 40

Let us understand MVC Support in Visual Studio 2013.

The slide has a dark red header bar with the title 'MVC Support in Visual Studio 2013' in white. Below the header is a white content area containing a bulleted list. The list starts with a diamond symbol followed by 'Visual Studio 2013:' and then two plus signs indicating further details. The content of the slide is partially obscured by a large, diagonal watermark reading 'For Aptech Centre Use Only'.

- ◆ Visual Studio 2013:
  - + Simplifies the process of creating ASP.NET MVC applications by providing various in-built templates.
  - + Provides an MVC template that automatically creates an MVC application structure with the basic files to run the application.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 40

In slide 40, explain to the students that Visual Studio 2013 simplifies the process of creating ASP.NET MVC applications by providing various in-built templates.

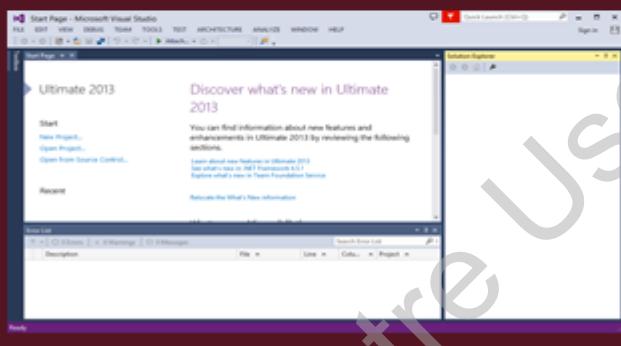
This IDE provides an MVC template that automatically creates an MVC application structure with the basic files to run the application.

## Slide 41

Let us understand how to create ASP.NET MVC project in Visual Studio 2013.

### Creating an ASP.NET MVC Project 1-5

- ♦ To create an ASP.NET MVC application using Visual Studio 2013, you need to perform the following tasks:
  1. Press the Windows+Q key on keyboard.
  2. In the Search box that appears, start typing Visual Studio 2013. The Visual Studio 2013 icon appears under the Apps section.
  3. Double-click the Visual Studio 2013 icon. The Start Page of Visual Studio 2013 appears, as shown in the following figure:

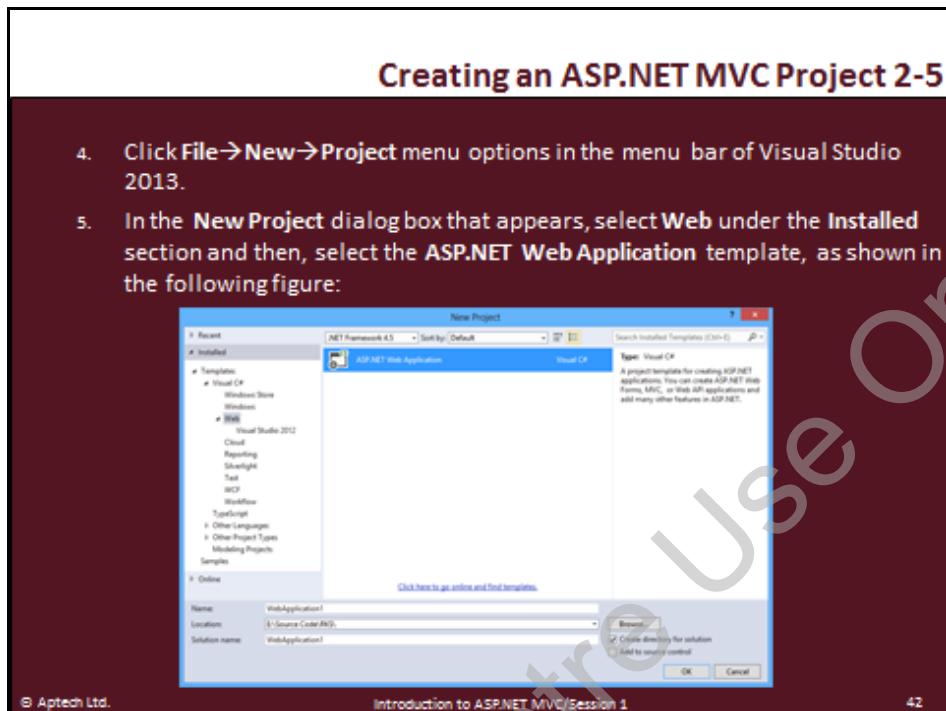


© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 41

In slide 41, explain to the students the steps as displayed on the slide to create an ASP.NET MVC project in Visual Studio 3013.

## Slide 42

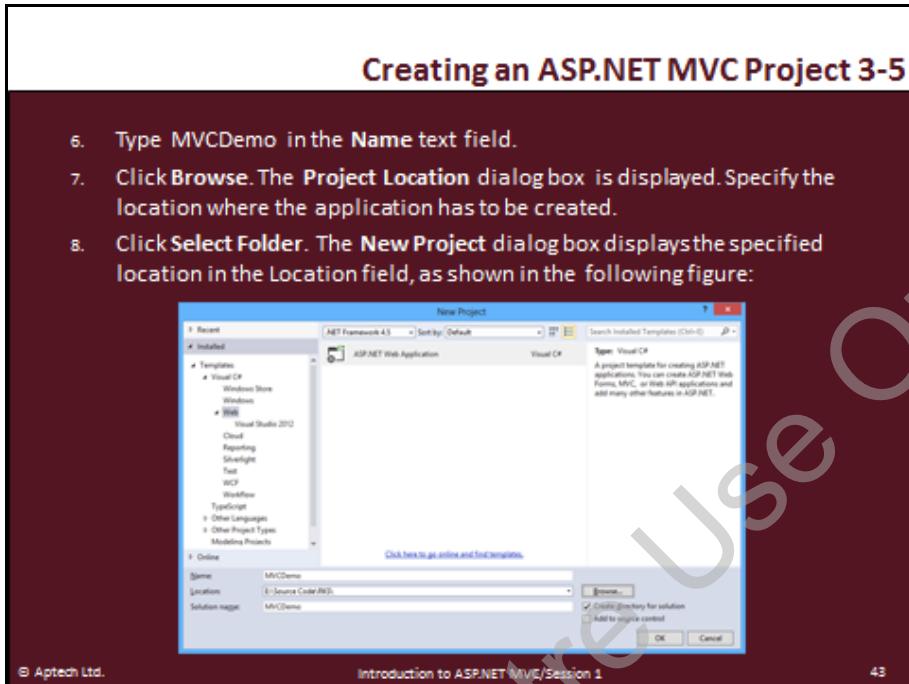
Let us learn to create an ASP.NET MVC Project.



In slide 42, explain to the students the steps as displayed on the slide to select the ASP.NET Web Application template.

### Slide 43

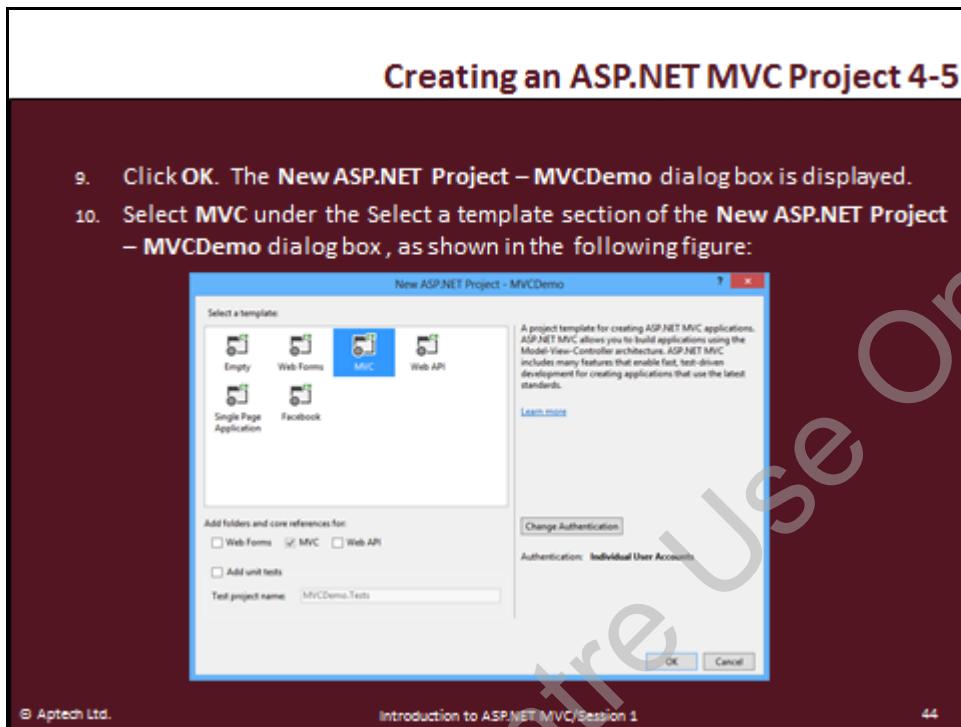
Let us learn how to enter project location, name, and type of project.



In slide 43, explain to the students the steps as displayed on the slide to specify the project location.

## Slide 44

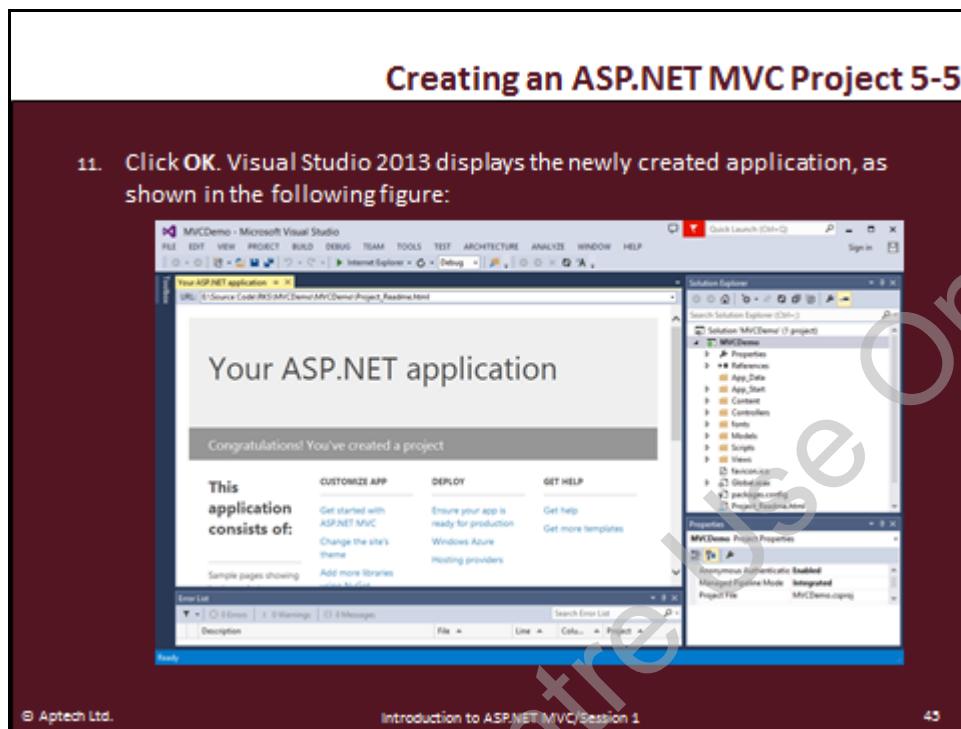
Let us understand how to select MVC template from the given list.



In slide 44, explain to the students the steps as displayed on the slide select MVC as the project template.

## Slide 45

Let us understand how to show the list of newly created MVC applications.



In slide 45, explain to the students the steps as displayed on the slide to complete the process of project creation.

## Slide 46

Let us understand the structure of an ASP.NET MVC Project.

### Structure of an ASP.NET MVC Project 1-2

- ◆ When you create an ASP.NET MVC Web application in Visual Studio 2013, it automatically adds several files and folders to the project.
- ◆ Following figure shows the files and folders that Visual Studio 2013 creates when you create an ASP.NET MVC application:

The screenshot shows the Visual Studio Solution Explorer with the title "Structure of an ASP.NET MVC Project 1-2". The project "MVCDemo" is selected. Inside the project, there are several files and folders: App\_Data, Content, Controllers, Fonts, Models, Scripts, and Views. Under Views, there are favicon.ico, Global.asax, packages.config, Project\_Readme.html, Startup.cs, and Web.config. Other files include Startup.Auth.cs, RouteConfig.cs, FilterConfig.cs, and BundleConfig.cs. The "App\_Data" folder is highlighted with a blue selection bar.

Use slide 46 and tell the students that when they create an ASP.NET MVC Web application in Visual Studio 2013, it automatically adds several files and folders to the project.

You can refer the figure displayed on the slide that shows the files and folders that Visual Studio 2013 creates when they create an ASP.NET MVC application.

## Slide 47

Let us understand the top-level directory structure of an ASP.NET MVC application.

### Structure of an ASP.NET MVC Project 2-2

- ◆ The top-level directory structure of an ASP.NET MVC application contains the following folders:
  - ◆ **Controllers:** Contains the Controller classes that handle URL requests.
  - ◆ **Models:** Contains the classes that represent and manipulate data and business objects.
  - ◆ **Views:** Contains the UI template files that are responsible for rendering output, such as HTML.
  - ◆ **Scripts:** Contains the JavaScript library files.
  - ◆ **Images:** Contains the images that you need to use in your application.
  - ◆ **Content:** Contains the CSS and other site content, other than scripts and images.
  - ◆ **Filters:** Contains the filter code.
  - ◆ **App\_Data:** Contains data files that you need to read/write.
  - ◆ **App\_Start:** Contains the files containing configuration code that you can use features like Routing, Bundling, and Web API.

© Aptech Ltd.      Introduction to ASP.NET MVC/Session 1      47

Use slide 47 to explain the students about the top-level directory structure of an ASP.NET MVC application that contains the following folders:

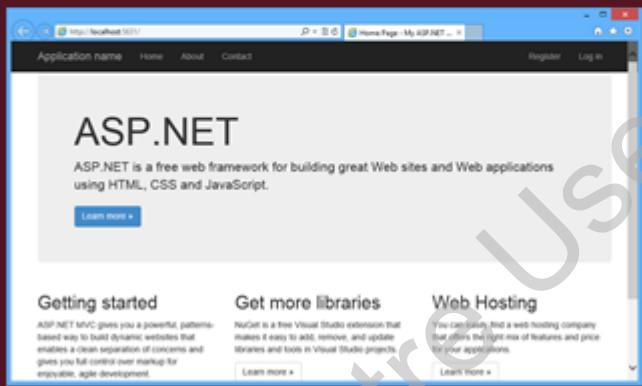
- Controller folders contain the Controller classes that handle URL requests.
- Models folder contains the classes that represent and manipulate data and business objects.
- Views folder contains the UI template files that are responsible for rendering output, such as HTML.
- Scripts folder contains the JavaScript library files.
- Images folder contains the images that user need to use in application.
- Content folder contains the CSS and other site content, other than scripts and images.
- Filters folder contains the filter code.
- App\_Data folder contains data files that user need to read/write.
- App\_Start folder contains the files containing configuration code that user can use features such as Routing, Bundling, and Web API.

## Slide 48

Let us understand the project execution.

### Executing ASP.NET MVC Project

- ◆ To execute an application created in Visual Studio 2013 you need to click **Debug → Start without debugging** from the menu bar. The browser will display the output of the default application, as shown in the following figure:



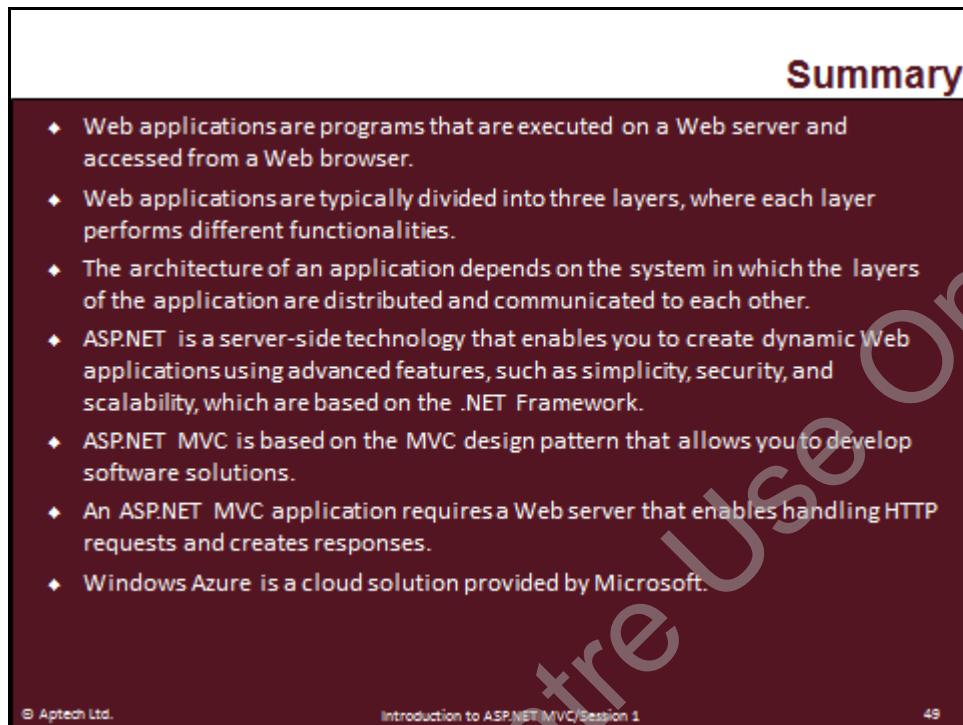
The screenshot shows a Microsoft Edge browser window with the title 'Executing ASP.NET MVC Project'. The main content area displays the 'ASP.NET' welcome page. The page includes a heading 'ASP.NET', a brief description stating 'ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS and JavaScript.', and a 'Learn more' button. Below this, there are three sections: 'Getting started', 'Get more libraries', and 'Web Hosting', each with a 'Learn more' button. At the bottom of the browser window, the status bar shows '© Aptech Ltd.' and 'Introduction to ASP.NET MVC/Session 1'. The slide number '48' is visible in the bottom right corner.

In slide 48, tell the students that to execute an application created in Visual Studio 2013 they need to click **Debug → Start without debugging** from the menu bar.

You can refer the figure as displayed on the slide that shows the browser that displaying the output of the default application.

## Slide 49

Let us summarize the session.



The slide has a dark blue header with the word "Summary" in white. The main content area is white with a dark blue border. It contains a bulleted list of nine points about web applications and ASP.NET MVC. At the bottom, there is a dark blue footer bar with white text: "© Aptech Ltd.", "Introduction to ASP.NET MVC/Session 1", and "49". A large, semi-transparent watermark reading "Aptech Content Only" is diagonally across the slide.

**Summary**

- ◆ Web applications are programs that are executed on a Web server and accessed from a Web browser.
- ◆ Web applications are typically divided into three layers, where each layer performs different functionalities.
- ◆ The architecture of an application depends on the system in which the layers of the application are distributed and communicated to each other.
- ◆ ASP.NET is a server-side technology that enables you to create dynamic Web applications using advanced features, such as simplicity, security, and scalability, which are based on the .NET Framework.
- ◆ ASP.NET MVC is based on the MVC design pattern that allows you to develop software solutions.
- ◆ An ASP.NET MVC application requires a Web server that enables handling HTTP requests and creates responses.
- ◆ Windows Azure is a cloud solution provided by Microsoft.

© Aptech Ltd. Introduction to ASP.NET MVC/Session 1 49

In slide 49, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- Web applications are programs that are executed on a Web server and accessed from a Web browser.
- Web applications are typically divided into three layers, where each layer performs different functionalities.
- The architecture of an application depends on the system in which the layers of the application are distributed and communicated to each other.
- ASP.NET is a server-side technology that enables you to create dynamic Web applications using advanced features, such as simplicity, security, and scalability, which are based on the .NET Framework.
- ASP.NET MVC is based on the MVC design pattern that allows developing software solutions.
- An ASP.NET MVC application requires a Web server that enables handling HTTP requests and creates responses.
- Windows Azure is a cloud solution provided by Microsoft.

### **1.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. What is AJAX?
2. In what situation will you use jQuery?
3. In what situation will you consider migrating from a Two-tier to a Three-tier architecture?

## Session 2 -

### Controllers in ASP.NET MVC

#### **2.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth. You can revise the topics that were already covered in the previous session. Ask students what they can recall from the last session. Ask some questions related to MVC topics that were covered previously. It will give you fair idea about what the students already know.

#### **2.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe controllers
- Describe how to work with action methods
- Explain how to invoke action methods
- Explain routing requests
- Describe URL patterns

#### **2.1.2 Teaching Skills**

To teach this session successfully, you must know about the controller and action methods. You should know how to handle action methods in an ASP.NET MVC application. You should also be aware of routing requests and URL patterns.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

### Overview of the Session

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the controllers in an ASP.NET MVC application. They will learn about how to create and invoke action methods in an application. This session will also discuss about routing requests and URL patterns.

### 2.2 In-Class Explanation

#### Slide 3

Let us understand the Controller.

The slide has a dark red header bar with the title "Working with Controllers 1-2" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains a bulleted list of six points describing what a controller does in an ASP.NET MVC application. At the bottom of the slide, there is a footer bar with the text "© Aptech Ltd.", "Controllers in ASP.NET MVC/Session 2", and the number "3". A large watermark reading "Aptech Centre Only" is diagonally across the slide.

- ◆ A controller, in an ASP.NET MVC application does the following:
  - ◆ Manages the flow of the application.
  - ◆ Is responsible for intercepting incoming requests and executing the appropriate application code.
  - ◆ Communicates with the models of the application and selects the required view to be rendered for the request.
  - ◆ Is a C# class that extends the `Controller` class of the `System.Web.Mvc` namespace.
  - ◆ Allows separating the business logic of the application from the presentation logic.

Use slide 3 to explain that a controller, in an ASP.NET MVC application manages the flow of the application. It is responsible for intercepting incoming requests and executing the appropriate application code. It is the controller that communicates with models of an application and selects the required view to be rendered for a request.

Tell them that a controller is a C# class that extends the `Controller` class of the `System.Web.Mvc` namespace. In addition, it allows separating the business logic of the application from the presentation logic.

**Slide 4**

Understand the responsibilities of a controller.

**Working with Controllers 2-2**

- ◆ In an ASP.NET MVC application, a controller is responsible to:
  - ◆ Locate the appropriate method to call for an incoming request.
  - ◆ Validate the data of the incoming request before invoking the requested method.
  - ◆ Retrieve the request data and passing it to requested method as arguments.
  - ◆ Handle any exceptions that the requested method throws.
  - ◆ Help in rendering the view based on the result of the requested method.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      4

In slide 4, explain the students about the responsibilities of a controller in an ASP.NET MVC application.

Explain them that a controller is responsible for:

- Locating the appropriate method to call for an incoming request.
- Validating the data of the incoming request before invoking the requested method.
- Retrieving the request data and passing it to requested method as arguments.
- Handling any exceptions that the requested method throws.
- Helping in rendering the view based on the result of the requested method.

**Best Practice:**

Can we include other methods in addition to action method in a controller? We can use any number of methods in a controller to implement business logic based on the application requirements. However, it is recommended to implement such method in separate helper classes instead in the controller class. Doing so

decreases maintainability of controller classes and reusability of the business logic across different controllers.

## Slide 5

Let us understand how to create a controller.

The slide has a dark red background with white text. At the top right, it says 'Creating a Controller 1-6'. Below that is a bulleted list of five items:

- ◆ In ASP.NET MVC, the `ControllerBase` class of the `System.Web.Mvc` namespace is the base class for all controllers.
- ◆ The `Controller` class extends the `ControllerBase` class to provide a default implementation of a controller.
- ◆ To create a controller in an ASP.NET MVC application, you will need to create a C# class that extends the `Controller` class.
- ◆ Instead of creating a controller manually, you can use Visual Studio 2013 IDE, which also creates the folder structure for the application automatically.

At the bottom left is the copyright notice '© Aptech Ltd.' and at the bottom center is 'Controllers in ASP.NET MVC/Session 2'. There is a small number '5' at the bottom right.

In slide 5, tell the students that in ASP.NET MVC, the `ControllerBase` class of the `System.Web.Mvc` namespace is the base class for all controllers.

Then, tell them that the `Controller` class extends the `ControllerBase` class to provide a default implementation of a controller. To create a controller in an ASP.NET MVC application, the students need to create a C# class that extends the `Controller` class.

Also, mention that instead of creating a controller manually, they can use Visual Studio 2013 IDE, which also creates the folder structure for the application automatically.

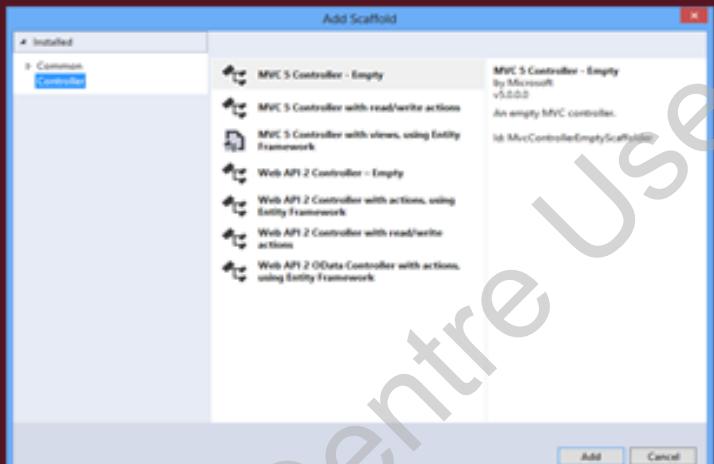
## Slide 6

Understand how to create a Controller.

### Creating a Controller 2-6

♦ In Visual Studio 2013 IDE, you can create a controller by performing the following steps:

1. Right-click the **Controllers** folder in the **Solution Explorer** window.
2. Select **Add→Controller** from the context menu that appears. The **Add Scaffold** dialog box is displayed, as shown in the following figure:



The screenshot shows the 'Add Scaffold' dialog box in Visual Studio. The 'Installed' tab is selected, displaying several controller templates under the 'Common' category. The 'Controllers' item is expanded. The 'MVC 5 Controller - Empty' template is highlighted. The dialog includes a detailed description of the template: 'An empty MVC controller.' and 'Id: MvcControllerEmptyScaffold'. At the bottom are 'Add' and 'Cancel' buttons. The background of the slide features a large watermark reading 'For Aptech Centre Use Only' diagonally across the content area.

© Aptech Ltd. Controllers in ASP.NET MVC/Session 2 5

Use slide 6 to explain the students about the steps to create a controller using Visual Studio 2013.

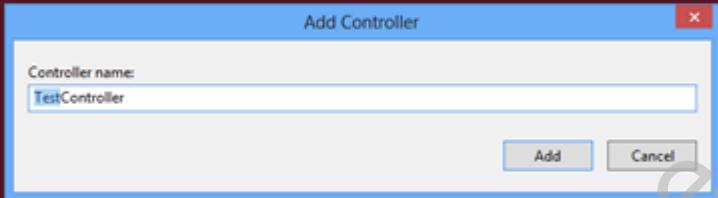
You can refer to the figure displayed on slide 6 that shows the Add Scaffold dialog box.

## Slide 7

Creating MVC 5 Empty Controller.

### Creating a Controller 3-6

3. Select the **MVC 5 Controller - Empty** in the **Add Scaffold** dialog box.  
4. Click **Add**. The **Add Controller** dialog box appears.  
5. Type **TestController** in the **Controller name** text field, as shown in the following figure:



6. Click **Add**. The **Solution Explorer** window displays the newly created **TestController** controller under the **Controllers** folder.

© Aptech Ltd. Controllers in ASP.NET MVC/Session 2 7

Use slide 7 to explain the steps required for creating a controller that are listed in the slide.

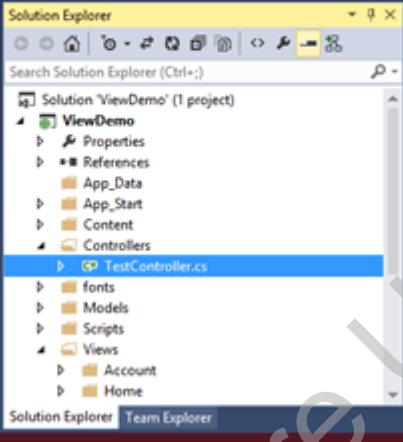
You can refer to the figure displayed on slide 7 that shows the Add Controller dialog box.

## Slide 8

Creating a Controller.

**Creating a Controller 4-6**

- ◆ Following figure shows the **Solution Explorer** window that displays the newly created controller under the Controllers folder:



The screenshot shows the 'Solution Explorer' window of a Microsoft Visual Studio IDE. The window title is 'Solution Explorer'. Inside, there is a tree view of a project named 'ViewDemo'. The 'Controllers' folder is expanded, and a file named 'TestController.cs' is selected, highlighted with a blue selection bar. Other visible folders include Properties, References, App\_Data, App\_Start, Content, Fonts, Models, Scripts, Views, Account, and Home. At the bottom of the window, tabs for 'Solution Explorer' and 'Team Explorer' are visible, along with copyright information for Aptech Ltd. and the slide title 'Controllers in ASP.NET MVC/Session 2'.

In slide 8, refer to the figure displayed on the slide that shows the newly created controller in the Solution Explorer window. The controller folder contains all the controllers created for the current application. If you have a controller named 'Test' then the file that will be automatically created will have name as 'TestController.cs'.

## Slide 9

Understanding syntax for creation of MVC Controller.

### Creating a Controller 5-6

- ◆ You can use the following syntax for creating a Controller class:

**Code Snippet:**

```
using System.Web.Mvc;
public class <Controller_Name>Controller : Controller
{
    //Some code
}
```

where,

- ◆ `<Controller_Name>`: is a name of the controller.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      9

Use slide 9 to explain them the syntax to create a Controller class as displayed on the slide. Tell them that whenever a controller is created, it is derived from the in-built class 'Controller' and it derives all the properties and methods or functions of the 'Controller' class.

## Slide 10

Understanding the skeleton code of ASP.NET MVC.

### Creating a Controller 6-6

- Following is the skeleton code of a Controller class:

**Code Snippet:**

```
using System.Web.Mvc
public class TestController : Controller
{
    //Some code
}
```

- In this code, a controller is created with the name `TestController`.

© Aptech Ltd. Controllers in ASP.NET MVC/Session 2 10

Use slide 10 to explain them the skeleton code of the `TestController` controller class, as displayed on the slide. In the code snippet as shown on the slide, a controller called '`TestController`' is created and it is based on or derived from '`Controller`' class.

## Slide 11

Let us understand how to work with Action Methods.

### Working with Action Methods 1-5

- ◆ A controller class can contains one or more action methods, also known as controller actions.
- ◆ Action methods:
  - ◆ Are responsible for processing the requests that are sent to the controller.
  - ◆ Typically returns an `ActionResult` object that encapsulates the result of executing the method.
- ◆ Following figure shows the working of action methods:

The diagram illustrates the flow of requests and responses in the MVC Framework. It shows a 'Web Browser' on the left and the 'MVC Framework' on the right. The browser sends an 'HTTP Request URL' (step 1) to the framework. The framework then performs an 'Invoke' (step 2) on the 'HomeController'. Inside the HomeController, an 'Index()' action method is executed (step 3). Finally, an 'ActionResult' is returned from the action method to the browser (step 4), which then sends an 'HTTP Response' back to the user.

© Aptech Ltd. Controllers in ASP.NET MVC/Session 2 11

In slide 11, tell the students that a controller class can contain one or more action methods, also known as controller actions.

Then, explain them that action methods are responsible for processing the requests that are sent to the controller. These methods typically return an `ActionResult` object that encapsulates the result of executing the method.

You can refer to the figure displayed on slide 11 that demonstrates the working of action methods.

## Slide 12

Let us understand how to work with Action Methods.

### Working with Action Methods 2-5

- ◆ The steps in the preceding figure are as follows:
  1. The browser sends an HTTP request.
  2. The MVC Framework invokes the controller action method based on the request URL.
  3. The action method executes and returns an `ActionResult` object. This object encapsulates the result of the action method execution.
  4. The MVC Framework converts an `ActionResult` to HTTP response and sends the response back to the browser.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      12

In slide 12, you refer to the figure displayed on slide 11 and explain the following steps demonstrated on the figure.

1. The browser sends an HTTP request.
2. The MVC Framework invokes the controller action method based on the request URL.
3. The action method executes and returns an `ActionResult` object. This object encapsulates the result of the action method execution.
4. The MVC Framework converts an `ActionResult` to HTTP response and sends the response back to the browser.

## Slide 13

Let us understand how to work with Action Methods.

### Working with Action Methods 3-5

- ◆ Rules that you need to consider while creating an action method are as follows:
  - ◆ They must be declared as public.
  - ◆ They cannot be declared as static.
  - ◆ They cannot have overloaded versions based on parameters.
- ◆ Following is the syntax for creating an action method in a Controller class:

**Code Snippet:**

```
public ActionResult <ActionMethod_Name>()
{
    /*Code to execute logic and return the result as ActionResult*/
}
```

where,

- ◆ <ActionMethod\_Name>: is a name of the action method.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      13

Use slide 13 to explain the rules that the students need to consider while creating an action method.

Tell them that while creating an action method they should remember that an action method must be declared as public, they cannot be declared as static, and they cannot have overloaded versions based on parameters.

Refer to the code snippet displayed on slide 13 to explain the syntax for creating an action method in a Controller class.

## Slide 14

Creating two action methods Index and About.

### Working with Action Methods 4-5

- ◆ Following code creates two action methods with the name Index and About in the HomeController controller class:

**Code Snippet:**

```
using System.Web.Mvc;
public class HomeController : Controller
{
    public ActionResult Index()
    {
        /*Code to execute logic and return the result as ActionResult*/
    }
    public ActionResult About()
    {
        /*Code to execute logic and return the result as ActionResult*/
    }
}
```

- ◆ The code creates two action methods, named Index and About in the HomeController controller class. Both these action methods are declared as public and to return ActionResult objects.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      14

Use slide 14 to explain the code snippet that creates two action methods with the name Index and About in the HomeController controller class.

Explain them that the given code on the slide creates two action methods, named Index and About in the HomeController controller class. Both these action methods are declared as public and to return ActionResult object.

## Slide 15

Understand how to return other types of data using Action Methods.

### Working with Action Methods 5-5

- ♦ Although, most of the action methods return an `ActionResult` object, an action method can also return other types, such as `String`, `int`, or `bool`, as shown in the following code:

**Code Snippet:**

```
using System.Web.Mvc;
public class HomeController : Controller
{
    public ActionResult Index()
    {
        /*Code to execute logic and return the result as ActionResult*/
    }
    public ActionResult About()
    {
        /*Code to execute logic and return the result as ActionResult*/
    }
}
```

- ♦ This code creates two action methods, named `IsValid` that returns a `bool` value and `Contact` that returns a `String` value.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      15

In slide 15, refer to the code displayed and then explain the students that although, most of the action methods return an `ActionResult` object, an action method can also return other types, such as `String`, `int`, or `bool`.

Tell them that the code displayed on the slide creates two action methods, named `IsValid` that returns a `bool` value and `Contact` that returns a `String` value.

## Slide 16

Let us understand the ActionResult object.

### Action Results 1-2

- ◆ **ActionResult:**
  - ◆ Is an abstract base class for all implementing classes that provides different types of results.
  - ◆ Consists of HTML in combination with server-side and client-side scripts to respond to user actions.
- ◆ Following table shows the commonly used classes that extend the ActionResult class to provide different implementations of the results of an action method:

Classes	Description
ViewResult	Renders a view as an HTML document.
PartialViewResult	Renders a partial view, which is a sub-view of the main view.
EmptyResult	Returns an empty response.
RedirectResult	Redirects a response to another action method.
JsonResult	Returns the result as JSON, also known as JavaScript Object Notation (JSON). JSON is an open standard format to store and exchange text information.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      16

In slide 16, explain the students that ActionResult is an abstract base class for all implementing classes that provides different types of results. This object consists of HTML in combination with server-side and client-side scripts to respond to user actions.

Refer to the table displayed on slide 16 to explain the commonly used classes that extend the ActionResult class to provide different implementations of the results of an action method.

## Slide 17

Let us understand the classes that extend the ActionResult object.

Action Results 2-2	
Classes	Description
JavaScriptResult	Returns JavaScript that executes on the client browser.
ContentResult	Returns the content based on a defined content type, such as XML.
FileContentResult	Returns the content of a binary file.
FileStreamResult	Returns the content of a file using a Stream object.
FilePathResult	Returns a file as a response.

Slide 17 is continuation of slide 16. It shows the table with classes and description of the same. The entire table shown here gives you an idea about which classes are available and what is the function of each class.

## Slide 18

Let us understand how to invoke action methods.

### Invoking Action Methods 1-3

- ◆ In an ASP.NET MVC application, you can create multiple action methods in a controller.
- ◆ You can invoke an action method by specifying a URL in the Web browser containing the name of the controller and the action method to invoke.
- ◆ Following code shows the general syntax to invoke an action method:

Code Snippet:

```
http:// <domain_name> /<controller_name>/<actionmethod_name>
```

where,

- ◆ <domain\_name>: Is the domain name of the application.
- ◆ <controller\_name>: Is the name of the controller without the Controller suffix.
- ◆ <actionmethod\_name>: Is the name of the action method to invoke.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      18

In slide 18, tell the students that in an ASP.NET MVC application, they can create multiple action methods in a controller. They can invoke an action method by specifying a URL in the Web browser containing the name of the controller and the action method to invoke.

Refer to the code snippet displayed on slide 18 and explain them that in the code:

- <domain\_name>: represents the domain name of the application.
- <controller\_name>: represents the name of the controller without the Controller suffix.
- <actionmethod\_name>: represents the name of the action method to invoke.

## Slide 19

Let us understand how to invoke action methods.

### Invoking Action Methods 2-3

- ◆ Consider the following URL:  
`http:// mvceexample.com/Home/Registration`
- ◆ When this URL is sent to the application through a Web browser, the MVC Framework performs the following tasks:
  - ◆ Searches for the `HomeController` controller class.
  - ◆ Searches for the `Registration()` action method in the `HomeController` controller class.
  - ◆ Executes the `Registration()` action method.
  - ◆ Returns the response back to the browser.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      19

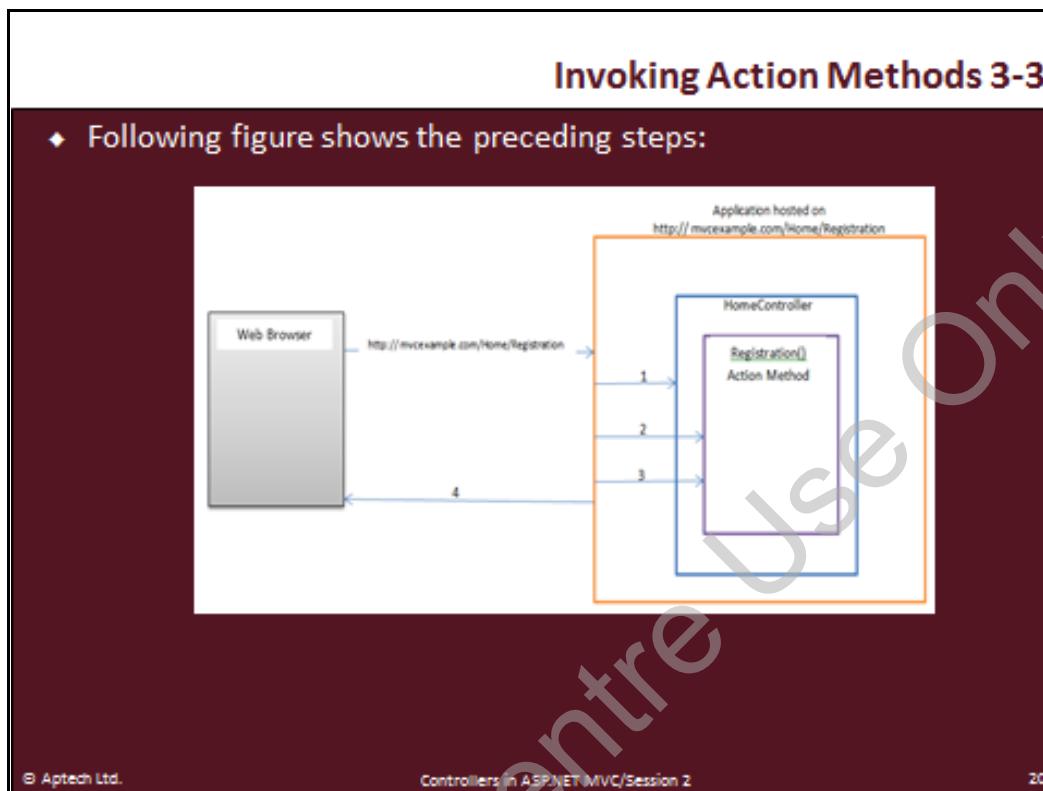
In slide 19, tell the students to consider the URL given in the slide.

Tell the students that when this URL is sent to the application through a Web browser, the MVC Framework performs the following tasks:

- Searches for the `HomeController` controller class.
- Searches for the `Registration()` action method in the `HomeController` controller class.
- Executes the `Registration()` action method.
- Returns the response back to the browser.

**Slide 20**

Understanding how Action Methods are invoked.



Using slide 20, explain to students that a http request is sent to the server by the browser. When browser receives such a request, it is sent to the controller for further processing. Home controller decides which action is to be invoked then the related action method 'Registration()' is called and invoked.

**Slide 21**

Let us understand how to pass parameters.

### Passing Parameters 1-2

- ◆ Sometimes you may need to provide input other than the Web page name while requesting for a Web page.
- ◆ Consider the following URL:
  - ◆ `http://www.mvceexample.com/student/details?Id=006`
  - ◆ The preceding URL will invoke the Details action method of the StudentController controller class.
  - ◆ The URL also contains an `Id` parameter with the value 006.
- ◆ The Details action method must accept an `Id` parameter of type `string` in order to return student records based on the `Id` value.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      21

In slide 21, explain the students that sometimes they may need to provide input other than the Web page name while requesting for a Web page.

Tell them to consider the given URL in the slide.

Then, explain them that this URL will invoke a Details action method of the StudentController controller class. This URL also contains an `Id` parameter with the value 006. Tell them that the Details action method must accept an `Id` parameter of type `string` in order to return student records based on the `Id` value.

## Slide 22

Let us understand how to pass parameters.

### Passing Parameters 2-2

- Following code shows the Details action that accepts an Id parameter:

**Code Snippet:**

```
public ActionResult Details(string Id)
{
    /*Return student records based on the Id parameter as an
    ActionResult object*/
}
```

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      22

In slide 22, refer to the code given on the slide and explain how the Details action accepts an Id parameter. Depending on the Id passed the decision is taken by the controller and related controller is initiated. It processes the desired controller and the result is given back as ActionResult.

## Slide 23

Let us understand the request routing.

### Routing Requests

- ◆ MVC Framework introduces routing that allows you to define URL patterns with placeholders that maps to request URLs pattern.
- ◆ In an ASP.NET MVC application, routing:
  - ◆ Defines how the application will process and respond to incoming HTTP request.
  - ◆ Properly describes the controller action to which the requested needs to be routed.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      23

Use slide 23, to tell the students that MVC Framework introduces routing that allows them to define URL patterns with placeholders that maps to request URLs pattern.

Explain them that in an ASP.NET MVC application, routing defines how the application will process and respond to incoming HTTP request. In addition, routing properly describes the controller action to which the requested needs to be routed.

## Slide 24

Let us understand the use of routing.

### Uses of Routing

- ◆ Routing is a process that maps incoming requests to specified controller actions.
- ◆ Two main functions of routing are as follows:
  - ◆ Mapping incoming requests to controller action.
  - ◆ Constructing outgoing URLs corresponding to controller actions.
- ◆ Routing is achieved by configuring route patterns in the application, that includes:
  - ◆ Creating the route patterns
  - ◆ Registering the patterns with the route table of the MVC Framework
- ◆ At the time of creating an ASP.NET MVC application the application can register multiple routing patterns with the MVC Framework's route table.
- ◆ Route tables provide the information on how the routing engine processes requests that matches those patterns.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      24

In slide 24, tell the students that routing is a process that maps incoming requests to specified controller actions.

Tell them that the two main functions of routing include mapping incoming requests to controller action and constructing outgoing URLs corresponding to controller actions.

Next, tell them that routing is achieved by configuring route patterns in the application, that includes creating the route patterns and registering the patterns with the route table of the MVC Framework. Tell them that at the time of creating an ASP.NET MVC application, the application can register multiple routing patterns with the MVC Framework's route table.

In addition, tell them that route tables provide the information on how the routing engine processes requests that matches those patterns.

## Slide 25

Let us understand the default route.

The slide has a dark red header bar with the title "The Default Route 1-2". Below the header, there is a list of bullet points:

- ◆ An MVC application requires a route to handle user requests.
- ◆ When you create an ASP.NET MVC application in Visual Studio 2013, a route is automatically configured in the `RouteConfig.cs` file.
- ◆ Following code shows the `MapRoute()` method:

A code snippet box contains the following C# code:

```
routes.MapRoute(
    name: "Default",
    url: "{controller}/{action}/{id}",
    defaults: new { controller = "Home", action = "Index", id =
UrlParameter.Optional }
);
```

At the bottom of the slide, there are footer details: "© Aptech Ltd.", "Controllers in ASP.NET MVC/Session 2", and "25".

In slide 25, tell the students that an MVC application requires a route to handle user requests.

Explain them that when they create an ASP.NET MVC application in Visual Studio 2013, a route is automatically configured in the `RouteConfig.cs` file.

Refer to the code displayed on the slide to explain the `MapRoute()` method.

## Slide 26

Understanding Default Route.

### The Default Route 2-2

- ◆ In the code:
  - ◆ The routes is of type `System.Web.Routing.RouteCollection` represents a collection of routes for the application.
  - ◆ The `MapRoute()` method defines a route named `Default`, a URL pattern, and a default route.
  - ◆ The default route is used if the request URL does not match with the defined URL pattern defined in the `MapRoute()` method. For example, if a request URL does not contain the name of a controller and an action, the request will be routed to the `Index` action of the `Home` controller.

© Aptech Ltd. Controllers in ASP.NET MVC/Session 2 26

In slide 26, refer to the code displayed on slide 25 and explain them that in the code:

- The routes are of type `System.Web.Routing.RouteCollection` represents a collection of routes for the application.
- The `MapRoute()` method defines a route named `Default`, a URL pattern, and a default route.
- The default route is used if the request URL does not match with the defined URL pattern defined in the `MapRoute()` method. For example, if a request URL does not contain the name of a controller and an action, the request will be routed to the `Index` action of the `Home` controller.

## Slide 27

Let us understand about registering default route.

**Registering Default Route 1-3**

- ◆ In an ASP.NET MVC application, the Global.asax file:
  - ◆ Initializes the application with the features of the MVC Framework when the application starts.
  - ◆ Contains a `MVCApplication` class with the `Application_Start()` method where you need to register the default route so that the Framework uses the route when a request starts coming to the application.

© Aptech Ltd. Controllers in ASP.NET MVC/Session 2 27

In slide 27, explain the students about the Global.asax file.

Tell them that in an ASP.NET MVC application, the Global.asax file initializes the application with the features of the MVC Framework when application starts.

In addition, tell them that this file contains a `MVCApplication` class with the `Application_Start()` method where they need to register the default route so that the Framework uses the route when a request starts coming to the application.

## Slide 28

Let us understand about registering default route.

### Registering Default Route 2-3

+ Following code shows the MVCApplication class of the Global.asax file:

**Code Snippet:**

```
using System.Collections.Generic;
using System.Linq;
using System.Web;
using System.Web.Http;
using System.Web.Mvc;
using System.Web.Optimization;
using System.Web.Routing;

namespaceUrlsAndRoutes {
    public class MvcApplication : System.Web.HttpApplication {
        protected void Application_Start() {
            RouteConfig.RegisterRoutes(RouteTable.Routes);

            /*Code for registering other MVC components*/
        }
    }
}
```

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      28

In slide 28, explain the given code that shows the MVCApplication class of the Global.asax file. Tell the students that MvcApplication is derived from System.Web.HttpApplication which has in-built method called Application\_Start().

## Slide 29

Let us understand about registering default route.

### Registering Default Route 3-3

- ◆ In this code:
  - ◆ The `MVCApplication` class extends the `System.Web.HttpApplication` class that defines the common features required by the application objects in an ASP.NET application.
  - ◆ The `Application_Start()` method calls the `RouteConfig.RegisterRoutes()` method to register the default route to be used in the application.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      29

In slide 29, refer to the code displayed on slide 28 and explain the students that in the code the `MVCApplication` class extends the `System.Web.HttpApplication` class that defines the common features required by the application objects in an ASP.NET application.

Then, tell them that the `Application_Start()` method calls the `RouteConfig.RegisterRoutes()` method to register the default route to be used in the application.

## Slide 30

Let us understand URL pattern.

The slide has a dark red header bar with the title 'URL Patterns 1-2' in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following text:

- ◆ URL pattern:
  - ◆ Is required to be defined when you create a route.
  - ◆ Is compared with the URL of a request by the route engine of the MVC Framework.
  - ◆ Contains literal values and placeholders separated by the slash (/) character. Following is an example of the URL Pattern:  
    '{controller}/{action}/{id}'
- ◆ URL that will match the preceding pattern:  
    `http://www.mvceexample.com/student/records/36`
- ◆ When the routing engine matches the preceding URL with the URL pattern it performs the following actions:
  - ◆ Assign student as the value of the {controller} placeholder
  - ◆ Assigns records as the value of the {action} placeholder
  - ◆ Assign 36 as the value of the {id} placeholder

At the bottom of the slide, there is a footer bar with the following text:  
© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      30

Use slide 30 to explain the students that URL pattern is required to be defined when they create a route. It is compared with the URL of a request by the route engine of the MVC Framework.

In addition, tell them that URL pattern contains literal values and placeholders separated by the slash (/) character.

Refer to the URL given on the slide as an example.

Explain them that when the routing engine matches the preceding URL with the URL pattern it performs the following actions:

- Assign student as the value of the {controller} placeholder
- Assigns records as the value of the {action} placeholder
- Assign 36 as the value of the {id} placeholder

**Slide 31**

Understanding the URL Literals and placeholders.

**URL Patterns 2-2**

- ◆ A URL parameter can also have a combination of literal values and placeholders.  
`"student/{action}/{id}"`
- ◆ Some of the URLs that will match with the preceding URL pattern are:
  - ◆ `http://www.mvceexample.com/student/records/36`
  - ◆ `http://www.mvceexample.com/student/delete/21`
  - ◆ `http://www.mvceexample.com/student/view/23`

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      31

Show slide 31 and tell the students that a URL parameter can also have a combination of literal values and placeholders, for example, "student/{action}/{id} ". The first call to this URL sends Id as 36, second sends 21, and the third sends 23.

## Slide 32

Let us understand how to ordering the routes.

### Ordering Routes 1-2

- ◆ Sometimes you may need to register multiple routes in an ASP.NET MVC application.
- ◆ For that you can configure the sequence in which the routes will execute.
- ◆ A route engine starts matching a request URL with a URL pattern starting from the first registered route.
- ◆ When a matching route is encountered the route engine stops the matching process.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      32

In slide 32, tell the students that sometimes they may need to register multiple routes in an ASP.NET MVC application. For that they can configure the sequence in which the routes will execute.

Next, tell them that a route engine starts matching a request URL with a URL pattern starting from the first registered route. When a matching route is encountered the route engine stops the matching process.

### Slide 33

Let us understand how to ordering the routes.

### Ordering Routes 2-2

- ◆ Following code snippet demonstrates two routes:

**Code Snippet:**

```
routes.MapRoute(
    name: "general",
    url: "{controller}/{action}",
    defaults: new { controller = "Home", action = "Index" });
    routes.MapRoute(
        name: "manager",
        url: "Manager/{action}",
        defaults: new { controller = "Manager", action = "Browse" }
    );
```

- ◆ This code:
  - ◆ Contains two placeholders and sets the default value of the controller parameter to Home and the action parameter to Index.
  - ◆ Second route contains a literal Manager, and a placeholder, and sets the default value of the controller parameter to Manager and the action parameter to Browse.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      33

Use slide 33 and refer to the code displayed on the slide.

Then, explain them that the code contains two placeholders and sets the default value of the controller parameter to Home and the action parameter to Index.

Then, tell them that the second route contains a literal Manager, and a placeholder, and sets the default value of the controller parameter to Manager and the action parameter to Browse.

## Slide 34

Let us understand the constraining routes.

### Constraining Route 1-4

- ◆ In an ASP.NET MVC application, the routing engine enables you to apply constraints around the placeholder values.
- ◆ You can use constraints when an application have identical route URLs but based on the application requirement the route engine should resolve the URLs to different controllers or actions.
- ◆ Following code shows a route with a route constraint:

Code Snippet:

```
routes.MapRoute(
    "Product",
    "{controller}/{action}/{id}",
    new { controller = "Product", action = "Browse", id =
        UrlParameter.Optional },
    new { id = "(Jewellery|Jeans|Mobile)" }
);
```

- ◆ In this code, a constraint is applied to the route, so that id placeholder can have only one of the Jewellery, Jeans, and Mobile values.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      34

In slide 34, explain to the students that in an ASP.NET MVC application, the routing engine enables them to apply constraints around the placeholder values.

Tell them that they can use constraints when an application have identical route URLs but based on the application requirement the route engine should resolve the URLs to different controllers or actions. Refer to the code displayed on the slide and tell them that in this code, a constraint is applied to the route, so that id placeholder can have only one of the Jewellery, Jeans, and Mobile values.

**Slide 35**

Let us understand the constraining routes.

Constraining Route 2-4	
URL	Matching Results
http://www.mvceample.com	Yes. The default values of the controller and action are specified as Product and Browse respectively.
http://www.mvceample.com/Product	Yes. The default value of the action is specified as Browse.
http://www.mvceample.com/Product/Browse	Yes. The id is specified as an optional parameter.
http://www.mvceample.com/Product/Browse/Jewellery	Yes. Jewellery specified in the URL is present in the list containing valid values for id parameter.
http://www.mvceample.com/Product/Browse/Jeans	Yes. Jeans specified in the URL is present in the list containing valid values for id parameter.

In slide 35, refer to the table displayed on the slide that describes whether the routing engine will match different URLs based on the routing constraints. With the help of the given table you can learn how the values called Id are passed along with Request Object.

**Slide 36**

Let us understand the constraining routes.

Constraining Route 3-4	
URL	Matching Results
<code>http://www.mvceexample.com/Product/Browse/Mobile</code>	Yes. Mobile specified in the URL is present in the list containing valid values for id parameter.
<code>http://www.mvceexample.com/Product/Browse/Laptop</code>	No. Laptop specified in the URL is not present in the list containing valid values for id parameter.
<code>http://www.mvceexample.com/Product/Browse/Glasses</code>	No. Glasses specified in the URL is not present in the list containing valid values for id parameter.

Slide 36 continues the table as shown on slide 35. Here, you can note that in the last two rows of the table the URL does not return anything as the Id does not match with any of the existing controller with the result returned by these URLs will be an error.

## Slide 37

Let us understand the constraining routes.

### Constraining Route 4-4

- Following code shows a route with a route constraint which specifies that the id placeholder can match only with an integer value:

**Code Snippet:**

```
routes.MapRoute(
    name: "Product",
    url: "{controller}/{action}/{id}",
    defaults: new { controller = "Product", action = "Browse",
        id=UrlParameter.
    Optional},
    constraints: new { id = @"\d+" }
);
```

- In this code, a route with three placeholders, controller, action, and id. In this route, the default value for the controller placeholder is set to Product, the action placeholder is set to Browse.
- In addition, a constraint is applied to an id optional parameter. This constraint uses a regular expression to specify that the id parameter can match only with an integer value.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      37

In slide 37, refer to the code displayed on the slide.

Tell them that this code shows a route with a route constraint which specifies that the id placeholder can match only with an integer value.

Tell them that the code shows a route with three placeholders, controller, action, and id. In the route, the default value for the controller placeholder is set to Product, the action placeholder is set to Browse. In addition, tell them that a constraint is applied to an id optional parameter. This constraint uses a regular expression to specify that the id parameter can match only with an integer value.

## Slide 38

Let us understand how to ignore a route.

### Ignoring a Route

- ◆ The MVC Framework provides you the flexibility to ignore routes.
- ◆ You can use the `IgnoreRoute()` method of the `RoutesTable` class to specify a route that the MVC routing engine should ignore.
- ◆ Following code shows how to ignore a route:

**Code Snippet:**

```
routes.IgnoreRoute("{resource}.axd/{*pathInfo}");
```

◆ In this code, the `routes.IgnoreRoute()` method specifies that resources with the `.axd` extension should be ignored by the route engine and served directly as response.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      38

In slide 38, explain to the students that the MVC Framework provides the flexibility to ignore routes.

Tell them that they can use the `IgnoreRoute()` method of the `RoutesTable` class to specify a route that the MVC routing engine should ignore.

Next, refer to the code displayed on the slide and explain them that in this code, the `routes.IgnoreRoute()` method specifies that resources with the `.axd` extension should be ignored by the route engine and served directly as response.

## Slide 39

Let us summarize the session.

**Summary**

- ◆ A controller is responsible for intercepting incoming requests and executing the appropriate application code.
- ◆ To create a controller in an ASP.NET MVC application, you will need to create a C# class that extends the Controller class.
- ◆ A controller class can contain one or more action methods, also known as controller actions.
- ◆ Although, most action methods return an ActionResult object, an action method can also return other types, such as String, int, or bool.
- ◆ Routing is a process that maps incoming requests to specified controller actions.
- ◆ When you create a route, you need to define a URL pattern that can contain literal values and placeholders separated by the slash (/) character for the route.
- ◆ The routing engine allows you to apply constraints around the placeholder values and also ignore routes.

© Aptech Ltd.      Controllers in ASP.NET MVC/Session 2      39

In slide 39, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- A controller is responsible for intercepting incoming requests and executing the appropriate application code.
- To create a controller in an ASP.NET MVC application, you will need to create a C# class that extends the Controller class.
- A controller class can contain one or more action methods, also known as controller actions.
- Although, most action methods return an ActionResult object, an action method can also return other types, such as String, int, or bool.
- Routing is a process that maps incoming requests to specified controller actions.
- When you create a route, you need to define a URL pattern that can contain literal values and placeholders separated by the slash (/) character for the route.
- The routing engine allows applying constraints around the placeholder values and also ignoring routes.

## **2.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. What are the rules you need to consider while creating an action method?
2. Which method will you use to configure a route?
3. How will you ignore a route?

## Session 3

### Views in ASP.NET MVC

#### **3.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth. This session deals with views, Razor engine, and HTML helper methods. The trainer must go through the material properly and learn all the topics given here.

##### **3.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe views
- Explain and describe the Razor engine
- Define and describe the HTML helper methods

##### **3.1.2 Teaching Skills**

To teach this session successfully, you must know about views and Razor engine in an ASP.NET MVC application. You should also be aware of the HTML helper methods.

You should teach the concepts in the theory class using slides and LCD projectors.

##### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the given order during In-Class activities.

#### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the views and Razor engine in an ASP.NET MVC application. They will learn about how to create and use views and Razor engine in an application. This session will also discuss about HTML helper methods.

### 3.2 In-Class Explanation

#### Slide 3

Let us understand views.

The slide has a dark red background with white text. At the top right, it says 'Working with Views'. Below that is a bulleted list:

- ◆ To display HTML content to the user, you can instruct the controller action in the application to return a view.
- ◆ A view:
  - ◆ Provides the UI of the application to the user.
  - ◆ Is used to display content of an application and also to accept user inputs.
  - ◆ Uses model data to create this UI.
  - ◆ Contains both HTML markup and code that runs on the Web server.

At the bottom left is the Aptech logo, and at the bottom right are the words 'Views in ASP.NET MVC/Session 3' and the number '3'.

Use slide 3 to explain the students that they can instruct the controller action in the application to return a view to display HTML content to the user.

Tell them that a view provides the UI of the application to the user. It is used to display content of an application and also to accept user inputs.

In addition, tell them that a view uses model data to create this UI and contains both HTML markup and code that runs on the Web server.

## Slide 4

Let us understand view engines.

The slide has a dark red header bar with the title 'View Engines' in white. Below the header is a white content area containing a bulleted list. At the bottom of the slide is a dark red footer bar with the Aptech logo, the slide title, and the number '4'.

**View Engines**

- ◆ Are part of the MVC Framework that converts the code of a view into HTML markup that a browser can understand.
- ◆ Are divided in the following two categories:
  - ◆ **Web Form view engine:** Is a legacy view engine for views that use the same syntax as ASP.NET Web Forms.
  - ◆ **Razor view engine:** Is the default view engine starting from MVC3. This view engine does not introduce a new programming language, but instead introduces new markup syntax to make transitions between HTML markups and programming code simpler.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 4

In slide 4, explain the students that view engines are part of the MVC Framework that converts the code of a view into HTML mark-up that a browser can understand.

Tell them that view engines are divided in two categories.

The first one is Web Form view engine. It is a legacy view engine for views that use the same syntax as ASP.NET Web Forms.

Then, the second one is the Razor view engine. It is the default view engine starting from MVC 3. This view engine does not introduce a new programming language, but instead introduces new mark-up syntax to make transitions between HTML mark-ups and programming code simpler.

## Slide 5

Let us understand how to specify a view for an action.

**Specifying the View for an Action 1-8**

- ◆ While creating an ASP.NET MVC application, you often need to specify a view that should render the output for a specific action.
- ◆ When you create a new project in Visual Studio .NET, the project by default contains a Views directory.
- ◆ In an application, if a controller action returns a view, your application should contain the following:
  - ◆ A folder for the controller, with the same name as the controller without the Controller suffix.
  - ◆ A view file in the Home folder with the same name as the action.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 5

In slide 5, tell the students that while creating an ASP.NET MVC application, they often need to specify a view that should render the output for a specific action. When they create a new project in Visual Studio .NET, the project by default contains a Views directory.

Then, tell them that in an application, if a controller action returns a view, the application should contain a folder for the controller, with the same name as the controller without the Controller suffix.

Also mention that the application should contain a view file in the Home folder with the same name as the action.

### Information:

By convention, the Views folder contains one folder for every controller, with the same name as the controller.

## Slide 6

Let us understand the `Index` action.

### Specifying the View for an Action 2-8

- Following code shows an `Index` action that returns an `ActionResult` object through a call to the `View()` method of the Controller class:

**Code Snippet:**

```
public class HomeController : Controller {
    public ActionResult Index()
    {
        return View();
    }
}
```

- In this code, the `Index` action of the controller named `HomeController` that returns the result of a call to the `View()` method. The result of the `View()` method is an `ActionResult` object that renders a view.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 6

In slide 6, explain the code that shows an `Index` action that returns an `ActionResult` object through a call to the `View()` method of the `Controller` class.

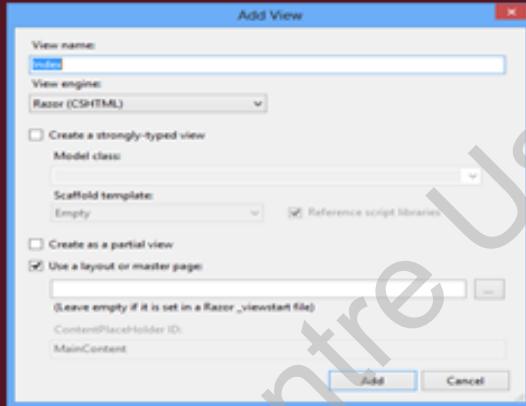
Then, explain them that in the code displayed on the slide, the `Index` action of the controller named `HomeController` that returns the result of a call to the `View()` method. The result of the `View()` method is an `ActionResult` object that renders a view.

## Slide 7

Let us understand how to create a View.

### Specifying the View for an Action 3-8

- ◆ Visual Studio 2013 simplifies the process of creating a view.
- ◆ You can create the view file by performing the following steps:
  1. Right-click inside the action method for which you need to create a view.
  2. Select Add View from the context menu that appears. The Add View dialog box is displayed, as shown in the following figure:



© Aptech Ltd. Views in ASP.NET MVC/Session 3 7

Use slide 7 to explain the students that Visual Studio 2013 simplifies the process of creating a view.

Use this slide to explain the steps to create a view, which are listed on the slide.

Then, refer to the figure displayed on the slide that shows the **Add View** dialog box.

**Tip:**

Another way of creating a View for any controller is, first, right-click a controller for which a view is to be created, then click **Create View**, it will start creating view and the name of the view will be same as the name of the controller.

## Slide 8

Let us understand the directory structure.

**Specifying the View for an Action 4-8**

3. Click Add. Visual Studio.NET automatically creates an appropriate directory structure and adds the view file to it.  
Following figure shows the view file that Visual Studio.NET creates for the Index action method of the HomeController class in the Solution Explorer window:

The screenshot shows the Visual Studio Solution Explorer window. The tree view displays the following directory structure under the 'Views' folder:

- Views
- Home
- Index.cshtml

The 'Index.cshtml' file is highlighted with a blue selection bar at the bottom of the list. The status bar at the bottom of the window shows 'Views in ASP.NET MVC/Session 3'. The bottom right corner of the slide has a small number '8'.

In slide 8, explain the figure that shows the Views folder where Visual Studio.NET creates the Index action method of the HomeController class, in the Solution Explorer window. Tell the students that the file structure is automatically created by Visual Studio and need not mention where to store the view that are created just now.

## Slide 9

Let us explain the contents of `Index.cshtml` file.

### Specifying the View for an Action 5-8

- ♦ In the `Index.cshtml` file, you can add the following code that the view should display:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Test View</title>
  </head>
  <body>
    <h1> Welcome to the Website </h1>
  </body>
</html>
```
- ♦ This code creates a view with a title and a message as a heading.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 9

In slide 9, explain the code and tell the students that in the `Index.cshtml` file, they can add the code that the view should display.

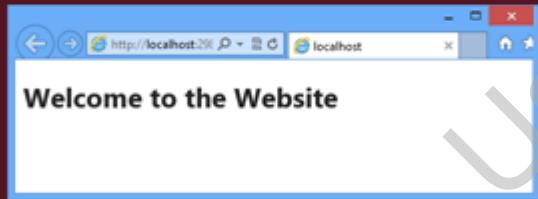
Tell them that this code creates a view with a title and a message as a heading.

## Slide 10

Let us execute and see the output of Index action.

### Specifying the View for an Action 6-8

- ◆ When you access the Index action of the HomeController from a browser, the Index.cshtml view will be displayed.
- ◆ You can use the following URL to access the Index action of the HomeController:  
`http://localhost:1267/Home/Index`
- ◆ Following figure shows the Index.cshtml view rendered in the browser:



The screenshot shows a Microsoft Edge browser window. The title bar says "http://localhost:2915" and "localhost". The main content area displays the text "Welcome to the Website". At the bottom of the slide, there is footer text: "© Aptech Ltd.", "Views in ASP.NET MVC/Session 3", and "10".

Use slide 10 to explain the students that when they access the Index action of the HomeController from a browser, the Index.cshtml view will be displayed.

Then, tell them that they can use the following URL to access the Index action of the HomeController: `http://localhost:1267/Home/Index`

Refer the figure displayed on the slide that shows the Index.cshtml view rendered in the browser.

## Slide 11

Let us learn and understand how to call another View .

### Specifying the View for an Action 7-8

- ◆ You can also render a different view from an action method.
- ◆ To return a different view, you need to pass the name of the view as a parameter.
- ◆ Following code shows how to return a different view:  
**Code Snippet:**

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        return View("TestIndex");
    }
}
```
- ◆ This code will search for a view inside the **/Views/Home** folder, but render the **TestIndex** view instead of rendering the **Index** view.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 11

In slide 11, tell the students that they can also render a different view from an action method. To return a different view, they need to pass the name of the view as a parameter.

Then, refer to the code displayed on the slide that shows how to return a different view.

Explain them that the referred code searches for a view inside the **/Views/Home** folder, but render the **TestIndex** view instead of rendering the **Index** view.

## Slide 12

Let us understand how to call a View from different folder.

### Specifying the View for an Action 8-8

- ♦ While developing an ASP.NET MVC application, you might also need to render a view that is present in a different folder instead of the default folder.
- ♦ To render such view, you need to specify the path to the view.
- ♦ Following code shows displaying a view named `Index.cshtml` present in the `/Views/Demo` folder:

**Code Snippet:**

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        return View("~/Views/Demo/Welcome.cshtml");
    }
}
```

♦ This code will display the view, named `Welcome.cshtml` defined inside the `/Views/Demo` folder.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 12

In slide 12, explain the students that while developing an ASP.NET MVC application, they might also need to render a view that is present in a different folder instead of the default folder. To render such view, they need to specify the path to the view.

Then, refer to the code displayed on the slide that shows how to display a view named `Index.cshtml` present in the `/Views/Demo` folder.

Then, explain them that this code will display the view, named `Welcome.cshtml` defined inside the `/Views/Demo` folder.

#### Discussion:

Initiate a discussion by explaining that the tilde (~) sign is used to refer to the root application folder. While using the tilde symbol, student must give the file extension of the view because this method bypasses MVC's default convention of finding a view.

## Slide 13

Let us understand passing data from a controller to a view.

**Passing Data from a Controller to a View 1-14**

- ◆ In an ASP.NET MVC application, a controller typically performs the business logic of the application and needs to return the result to the user through a view.
- ◆ You can use the following objects to pass data between controller and view:
  - + ViewData
  - + ViewBag
  - + TempData

© Aptech Ltd.      Views in ASP.NET MVC/Session 3      13

Use slide 13 to explain that in an ASP.NET MVC application, a controller typically performs the business logic of the application and needs to return the result to the user through a view.

Tell them that they can use objects, such as ViewData, ViewBag, and TempData to pass data between controller and view.

## Slide 14

Let us understand ViewData.

### Passing Data from a Controller to a View 2-14

- ◆ **ViewData**
  - ◆ Passes data from a controller to a view.
  - ◆ Is a dictionary of objects that is derived from the `ViewDataDictionary` class.
  - ◆ Some of the characteristics of `ViewData` are as follows:
    - ◆ The life of a `ViewData` object exists only during the current request.
    - ◆ The value of `ViewData` becomes null if the request is redirected.
    - ◆ `ViewData` requires typecasting when you use complex data type to avoid error.
  - ◆ The general syntax of `ViewData` is as follows:

**Syntax:**

```
ViewData[<key>] = <Value>;
```

where,

- ◆ Key: Is a `String` value to identify the object present in `ViewData`.
- ◆ Value: Is the object present in `ViewData`. This object may be a `String` or a different type, such as `DateTime`.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 14

Use slide 14 to tell the student that `ViewData` allows passing data from a controller to a view. It is a dictionary of objects that is derived from the `ViewDataDictionary` class.

Then, explain them the following characteristics of `ViewData` object:

- The life of a `ViewData` object exists only during the current request.
- The value of `ViewData` becomes null if the request is redirected.
- `ViewData` requires typecasting when you use complex data type to avoid error.

Next, refer to the syntax of `ViewData` object displayed on the slide and explain it.

## Slide 15

Let us understand how to add data into ViewData.

### Passing Data from a Controller to a View 3-14

- Following code shows a ViewData with two key-value pairs in the Index action method of the HomeController class:

**Code Snippet:**

```
public class HomeController : Controller {
    public ActionResult Index()
    {
        ViewData["Message"] = "Message from ViewData";
        ViewData["CurrentTime"] = DateTime.Now;
        return View();
    }
}
```

- In this code a ViewData is created with two key-value pairs.
- The first key named Message contains the String value, Message from ViewData.
- The second key named CurrentTime contains the value, DateTime.Now.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 15

In slide 15, refer to the code displayed on the slide that shows a ViewData object with two key-value pairs in the Index action method of the HomeController class.

Then, explain the students that in this code a ViewData object is created with two key-value pairs. Where, the first key named Message contains the String value, Message from ViewData ; and the second key named CurrentTime contains the value, DateTime .Now.

## Slide 16

Let us understand how to retrieve data from ViewData.

### Passing Data from a Controller to a View 4-14

- Following code shows retrieving the values present in ViewData:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Index View</title>
  </head>
  <body>
    <p> @ViewData["Message"] </p>
    <p> @ViewData["CurrentTime"] </p>
  </body>
</html>
```

◆ In this code, ViewData is used to display the values of the Message and CurrentTime keys.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 16

In slide 16, refer to the code that shows how to retrieve the values present in ViewData.

Tell them that in this code, ViewData is used to display the values of the Message and CurrentTime keys.

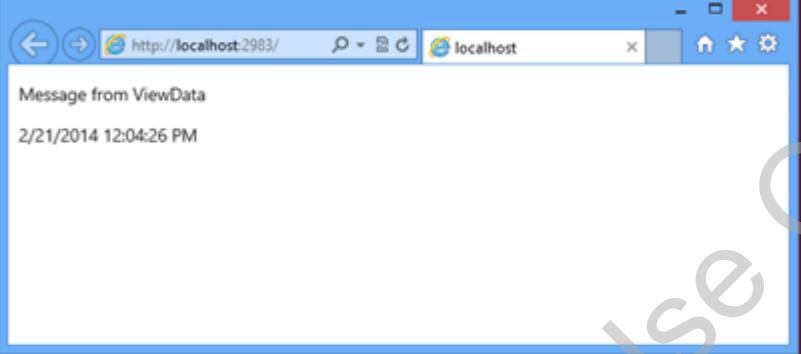
Refer to figure displayed on the next slide that shows the output of the ViewData. Tell the students that ViewData object can be used to pass on values or objects from one view to another view.

## Slide 17

Let us see the output of ViewData.

**Passing Data from a Controller to a View 5-14**

- Following figure shows the output of the ViewData:



The screenshot shows a Microsoft Edge browser window with the URL <http://localhost:2983/>. The page content is "Message from ViewData" followed by the date and time "2/21/2014 12:04:26 PM". The browser interface includes standard navigation buttons, a search bar, and a tab labeled "localhost".

© Aptech Ltd. Views in ASP.NET MVC/Session 3 17

Using slide 17, show the students the output and explain to them what is the role of ViewData. Tell them that when we need to pass some data or values from one view to another view, we can make use of the ViewData object in MVC to transfer such data. In this example, the output shows the message and current date and time passed using ViewData.

## Slide 18

Let us understand ViewBag.

### Passing Data from a Controller to a View 6-14

- ◆ **ViewBag:**
  - ◆ Is a wrapper around ViewData.
  - ◆ Exists only for the current request and becomes null if the request is redirected.
  - ◆ Is a dynamic property based on the dynamic features introduced in C# 4.0.
  - ◆ Does not require typecasting when you use complex data type.
- ◆ The general syntax of ViewBag is as follows:

Syntax:

```
ViewBag.<Property> = <Value>;
```

where,

- ◆ Property: Is a String value that represents a property of ViewBag.
- ◆ Value: Is the value of the property of ViewBag.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 18

In slide 18, tell the students that ViewBag is a wrapper around ViewData. It exists only for the current request and becomes null if the request is redirected.

Tell them that ViewBag is a dynamic property based on the dynamic features introduced in C# 4.0. It does not require typecasting when you use complex data type.

Refer to the syntax of ViewBag object displayed on the slide and explain it.

## Slide 19

Let us understand how to read properties from ViewBag.

### Passing Data from a Controller to a View 7-14

- Following code shows a ViewBag with two properties in the Index action method of the HomeController class:

**Code Snippet:**

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        ViewBag.Message = "Message from ViewBag";
        ViewBag.CurrentTime = DateTime.Now;
        return View();
    }
}
```

In this code a ViewBag is created with the following two properties:

  - The first property named Message contains the String value, 'Message from ViewBag'.
  - The second property named CurrentTime contains the value, DateTime.Now.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 19

In slide 19, refer to the code that shows a ViewBag with two properties in the Index action method of the HomeController class.

Tell the students that in this code, a ViewBag object is created with the following two properties: The first property named Message contains the String value, 'Message from ViewBag'. The second property named CurrentTime contains the value, DateTime.Now.

## Slide 20

Let us learn to retrieve values from ViewBag .

### Passing Data from a Controller to a View 8-14

- Following code shows retrieving the values present in ViewBag:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Index View</title>
  </head>
  <body>
    <p>
      @ViewBag.Message
    </p>
    <p>
      @ViewBag.CurrentTime
    </p>
  </body>
</html>
```

- In this code, ViewBag is used to display the values of Message and CurrentTime properties.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 20

In slide 20, refer to the code that shows how to retrieve the values present in ViewBag.

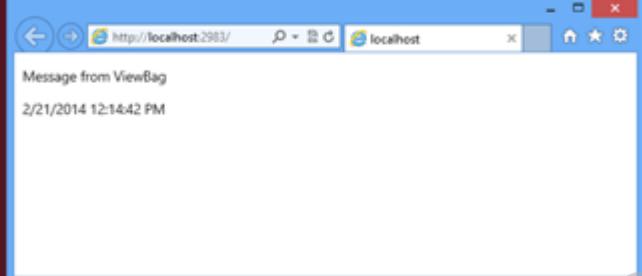
Tell them that In this code, the ViewBag object is used to display the values of Message and CurrentTime properties. Most of the time, ViewData and ViewBag are similar but ViewData is a Dictionary object that is derived from ViewDataDictionary object, while ViewBag is a dynamic property that takes advantage of dynamic features in Visual C# 4.0. ViewData requires typecasting while reading or getting data, whereas in the case of ViewBag , typecasting is not required. The data returned by ViewBag is automatically typecasted to the type of receiving object.

**Slide 21**

Let us see the output after retrieving ViewBag .

**Passing Data from a Controller to a View 9-14**

- ◆ Following figure shows the output of the ViewBag:



- ◆ When you use a ViewBag to store a property and its value in an action, that property can be accessed by both ViewBag and ViewData in a view.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 21

In slide 21, refer to the figure that shows the output of the ViewBag object.

Then, tell them that when they use a ViewBag to store a property and its value in an action, that property can be accessed by both ViewBag and ViewData in a view.

## Slide 22

Let us understand how a controller action store a ViewBag property.

### Passing Data from a Controller to a View 10-14

- Following code shows a controller action storing a ViewBag property:

**Code Snippet:**

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        ViewBag.CommonMessage = "Common message accessible to both
ViewBag and ViewData";
        return View();
    }
}
```

- In this code, a ViewBag is created with a property named CommonMessage.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 22

In slide 22, refer to the code that shows a controller action storing a ViewBag property.

Tell the students that in this code, a ViewBag is created with a property named CommonMessage. A message is stored that can be used later or passed to View. A view can receive and read or utilise the passed data.

## Slide 23

Let us understand a view that uses both ViewData and ViewBag.

### Passing Data from a Controller to a View 11-14

- Following code shows a view that uses both ViewData and ViewBag to access the CommonMessage property stored in ViewBag:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
  <head>
    <title>Index View</title>
  </head>
  <body>
    <p>
      <em>Accessed from ViewData:</em> @ViewData["CommonMessage"]
    </p>
    <p>
      <em>Accessed from ViewBag:</em> @ViewBag.CommonMessage
    </p>
  </body>
</html>
```

- This code uses both ViewData and ViewBag to display the value of the CommonMessage property stored in ViewBag.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 23

In slide 23, refer to the code that shows a view that uses both ViewData and ViewBag to access the CommonMessage property stored in ViewBag.

Refer to the figure displayed on the next slide that shows the output of ViewData and ViewBag.

## Slide 24

Let us see the output of the ViewBag and ViewData.

**Passing Data from a Controller to a View 12-14**

- ◆ Following figure shows output of ViewData and ViewBag:

The screenshot shows a Microsoft Edge browser window with the URL `http://localhost:2983/`. The page content displays two lines of text: "Accessed from ViewData: Common message accessible to both ViewBag and ViewData" and "Accessed from ViewBag: Common message accessible to both ViewBag and ViewData". The browser interface includes standard controls like back, forward, search, and refresh, along with a tab labeled "localhost". At the bottom of the slide, there is footer text: "© Aptech Ltd.", "Views in ASP.NET MVC/Session 3", and the number "24". A large watermark reading "For Aptech Centre Use Only" is diagonally across the slide.

Using slide 24, show the output of the ViewBag and ViewData.

## Slides 25 and 26

Let us understand passing data using TempData.

### Passing Data from a Controller to a View 13-14

- ◆ **TempData:**
  - ◆ Is a Dictionary object derived from the `TempDataDictionary` class.
  - ◆ Stores data as key-value pairs.
  - ◆ Allows passing data from the current request to the subsequent request during request redirection.

The general syntax of `TempData` is as follows:

**Syntax:**

```
TempData[<Key>] = <Value>;
```

where,

- ◆ Key: Is a String value to identify the object present in `TempData`.
- ◆ Value: Is the object present in `TempData`.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 25

In slide 25, tell the students that `TempData` is a Dictionary object derived from the `TempDataDictionary` class. It stores data as key-value pairs.

Tell them that `TempData` object allows passing data from the current request to the subsequent request during request redirection.

Explain them the syntax of using `TempData` as displayed on the slide.

## Passing Data from a Controller to a View 14-14

- ♦ Following code shows how to use TempData to pass values from one view to another view through request redirection:

**Code Snippet:**

```
public class HomeController : Controller {
    public ActionResult Index() {
        ViewData["Message"] = "ViewData Message";
        ViewBag.Message = "ViewBag Message";
        TempData["Message"] = "TempData Message";
        return RedirectToAction("Home/About");
    }
    public ActionResult About() {
        return View();
    }
}
```

- ♦ This code creates two actions. The Index action stores value to ViewData, ViewBag, and TempData objects. Then, redirects the request to the About action by calling the Redirect() method. The About Action returns the corresponding view, which is the About.cshtml view.

© Aptech Ltd. 26

Use slide 26 to refer to the code that shows how to use TempData to pass values from one view to another view through request redirection.

Refer to the code and tell them that this code creates two actions. The Index action stores value to ViewData, ViewBag, and TempData objects. Then, redirects the request to the About action by calling the Redirect() method. The About action returns the corresponding view, which is the About.cshtml view.

## Slides 27 to 31

Let us understand partial view.

The slide has a dark red background with white text. At the top center, it says "Using Partial Views 1-5". Below that, there are two main bullet points:

- ◆ In an ASP.NET MVC application, a partial view:
  - ◆ Represents a sub-view of a main view.
  - ◆ Allows you to reuse common markups across the different views of the application.
- ◆ To create a partial view in Visual Studio .NET, you need to perform the following steps:
  1. Right-click the Views/Shared folder in the Solution Explorer window and select Add→View. The Add View dialog box is displayed.
  2. In the Add View dialog box, specify a name for the partial view in the View Name text field.
  3. Select the Create as a partial view check box.

At the bottom left, it says "© Aptech Ltd.". In the center, it says "Views in ASP.NET MVC/Session 3". At the bottom right, it says "27".

In slide 27, explain the students that in an ASP.NET MVC application, a partial view represents a sub-view of a main view. It allows them to reuse common mark-ups across the different views of the application. Explain them the steps displayed on the slide to create a partial view.

Refer to the figure displayed on the next slide that shows how to create a partial view in Visual Studio 2013.

### Using Partial Views 2-5

- Following figure shows how to create a partial view in Visual Studio 2013:

4. Click Add button to create the partial view.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 28

Using slide 28, show the dialog box **Add View** where you can specify the details such as **View name**, **View engine**, and so on.

### Using Partial Views 3-5

- In the partial view, you can add the markup that you need to display in the main view, as shown in the following code:

```
<h3> Content of partial view. </h3>
```

- The general syntax of partial view is as follows:

**Syntax:**

```
@Html.Partial(<partial_view_name>)
```

where,

- partial\_view\_name: Is the name of the partial view without the .cshtml file extension.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 29

In slide 29, tell the students that in a partial view, they can add the mark-up that they need to display in the main view, as shown in the following code:

<h3> Content of partial view. </h3>

Refer to the code displayed on the slide and explain it.

### Using Partial Views 4-5

- Following code shows a main view, named Index.cshtml that accesses the partial view, named \_TestPartialView.cshtml:

Code Snippet:

```
<!DOCTYPE html>
<html>
  <head>
    <title>Index View</title>
  </head>
  <body>
    <h1>
      Welcome to the Website
    </h1>
    <div>@Html.Partial("_TestPartialView")</div>
  </body>
</html>
```

- This code shows the markup of the main Index.cshtml view that displays a welcome message and renders the partial view, named \_TestPartialView.cshtml.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 30

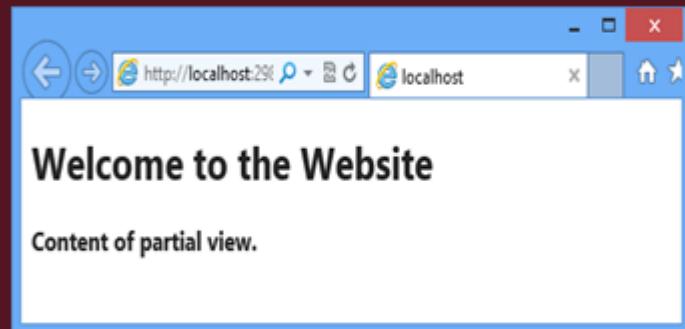
In slide 30, refer to the code displayed on the slide that shows a main view, named Index.cshtml that accesses the partial view, named \_TestPartialView.cshtml.

Explain them that this code shows the mark-up of the main Index.cshtml view that displays a welcome message and renders the partial view, named \_TestPartialView.cshtml.

Refer to the figure displayed on the next slide that shows the output of the main view, named Index.cshtml.

### Using Partial Views 5-5

- Following figure shows the output of the main view, named Index.cshtml:



Using slide 31, show the output of main view and partial view.

## Slides 32 and 33

Let us understand Razor engine.

The slide has a dark blue header with the title "Razor 1-2" in white. Below the header is a dark maroon section containing bullet points about the Razor engine. At the bottom of this section is a blue box labeled "Code Snippet". Inside the "Code Snippet" box is a block of C# Razor code. The footer of the slide is dark maroon and contains copyright information and a page number.

**Razor 1-2**

- ◆ **Razor:**
  - ◆ Is a syntax, based on the ASP.NET Framework that allows creating views.
  - ◆ Is simple and easy to understand for users who are familiar with the C#.NET or VB.NET programming languages.
  - ◆ Following code snippet shows a simple razor view containing both HTML markups for presentation and C# code:

**Code Snippet:**

```
@{var products = new string[] {"Laptop", "Desktop", "Printer"};}  
<html>  
  <head><title>Test View</title></head>  
  <body>  
    <ul>  
      @foreach (var product in products){  
        <li>The product is @product.</li>  
      }  
    </ul>  
  </body>  
</html>
```

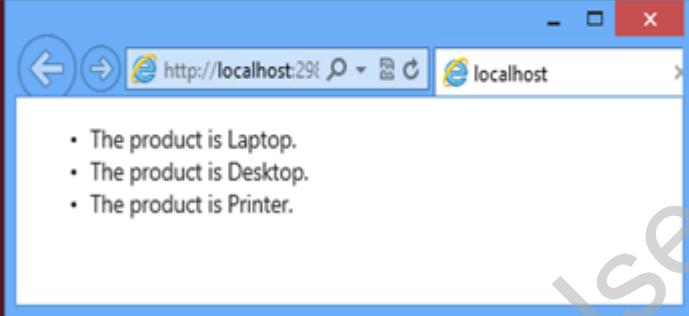
© Aptech Ltd. Views in ASP.NET MVC/Session 3 32

In slide 32, tell the students that Razor is a syntax based on the ASP.NET Framework that allows creating views. It is simple and easy to understand for users who are familiar with the C#.NET or VB.NET programming languages.

Next, refer to the code displayed on the slide that shows a simple razor view containing both HTML mark-ups for presentation and C# code.

## Razor 2-2

- ◆ In the preceding code:
  - ◆ A string [] is declared and initialized using Razor syntax.
  - ◆ Then, Razor syntax is used to iterate through the elements of the array and display each element.
  - ◆ The remaining code in the view is HTML code that displays the page title, body, and the array elements in a bullet list.
- ◆ Following figure shows the output of simple razor view:



The product is Laptop.  
The product is Desktop.  
The product is Printer.

Using slide 33, explain the code displayed on slide 32.

Then, explain them that in the code, a `string[]` is declared and initialized using Razor syntax. Then, Razor syntax is used to iterate through the elements of the array and display each element. The remaining code in the view is HTML code that displays the page title, body, and the array elements in a bullet list.

Then, refer to the figure displayed on the slide that shows the output of simple razor view.

## Slide 34

Let us understand Razor engine.

The slide has a dark red background with white text. At the top right, it says 'Razor Engine'. Below that is a bulleted list of features:

- ◆ The MVC Framework uses a view engine to convert the code of a view into HTML markup that a browser can understand.
- ◆ Razor engine:
  - ◆ Is used as the default view engine by the MVC Framework.
  - ◆ Compiles a view of your application when the view is requested for the first time.
  - ◆ Delivers the compiled view for subsequent requests until you make changes to the view.
  - ◆ Does not introduce a new set of programming language, but provides template markup syntax to segregate HTML markup and programming code in a view.
  - ◆ Supports Test Driven Development (TDD) which allows you to independently test the views of an application.

At the bottom left is '© Aptech Ltd.', in the center is 'Views in ASP.NET MVC/Session 3', and at the bottom right is '34'.

In slide 34, explain to the students that the MVC Framework uses a view engine to convert the code of a view into HTML mark-up that a browser can understand.

Tell them that Razor engine is used as the default view engine by the MVC Framework. It compiles a view of an application when the view is requested for the first time. Razor engine delivers the compiled view for subsequent requests until users make changes to the view.

Next, tell them that Razor engine does not introduce a new set of programming language, but provides template mark-up syntax to segregate HTML mark-up and programming code in a view. It supports Test Driven Development (TDD) which allows students to independently test the views of an application.

### Additional Information:

Apart from the Razor view engine, MVC also supports the ASPX view engine. Prior to MVC 3, ASPX was the only supported view engine. Razor was introduced with MVC 3 and is the default view engine moving forward.

## Slide 35 to 39

Let us understand Razor syntax rules.

**Razor Syntax Rules 1-5**

- ◆ A Razor:
  - ◆ First requires identifying the server-side code from the markup code to interpret the server-side code embedded inside a view file.
  - ◆ Uses the @ symbol, to separate the server-side code from the markup code.
- ◆ While creating a Razor view, you should consider following rules:
  - ◆ Enclose code blocks between @{} and {}
  - ◆ Start inline expressions with @
  - ◆ Variables are declared with the var keyword
  - ◆ Enclose strings in quotation marks
  - ◆ End a Razor code statement with semicolon (;)
  - ◆ Use the .cshtml extension to store a Razor view file that uses C# as the programming language
  - ◆ Use the .vbhtml extension to store a Razor view file that uses VB as the programming language.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 35

In slide 35, explain the students that a Razor first requires identifying the server-side code from the markup code to interpret the server-side code embedded inside a view file. It uses the @ symbol, to separate the server-side code from the markup code.

Tell them that while creating a Razor view, they should consider the following rules:

- Enclose code blocks between @{} and {}
- Start inline expressions with @
- Variables are declared with the var keyword
- Enclose strings in quotation marks
- End a Razor code statement with semicolon (;)
- Use the .cshtml extension to store a Razor view file that uses C# as the programming language
- Use the .vbhtml extension to store a Razor view file that uses VB as the programming language

## Razor Syntax Rules 2-5

- ♦ Razor supports code blocks within a view. A code block is a part of a view that only contains code written in C# or VB.
- ♦ The general syntax of using Razor is as follows:

**Syntax:**

```
@{ <code>}
```

where,

- code: is the C# or VB code that will execute on the server.

- ♦ Following code snippet shows two single-statement code blocks in a Razor view:

**Code Snippet:**

```
@{ var myMessage = "Hello World"; }
{@{ var num = 10; }}
```

• This code shows two single-statements that declares the variables, myMessage and num. The @{ characters mark the beginning of each code and the } character marks the end of the code.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 36

In slide 36, explain that Razor supports code blocks within a view. A code block is a part of a view that only contains code written in C# or VB.

Refer to the first code that shows the syntax of using Razor.

Then, refer to the second code that shows two single-statement code blocks in a Razor view.

Tell them that this code shows two single-statements that declares the variables, myMessage and num. The @{ characters mark the beginning of each code and the } character marks the end of the code.

### Razor Syntax Rules 3-5

- ◆ Razor also supports multi-statement code blocks where each code block can have multiple statements.
- ◆ Multi-statement code block allows you to ignore the use of the @ symbol in every line of code.
- ◆ Following code snippet shows a multi-statement code block in a Razor view:

#### Code Snippet:

```
@{  
    var myMessage = "Hello World";  
    var num = 10;  
}
```

- ◆ This code shows a multi-statement code block that declares the variables, myMessage and num.

In slide 37, explain to the students that Razor also supports multi-statement code blocks where each code block can have multiple statements. Multi-statement code block allows them to ignore the use of the @ symbol in every line of code.

Then, refer to the code displayed on the slide.

Tell them that this code shows a multi-statement code block that declares the variables, myMessage and num.

### Razor Syntax Rules 4-5

- ◆ Similar to code block, Razor uses the @ character for an inline expression.
- ◆ Following code snippet shows using inline expressions:

#### Code Snippet:

```
@{  
    var myMessage = "Hello World";  
    var num = 10;  
}  
@myMessage is numbered @num.
```

- ◆ This code uses two inline expressions that evaluates the `myMessage` and `num` variables.
- ◆ When the Razor engine encounters the @ character, it interprets the immediately following variable name as server-side code and ignores the following text.

In slide 38, explain to the students that similar to code block, Razor uses the @ character for an inline expression.

Next, refer to the code displayed on the slide and explain them that this code uses two inline expressions that evaluate the `myMessage` and `num` variables. When the Razor engine encounters the @ character, it interprets the immediately following variable name as server-side code and ignores the following text.

## Razor Syntax Rules 5-5

- ◆ At times, you may also require overriding the logic that Razor uses to identify the server-side code.
- ◆ Following code snippet shows including the @ symbol in an e-mail address:

**Code Snippet:**

```
<h3>The email ID is: john@mvcexample.com </h3>
```

- ◆ In this code, Razor:
  - ◆ Interprets the first @ symbol as an escape sequence character.
  - ◆ Uses an implicit code expression to identify the part of a server-side code.
- ◆ Sometimes, Razor may also interpret an expression as markup instead of running as server-side code.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 39

In slide 39, explain to the students that at times, they may also require overriding the logic that Razor uses to identify the server-side code.

Refer to the code that shows how to include the @ symbol in an e-mail address.

Explain them that in this code, Razor interprets the first @ symbol as an escape sequence character. Then, it uses an implicit code expression to identify the part of a server-side code.

Also mention that sometimes, Razor may also interpret an expression as mark-up instead of running as server-side code.

## Slides 40 and 41

Let us understand Variables.

The slide has a dark blue header with the title "Variables 1-2". The main content area is white with a dark blue sidebar on the left. The sidebar contains three bullet points:

- ◆ Are used to store data.
- ◆ In Razor, you declare and use a variable in the same way as you do in a C# program.
- ◆ Following code snippet shows declaring variables using Razor:

A blue box labeled "Code Snippet:" contains the following C# Razor code:

```
<!DOCTYPE html>
<html><body>
@{
    var heading = "Using variables";
    string greeting = "Welcome ASP.NET MVC Application";
    int num = 103;
    DateTime today = DateTime.Today;
    <h3>@heading</h3>
    <p>@greeting</p>
    <p>@num</p>
    <p>@today</p>
}
</body></html>
```

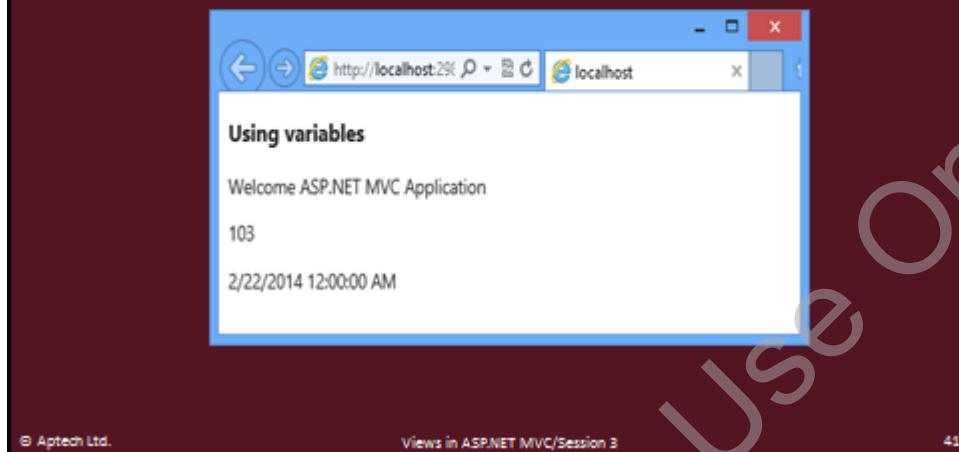
At the bottom of the slide, there is footer text: "© Aptech Ltd.", "Views in ASP.NET MVC/Session 3", and "40".

Use slide 40 to explain that variables are used to store data. In Razor, they declare and use a variable in the same way as they do in a C# program.

Refer to the code that shows how to declare variables using Razor.

## Variables 2-2

- ◆ The preceding code shows declaring four variables, such as heading, greeting, num, and today using Razor.
- ◆ Following figure shows the output of declaring variables using Razor:



In slide 41, explain the code displayed on slide 40 that declares four variables, such as heading, greeting, num, and today using Razor.

Refer to the figure displayed on the slide that shows the output of declaring variables using Razor.

## Slides 42 to 45

Let us understand Loops.

**Loops 1-4**

- ◆ While developing an ASP.NET MVC Web application, you might require executing same statement continually.
- ◆ In such scenario you can use loops.
- ◆ C# supports the following four types of loop constructs:
  - ◆ The `while` loop
  - ◆ The `for` loop
  - ◆ The `do...while` loop
  - ◆ The `foreach` loop

© Aptech Ltd. Views in ASP.NET MVC/Session 3 42

In slide 42, explain to the students that while developing an ASP.NET MVC Web application, they might require executing same statement continually. In such scenario, they can use loops.

Tell them that C# supports the four types of loop constructs, such as `while` loop, `for` loop, `do...while` loop, and `foreach` loop.

## Loops 2-4

- ◆ **while loop:**
  - ◆ Is used to execute a block of code repetitively as long as the condition of the loop remains true.
  - ◆ Uses the **while** keyword at the beginning followed by parentheses.
  - ◆ Allows you to specify the number of time the loop continues inside the parentheses, then, a block to repeat.
  - ◆ Following code snippet shows the Razor code that uses a **while** loop to print the even numbers from 1 to 10:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
<body>
@{ var b = 0;
while (b < 7) {
    b += 1;
    <p>Text @b</p>
}
</body>
</html>
```

© Aptech Ltd. Views in ASP.NET MVC/Session 3

43

In slide 43, explain to the students that **while** loop is used to execute a block of code repetitively as long as the condition of the loop remains true. **While** loop uses the **while** keyword at the beginning followed by parentheses. It allows them to specify the number of time the loop continues inside the parentheses, then a block to repeat.

Refer to the code that shows the Razor code that uses a **while** loop to print the even numbers from 1 to 10.

Then, refer to the figure displayed on the next slide that shows the output of the razor code using **while** loop.

### Loops 3-4

- Following figure shows the output of the razor code using while loop:

The screenshot shows a browser window titled "While Loop". The page content is "Even Numbers" followed by a list of even numbers: 2, 4, 6, 8, and 10.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 44

Using slide 44, show the output of the code that is written on slide 43, using while loop. It displays the first five even numbers i.e. 2, 4, 6, 8, and 10.

### Loops 4-4

- When you know the number of time that a statement should execute, you can use a for loop.
- Similar to the while loop, you can use the for loop to iterate through elements.
- Following code snippet shows the Razor code that uses a for loop to print the even numbers from 1 to 10.

**Code Snippet:**

```
<!DOCTYPE html>
<html>
<body>
    <h1>Even Numbers</h1>
    @for (var num=1; num <= 11; num++) {
        if ((num % 2) == 0)
            <p>@num</p>
    }
</body>
</html>
```

- The code shows the for loop to print the even numbers from 1 to 10 using Razor.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 45

In slide 45, explain to the students that when they know the number of time that a statement should execute, they can use a for loop. Similar to the while loop, they can use the for loop to iterate through elements. Refer to the code that shows the Razor code that uses a for loop to print the even numbers from 1 to 10.

## Slides 46 to 49

Let us understand conditional statements.

### Conditional Statements 1-4

- ◆ Allows you to display dynamic content based on some specific conditions.
- ◆ Includes the `if` statement that returns `true` or `false`, based on the specified condition.
- ◆ Following code snippet shows using the `if...else` statements using Razor:

**Code Snippet:**

```
<!DOCTYPE html>
@[var mark=60;]
<html>
<body>
@if (mark>80)  {
    <p>You have failed in the exam.</p>
} else {
    <p>You have passed the exam.</p>
}
</body>
</html>
```

- ◆ The code uses the `if...else` statements using Razor.

© Aptech Ltd.      Views in ASP.NET MVC/Session 3      46

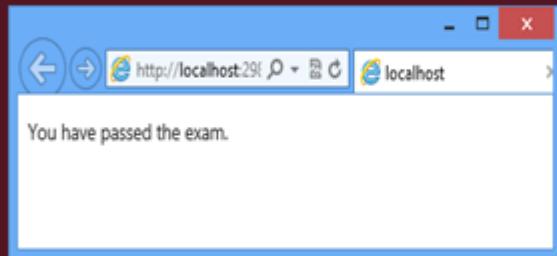
In slide 46, explain to the students that a conditional statement allows them to display dynamic content based on some specific conditions. It includes the `if` statement that returns `true` or `false`, based on the specified condition.

Refer to the code that shows how to use the `if...else` statements using Razor.

Then, refer to the figure displayed on the next slide that shows the output of the code using the `if...else` statements using Razor.

## Conditional Statements 2-4

- Following figure shows the output of the code using the if...else statements using Razor:



- You can also test a number of individual conditions in your code by using a switch statement in the view.

© Aptech Ltd.

Views in ASP.NET MVC/Session 3

47

In slide 47, explain to the students that they can also test a number of individual conditions in their code by using a switch statement in the view.

## Conditional Statements 3-4

- Following code snippet shows using the switch statement using Razor:

### Code Snippet:

```
<!DOCTYPE html>
 @{
    var day=DateTime.Now.DayOfWeek.ToString();
    var msg="";
    <html> <body>
        @switch(day)
        {
            case "Monday":
                msg="Today is Monday, the first working day.";
                break;
            case "Friday":
                msg="Today is Friday, the last working day.";
                break;
            default:
                msg="Today is " + day;
                break;
        }
    <p>@msg</p> </body> </html>
```

© Aptech Ltd.

Views in ASP.NET MVC/Session 3

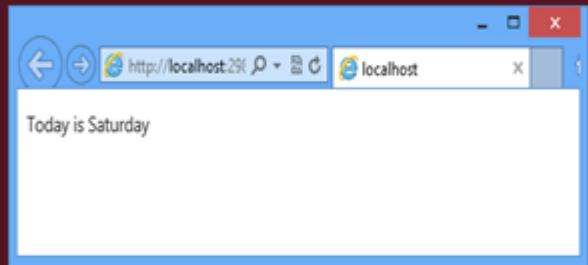
48

Use slide 48 to refer to the code that shows how to use the switch statement using Razor.

Refer to figure displayed on the next slide that shows the output of the code using switch statement.

### Conditional Statements 4-4

- Following figure shows the output of the code using switch statement:



The screenshot shows a Microsoft Edge browser window with the URL `http://localhost:2900` in the address bar. The page content area displays the text "Today is Saturday". The browser has a standard toolbar with back, forward, and search buttons, and a tab labeled "localhost".

© Aptech Ltd. Views in ASP.NET MVC/Session 3 49

Using slide 49, show the output of code that demonstrates use of switch statement in C#.

## Slides 50 and 51

Let us understand Arrays.

**Arrays 1-2**

- ◆ Allows you to store similar variables.
- ◆ Following code snippet shows using arrays to store similar variables using Razor:

**Code Snippet:**

```
<!DOCTYPE html>
@{
    string[] members = {"Joe", "Mark", "Stella"};
    int i = Array.IndexOf(members, "Stella") + 1;
    int len = members.Length;
    string x = members[2 - 1];
}
<html> <body>
<h3>Student Details</h3>
@foreach (var person in members) {
<p>@person</p>
}
<p>The number of students in class are @len</p>
<p>The student at position 2 is @x</p>
<p>Stella is now in position @i</p>
</body>
</html>
```

© Aptech Ltd. Views in ASP.NET MVC/Session 3 30

In slide 50, explain to the students that an array allows storing similar variables. Refer to the code that shows how to use the arrays to store similar variables using Razor. Refer to the figure displayed on the next slide that shows the output of arrays that stores similar variables.

**Arrays 2-2**

- ◆ Following figure shows the output of arrays that stores similar variables:

© Aptech Ltd. Views in ASP.NET MVC/Session 3 31

Using slide 51, show the output of the code. This is an example that shows the contents of an array that store similar variables.

## Slide 52

Let us understand HTML helper methods.

**HTML Helper Methods 1-17**

- ◆ In ASP.NET MVC Framework, helper methods:
  - ◆ Are extension methods to the `HtmlHelper` class, can be called only from views.
  - ◆ Simplifies the process of creating a view.
  - ◆ Allows generating HTML markup that you can reuse across the Web application.
- ◆ Some of the commonly used helper methods while developing an MVC application are as follows:
  - ◆ `Html.ActionLink()`
  - ◆ `Html.BeginForm()` and `Html.EndForm()`
  - ◆ `Html.Label()`
  - ◆ `Html.TextBox()`
  - ◆ `Html.TextArea()`
  - ◆ `Html.Password()`
  - ◆ `Html.CheckBox()`

© Aptech Ltd. Views in ASP.NET MVC/Session 3 52

In slide 52, explain to the students that in ASP.NET MVC Framework, helper methods are extension methods to the `HtmlHelper` class, can be called only from views. These methods simplify the process of creating a view and allow generating HTML mark-up that they can reuse across the Web application.

Tell them about the following commonly used helper methods:

- `Html.ActionLink()`
- `Html.BeginForm()` and `Html.EndForm()`
- `Html.Label()`
- `Html.TextBox()`
- `Html.TextArea()`
- `Html.Password()`
- `Html.CheckBox()`

### Discussion:

Initiate a discussion by explaining that though developing views using HTML is quite simple, yet it often becomes difficult to handle specific issues using HTML, such as ensuring that a URL within a link actually points to the correct location and form elements have proper names and are bound to the correct values to implement model binding. Conclude the discussion by explaining that to overcome these problems, ASP.NET MVC offers a standard set of helper methods that helps generating links and URLs from the routing configuration.

## Slides 53 and 54

Let us understand `Html.ActionLink()` helper method.

**HTML Helper Methods 2-17**

- ◆ `Html.ActionLink()` helper method allows you to generate a hyperlink that points to an action method of a controller class.
- ◆ The general syntax of the `Html.ActionLink()` helper method is as follows:

Syntax:

```
@Html.ActionLink(<link_text>, <action_method>,
<optional_controller>)
```

where,

- ◆ `link_text`: Is the text to be displayed as a hyperlink.
- ◆ `action_method`: Is the name of the action method that acts as the target for the hyperlink.
- ◆ `optional_controller`: Is the name of the controller that contains the action method that will get called by the hyperlink. This parameter can be omitted if the action method getting called is in the same controller as the action method whose view renders the hyperlink.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 53

In slide 53, explain to the students that the `Html.ActionLink()` helper method allows them to generate a hyperlink that points to an action method of a controller class.

Refer to the code that shows the syntax of the `Html.ActionLink()` helper and explain it as displayed on slide.

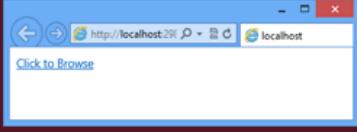
**HTML Helper Methods 3-17**

- ◆ Following code snippet shows using a `Html.ActionLink()` helper method:

Code Snippet:

```
<!DOCTYPE html>
<html>
<body>
    @Html.ActionLink("Click to Browse", "Browse", "Home")
</body>
</html>
```

- ◆ In this code, the `Click to Browse` text will be displayed as a hyperlink.
- ◆ The `Browse` action method of the `Home` controller acts as the target of the hyperlink, as shown in the following figure:



© Aptech Ltd. Views in ASP.NET MVC/Session 3 54

In slide 54, refer to the code that shows how to use an `Html.ActionLink()` helper method.

Tell them that in this code, the `Click to Browse` text will be displayed as a hyperlink. The `Browse` action method of the `Home` controller acts as the target of the hyperlink.

## Slides 55 and 56

Let us understand `Html.BeginForm()` and `Html.EndForm()` helper methods.

### HTML Helper Methods 4-17

- ◆ `Html.BeginForm()` helper method:
  - ◆ Allows you to mark the start of a form.
  - ◆ Coordinates with the routing engine to generate a URL.
  - ◆ Is responsible for producing the opening `<form>` tag.
- ◆ The general syntax of the `Html.BeginForm()` helper method is as follows:
 

**Syntax:**  
`@{Html.BeginForm(<action_method>,<controller_name>);}`

 where,
  - ◆ `action_method`: Is the name of the action method.
  - ◆ `controller_name`: Is the name of the controller class.
- ◆ Once you use the `Html.BeginForm()` helper method to start a form, you need to end a form using the `Html.EndForm()` helper method.

© Aptech Ltd.      Views in ASP.NET MVC/Session 3      35

In slide 55, explain to the students that the `Html.BeginForm()` helper method allows them to mark the start of a form. It coordinates with the routing engine to generate a URL and responsible for producing the opening `<form>` tag.

Refer to the code that shows the syntax of the `Html.BeginForm()` helper method and explain it.

### HTML Helper Methods 5-17

- ◆ Following code snippet shows using the `Html.BeginForm()` and `Html.EndForm()` helper methods:
 

**Code Snippet:**  

```
<!DOCTYPE html>
<html>
<body>
  @{Html.BeginForm("Browse","Home");}
  <p>Inside Form</p>
  @{Html.EndForm();}
</body>
</html>
```
- ◆ In this code the `Html.BeginForm()` method specifies the `Browse` action of the `Home` controller as the target action method to which the form data will be submitted.
- ◆ You can also avoid using the `Html.EndForm()` helper method to explicitly close the form by using the `@using` statement before the `Html.BeginForm()` method.

© Aptech Ltd.      Views in ASP.NET MVC/Session 3      36

Use slide 56 to refer to the code that shows how to use the `Html.BeginForm()` and `Html.EndForm()` helper methods.

Explain them that in this code, the `Html.BeginForm()` method specifies the `Browse` action of the `Home` controller as the target action method to which the form data will be submitted.

Tell them that they can also avoid using the `Html.EndForm()` helper method to explicitly close the form by using the `@using` statement before the `Html.BeginForm()` method.

For Aptech Centre Use Only

## Slides 57 and 58

Let us understand `Html.Label()` helper method.

### HTML Helper Methods 6-17

- ◆ `Html.Label()` helper method:
  - ◆ Allows you to display a label in a form.
  - ◆ Enables attaching information to other input elements, such as text inputs, and increase the accessibility of your application.
- ◆ The general syntax of the `Html.Label()` helper method is as follows:

Syntax:

```
@Html.Label(<label_text>)
```

where,

- ◆ `label_text`: Is the name of the label.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 57

In slide 57, explain to the students that the `Html.Label()` helper method allows displaying a label in a form. It enables attaching information to other input elements, such as text inputs, and increases the accessibility of an application

Refer to the code that shows the syntax of the `Html.Label()` helper method and explain it.

## HTML Helper Methods 6-17

♦ Following code snippet shows the `Html.Label()` method:

**Code Snippet:**

```
@Html.Label("name")
<!DOCTYPE html>
<html>
<body>
@(Html.BeginForm("Browse", "Home"))
@Html.Label("User Name:")
@(Html.EndForm())
</body>
</html>
```

♦ `Html.TextBox()` helper method:

- ♦ Allows you to display an input tag.
- ♦ Used to accept input from a user.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 58

In slide 58, refer to the code that shows how to use the `Html.Label()` method.

Tell them that `Html.TextBox()` helper method allows them to display an input tag and used to accept input from a user.

### Additional Information:

However, while writing the property name in the `Label()` method, there are chances that the students may misspell the name of the property. Such mistakes would not be detected at the time of compilation of the code. To overcome such problems, use the `Html.LabelFor()` helper method.

## Slide 59

Let us understand `Html.TextBox()` helper method.

### HTML Helper Methods 7-17

- ◆ The general syntax of the `Html.TextBox()` helper method is as follows:  
**Syntax:**

```
@Html.TextBox("textbox_text")
```

where,
  - ◆ `textbox_text`: is the name of the textbox
- ◆ Following code snippet shows using a `Html.TextBox()` method:  
**Code Snippet:**

```
<!DOCTYPE html>
<html>
<body>
    @Html.BeginForm("Browse", "Home")
    @Html.Label("User Name:")
    @Html.TextBox("textBox1")
    <br><br>
    <input type="submit" value="Submit">
    @Html.EndForm()
</body>
</html>
```

© Aptech Ltd. Views in ASP.NET MVC/Session 3 59

In slide 59, refer to the first code that shows the syntax of the `Html.TextBox()` helper method and explain it.

Then, refer the second code that shows how to use an `Html.TextBox()` method.

## Slides 60 and 61

Let us understand `Html.TextArea()` helper method.

### HTML Helper Methods 8-17

- ◆ `Html.TextArea()` helper method:
  - ◆ Allows you to display a `<textarea>` element for multi-line text entry.
  - ◆ Enables you to specify the number of columns and rows to be displayed in order to control the size of the text area.
- ◆ Following code snippet shows using a `Html.TextArea()` method:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
<body>
    @Html.BeginForm("Browse", "Home") :>
    @Html.Label("User Name:") <br>
    @Html.TextBox("textBox1") <br><br>
    @Html.Label("Address:") <br>
    @Html.TextArea("textArea1") <br><br>
    <input type="submit" value="Submit">
    @Html.EndForm() :>
</body>
</html>
```

© Aptech Ltd. Views in ASP.NET MVC/Session 3 60

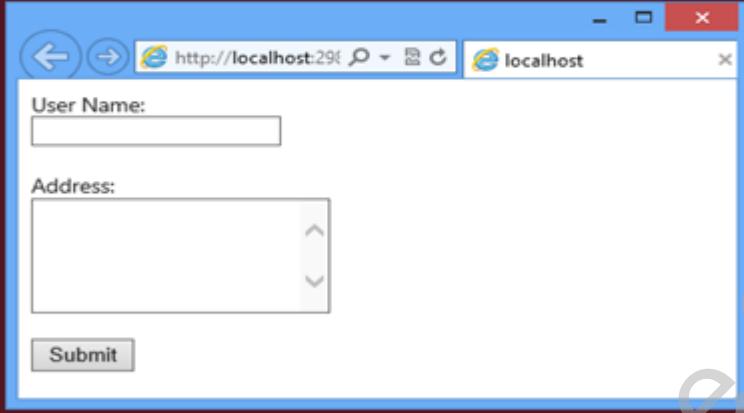
In slide 60, explain to the students that the `Html.TextArea()` helper method allows displaying a `<textarea>` element for multi-line text entry. It enables specifying the number of columns and rows to be displayed in order to control the size of the text area.

Refer to the code that shows how to use a `Html.TextArea()` method and explain it.

Then, refer to the figure displayed on the next slide that shows the output of using the `Html.TextArea()` helper method.

### HTML Helper Methods 9-17

◆ Following figure shows using the `Html.TextArea()` helper method:



The screenshot shows a Microsoft Edge browser window with the URL `http://localhost:294`. The page displays a simple form with two text input fields and a submit button. The first field is labeled "User Name:" and contains a single line of text. The second field is labeled "Address:" and is a multi-line text area with scroll bars. Below the fields is a "Submit" button. The browser status bar at the bottom indicates "Views in ASP.NET MVC/Session 3".

Using slide 61, explain how a TextArea is created using `Html.TextArea()` function. This is similar to using `<TextArea> .....</TextArea>` tag in HTML. The difference between `Html.TextArea()` and simple `<TextArea>` is that, the prior can be controlled and manipulated easily with C# code while the later cannot be manipulated without the help of scripting language.

## Slide 62

Let us understand `Html.Password()` helper method.

### HTML Helper Methods 10-17

- ◆ You can use the `Html.Password()` helper method to display a password field.
- ◆ Following code snippet shows using a `Html.Password()` method:

**Code Snippet:**

```
<!DOCTYPE html>
<html>
<body>
    @(Html.BeginForm("Browse", "Home"))
    @Html.Label("User Name:")
    <br>
    @Html.TextBox("textBox1")
    <br><br>
    @Html.Label("Address:")
    <br> @Html.TextArea("textArea1")
    <br>
    @Html.Label("Password:")
    <br>@Html.Password("password")
    <br><br>
    <input type="submit" value="Submit">
    @(Html.EndForm())
</body>
</html>
```

© Aptech Ltd. Views in ASP.NET MVC/Session 3 62

In slide 62, explain to the students that they can use the `Html.Password()` helper method to display a password field.

Refer to the code that shows how to use an `Html.Password()` method and explain it.

## Slide 63

Let us understand `Html.CheckBox()` helper method.

### HTML Helper Methods 11-17

- ◆ You can use the `Html.CheckBox()` helper method to display a check box that enables the user to select a true or false condition.
- ◆ Following code snippet shows using a `Html.CheckBox()` method:

Code Snippet:

```
<!DOCTYPE html>
<html><body>
@{Html.BeginForm("Browse", "Home");}
@Html.Label("User Name:<br>")
@Html.TextBox("textBox1")<br><br>
@Html.Label("Address:<br>")@Html.TextArea("textArea1")<br><br>
@Html.Label("Password:<br>")@Html.Password("password")<br><br>
@Html.Label("I need updates on my mail:<br>")
@Html.CheckBox("checkbox1")<br><br>
<input type="submit" value="Submit"> @({Html.EndForm()})
</body> </html>
```

- ◆ In this code, the `Html.CheckBox()` helper method renders a hidden input in addition to the check box input. The hidden input ensures that a value will be submitted, even if the user does not select the check box.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 63

In slide 63, explain to the students that they can use the `Html.CheckBox()` helper method to display a check box that enables the user to select a true or false condition.

Refer to the code that shows how to use an `Html.CheckBox()` method.

Then, explain them that in this code, the `Html.CheckBox()` helper method renders a hidden input in addition to the check box input. The hidden input ensures that a value will be submitted, even if the user does not select the check box.

## Slides 64 and 65

Let us understand `Html.DropDownList()` helper method.

### HTML Helper Methods 12-17

- ◆ `Html.DropDownList()` helper method:
  - ◆ Return a `<select>` element that shows a list of possible options and also the current value for a field.
  - ◆ Allows selection of a single item.
- ◆ The general syntax of the `Html.DropDownList()` helper method is as follows:

**Syntax:**

```
@Html.DropDownList("myList", new SelectList(new [] {<value1>, <value2>, <value3>}), "Choose")
```

where,

- ◆ `value1`, `value2`, and `value3` are the options available in the drop-down list.
- ◆ `Choose`: Is the value at the top of the list.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 64

In slide 64, explain to the students that the `Html.DropDownList()` helper method return a `<select>` element that shows a list of possible options and also the current value for a field. It allows selection of a single item.

Refer to the code that shows the syntax of the `Html.DropDownList()` helper method and explain it.

## HTML Helper Methods 13-17

- Following code snippet shows using a `Html.DropDownList()` method:

### Code Snippet:

```
<!DOCTYPE html>
<html><body> @(Html.BeginForm("Browse", "Home")) :
@Html.Label("User Name:")<br>
@Html.TextBox("textBox1")<br><br>
@Html.Label("Address:")<br> @Html.TextArea("textArea1")<br><br>
@Html.Label("Password:")<br>@Html.Password("password")<br><br>
@Html.Label("I need updates on my mail:")
@Html.CheckBox("checkbox1")<br> <br>
@Html.Label("Select your city:")
@Html.DropDownList("myList", new SelectList(new [] {"New York",
"Philadelphia", "California"}), "Choose")<br><br>
<input type="submit" value="Submit">
@(Html.EndForm())
</body></html>
```

- In this code, the `Html.DropDownList()` method creates a drop-down list in a form with `myList` as its name and contains three values that a user can select from the drop-down list.

Use slide 65 to refer to the code that shows how to use an `Html.DropDownList()` method.

Explain to the students that in this code, the `Html.DropDownList()` method creates a drop-down list in a form with `myList` as its name and contains three values that a user can select from the drop-down list.

## Slides 66 to 68

Let us understand `Html.RadioButton()` helper method.

The slide is titled "HTML Helper Methods 14-17". It contains the following text:

- ♦ The `Html.RadioButton()` helper method allows you to provide a range of possible options for a single value.
- ♦ The general syntax of the `Html.RadioButton()` helper method is as follows:

**Syntax:**

```
@Html.RadioButton("name", "value", isChecked)
```

where,

- ♦ `name`: Is the name of the radio button input element.
- ♦ `value`: Is the value associated with a particular radio button option.
- ♦ `isChecked`: Is a Boolean value that indicates whether the radio button option is selected or not.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 66

In slide 66, explain to the students that the `Html.RadioButton()` helper method allows providing a range of possible options for a single value.

Refer to the code that shows the syntax of the `Html.RadioButton()` helper method and explain it.

**HTML Helper Methods 15-17**

- Following code snippet shows using a `Html.RadioButton()` method:

**Code Snippet:**

```
<!DOCTYPE html>
<html> <body>
@Html.BeginForm("Browse", "Home");
@Html.Label("User Name:")<br>
@Html.TextBox("textBox1")<br><br>
@Html.Label("Address:")<br> @Html.TextArea("textArea1")<br><br>
@Html.Label("Password:")<br> @Html.Password("password")<br><br>
@Html.Label("I need updates on my mail:") @Html.CheckBox("checkbox1")<br> <br>
@Html.Label("Select your city: ") @Html.DropDownList("myList", new SelectList(new [] {"New York", "Philadelphia", "California"}), "Choose")<br> </br></br>
Male @Html.RadioButton("Gender", "Male", true)<br>
Female @Html.RadioButton("Gender", "Female")<br> </br>
<input type="submit" value="Submit">
@(Html.EndForm())
</body> </html>
```

- In this code, the `Html.RadioButton()` helper methods is used to create two radio buttonsto accept the gender of a user.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 67

In slide 67, refer to the code that shows how to use an `Html.RadioButton()` method.

Explain to the students that in this code, the `Html.RadioButton()` helper methods is used to create two radio buttons to accept the gender of a user.

Refer to the figure displayed on the next slide that shows how to use the `Html.RadioButton()` helper method.

**HTML Helper Methods 16-17**

- Following figure shows using the `Html.RadioButton()` helper method:

© Aptech Ltd. Views in ASP.NET MVC/Session 3 68

Using slide 68, show the output of `Html.RadioButton()` function. It is similar to creating a radio button using `<input type=radio name=gender>`.

## Slide 69

Let us understand `Url.Action()` helper method.

### HTML Helper Methods 17-17

- ◆ The `Url.Action()` helper method generates a URL that targets a specified action method of a controller.
- ◆ The general syntax of the `Url.Action()` helper method is as follows:

**Syntax:**

```
@Url.Action(<action_name>, <controller_name>)
```

where,

- ◆ `action_name`: Is the name of the action method.
- ◆ `controller_name`: Is the name of the controller class.

- ◆ Following code snippet shows the `Url.Action()` method:

**Code Snippet:**

```
<!DOCTYPE html>
<html> <body>
    <a href="@Url.Action("Browse", "Home")">Browse</a>
</body> </html>
```

- ◆ This code creates a hyperlink that targets the URL generated using the `Url.Action()` method. When a user clicks the hyperlink, the `Browse` action of the `Home` controller will be invoked.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 69

In slide 69, explain to the students that the `Url.Action()` helper method generates a URL that targets a specified action method of a controller.

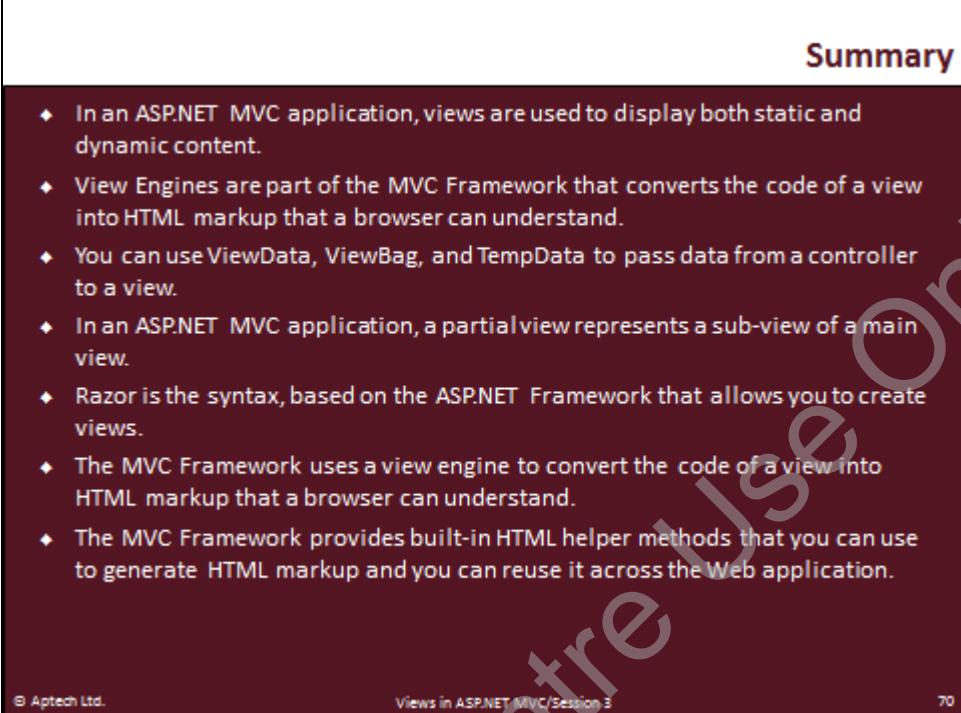
Refer to the first code that shows the syntax of the `Url.Action()` helper method and explain it.

Refer to the second code that shows how to use the `Url.Action()` method.

Explain the students that this code creates a hyperlink that targets the URL generated using the `Url.Action()` method. When a user clicks the hyperlink, the `Browse` action of the `Home` controller will be invoked.

## Slide 70

Let us summarize the session.



The slide has a dark blue header bar with the word "Summary" in white. The main content area is white with a dark blue sidebar on the left. The sidebar contains a list of bullet points about ASP.NET MVC views. At the bottom of the slide, there is a footer bar with three items: "© Aptech Ltd.", "Views in ASP.NET MVC/Session 3", and the number "70". A large watermark reading "Aptech Centre Only" is diagonally across the slide.

**Summary**

- ◆ In an ASP.NET MVC application, views are used to display both static and dynamic content.
- ◆ View Engines are part of the MVC Framework that converts the code of a view into HTML markup that a browser can understand.
- ◆ You can use ViewData, ViewBag, and TempData to pass data from a controller to a view.
- ◆ In an ASP.NET MVC application, a partial view represents a sub-view of a main view.
- ◆ Razor is the syntax, based on the ASP.NET Framework that allows you to create views.
- ◆ The MVC Framework uses a view engine to convert the code of a view into HTML markup that a browser can understand.
- ◆ The MVC Framework provides built-in HTML helper methods that you can use to generate HTML markup and you can reuse it across the Web application.

© Aptech Ltd. Views in ASP.NET MVC/Session 3 70

In slide 70, summarize the session. End the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- In an ASP.NET MVC application, views are used to display both static and dynamic content.
- View engines are part of the MVC Framework that converts the code of a view into HTML markup that a browser can understand.
- You can use ViewData, ViewBag, and TempData to pass data from a controller to a view.
- In an ASP.NET MVC application, a partial view represents a sub-view of a main view.
- Razor is the syntax, based on the ASP.NET Framework that allows you to create views.
- The MVC Framework uses a view engine to convert the code of a view into HTML markup that a browser can understand.
- The MVC Framework provides built-in HTML helper methods that can be used to generate HTML markup and reuse it across the Web application.

### **3.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. How will you render a different view from an action method?
2. How will you use TempData to pass values from one view to another view through request redirection?
3. Which HTML helper method will you use to display the names of model properties?

## Session 4

### Models in ASP.NET MVC

#### **4.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **4.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe models
- Explain how to create a model
- Describe how to pass model data from controllers to views
- Explain how to create strongly typed models
- Explain the role of the model binder
- Explain how to use scaffolding in Visual Studio .NET

#### **4.1.2 Teaching Skills**

To teach this session successfully, you must know about models in an ASP.NET MVC application. You must have knowledge of how to pass model data from controllers to views, strongly typed models, and role of the model binder. You should also be aware of using scaffolding in Visual Studio .NET.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

## In-Class Activities

Follow the order given here during In-Class activities.

### Overview of the Session

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the models in an ASP.NET MVC application. They will learn about how to create a model and pass model data from controllers to views. Then, they will be introduced to strongly typed models and role of the model binder. This session will also discuss about using scaffolding in Visual Studio .NET.

### 4.2 In-Class Explanation

#### Slide 3

Let us understand model.

The slide has a dark red header bar with the title 'Introducing Models' in white. The main content area is dark red with white text. It contains a bulleted list explaining what a model is in an ASP.NET MVC application and defining the three types of models based on the MVC pattern.

**Introducing Models**

- ◆ In an ASP.NET MVC application, a model:
  - ◆ Is a class containing properties that represents data of an application.
  - ◆ Represents data associated with the application.
- ◆ ASP.NET MVC Framework is based on the MVC pattern.
- ◆ The MVC pattern defines the following three types of models, where each model has specific purpose:
  - ◆ Data model: Represent classes that interact with a database. Data models are set of classes that can either follow the database-first approach or code-first approach.
  - ◆ Business model: Represent classes that implement a functionality that represents business logic of an application.
  - ◆ View model: Represent classes that provide information passed between controllers and views.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 3

Use slide 3 to explain that in an ASP.NET MVC application, a model is a class that contains properties that represents data of an application. It represents data associated with the application.

Tell them that ASP.NET MVC Framework is based on the MVC pattern.

In addition, tell them that the MVC pattern defines the following three types of models, where each model has specific purpose:

- **Data model:** Represents classes that interact with a database. Data models are set of classes that can either follow the database-first approach or code-first approach.
- **Business model:** Represents classes that implement a functionality that represents business logic of an application.
- **View model:** Represents classes that provide information passed between controllers and views.

For Aptech Centre Use Only

## Slide 4

Let us understand how to create a model.

### Creating a Model

- ◆ To create a model in an ASP.NET MVC application, you need to:
  - ◆ Create a public class.
  - ◆ Declare public properties for each information that the model represents.
- ◆ Following code snippet shows declaring a model class named User:

**Code Snippet:**

```
public class User
{
    public long Id { get; set; }
    public string name { get; set; }
    public string address { get; set; }
    public string email { get; set; }
}
```

- ◆ This code creates a model class named User that contains the Id, name, address, and email properties declared as public.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 4

In slide 4, explain the students that to create a model in an ASP.NET MVC application, they need to create a public class and declare public properties for each information that the model represents.

Then, refer to the code that shows how to declare a model class named User.

Explain them that this code creates a model class named User that contains the Id, name, address, and email properties declared as public.

## Slide 5

Let us understand how to access a model within a controller.

### Accessing a Model within a Controller

- ◆ In an ASP.NET MVC application when a user request for some information, the request is received by an action method.
- ◆ The action method is used to access the model storing the data.
- ◆ To access the model, you need to create an object of the model class and either retrieve or set the property values of the object.
- ◆ Following code shows the creating an object of the model class in the Index() action method:

**Code Snippet:**

```
public ActionResult Index()
{
    var user = new MVCModelDemo.Models.User();
    user.name = "John Smith";
    user.address = "Park Street";
    user.email = "john@mvcexample.com";
    return View();
}
```

- ◆ In this code the user is an object of the User class and the property values of the model is set to the data related to a user, such as name, address, and e-mail.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 5

In slide 5, tell the students that in an ASP.NET MVC application when a user request for some information, the request is received by an action method. The action method is used to access the model storing the data.

Then, tell them that to access the model, they need to create an object of the model class and either retrieve or set the property values of the object.

Next, refer to the code that shows how to create an object of the model class in the Index() action method.

Explain them that in this code, the user is an object of the User class and the property values of the model is set to the data related to a user, such as name, address, and e-mail.

## Slides 6 to 15

Let us understand how to pass model data from controller to view.

**Passing Model Data from Controller to View 1-10**

- ◆ Once you have accessed the model within a controller, you need to make the model data accessible to a view so that the view can display the data to the user.
- ◆ To do this, you need to pass the model object to the view while invoking the view.
- ◆ You can model the object as follows:
  - ◆ A single object
  - ◆ A collection of model objects

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      6

In slide 6, explain to the students that once they have accessed the model within a controller, they need to make the model data accessible to a view, so that the view can display the data to the user. To do this, they need to pass the model object to the view while invoking the view.

Then, tell them that they can model the object as a single object and a collection of model objects.

## Passing Model Data from Controller to View 2-10

- ♦ In an action method, you can create a model object and then pass the object to a view by using the `ViewBag` object.
- ♦ Following code shows passing the User model data from an action method to a view by using a `ViewBag` object:

### Code Snippet:

```
public ActionResult Index()
{
    var user = new MVCModelDemo.Models.User();
    user.name = "John Smith";
    user.address = "Park Street";
    user.email = "john@mvceexample.com";
    ViewBag.user = user;
    return View();
}
```

- ♦ In this code, an object of the `User` model class is created and initialized with values. The object is then, passed to the view by using a `ViewBag` object.

Use slide 7 to explain the students that in an action method, they can create a model object and then, pass the object to a view by using the `ViewBag` object.

Then, refer to the code that shows how to pass the `User` model data from an action method to a view by using a `ViewBag` object.

Finally, explain them that in this code, an object of the `User` model class is created and initialized with values. The object is then, passed to the view by using a `ViewBag` object.

### Passing Model Data from Controller to View 3-10

- ◆ You can access the data of the model object stored in the `ViewBag` object from within the view.
- ◆ Following code snippet shows accessing the data of the model object stored in the `ViewBag` object:

#### Code Snippet:

```
<!DOCTYPE html>
<html> <body>
<p> User Name: @ViewBag.user.name
</p>
<p> Address: @ViewBag.user.address
</p>
<p> Email: @ViewBag.user.email
</p>
</body> </html>
```

- ◆ In this code, the view accesses and displays the `name`, `address`, and `email` properties of the `User` model object stored in the `ViewBag` object.

In slide 8, tell them that they can access the data of the model object stored in the `ViewBag` object from within the view.

Then, refer to the code displayed on the slide that shows how to access the data of the model object stored in the `ViewBag` object.

Next, explain them that in this code, the view accesses and displays the `name`, `address`, and `email` properties of the `User` model object stored in the `ViewBag` object.

## Passing Model Data from Controller to View 4-10

- Following code shows passing a collection of model objects to a view:

### Code Snippet:

```
public ActionResult Index(){  
    var user = new List<User>();  
    var user1 = new User();  
    user1.name = "Mark Smith";  
    user1.address = "Park Street";  
    user1.email = "Mark@mvcexample.com";  
    var user2 = new User();  
    user2.name = "John Parker";  
    user2.address = "New Park";  
    user2.email = "John@mvcexample.com";  
    var user3 = new User();  
    user3.name = "Steave Edward ";  
    user3.address = "Melbourne Street";  
    user3.email = "steave@mvcexample.com";  
    user.Add(user1); user.Add(user2); user.Add(user3);  
    ViewBag.user = user; return View(); }
```

In slide 9, refer to the code that shows how to pass a collection of model objects to a view.

### Passing Model Data from Controller to View 5-10

- ◆ The preceding code:
  - + Creates and initializes three objects of the model class, named `User`.
  - + Then, a `List<User>` collection is created and the model objects are added to it.
  - + Finally, the collection is passed to the view by using a `ViewBag` object.
  - + Once you pass a collection of model objects to a view using a `ViewBag` object:
    - You can retrieve the collection stored in the `ViewBag` object from within the view.
    - You can iterate through the collection to retrieve each model object.
    - You can access their properties.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      10

In slide 10, explain the code displayed on slide 9 that creates and initializes three objects of the model class, named `User`. Then, a `List<User>` collection is created and the model objects are added to it. Finally, the collection is passed to the view by using a `ViewBag` object.

Then, tell them that once they pass a collection of model objects to a view using a `ViewBag` object, they can retrieve the collection stored in the `ViewBag` object from within the view. They can also iterate through the collection to retrieve each model object and access their properties.

### Passing Model Data from Controller to View 6-10

- ◆ Following code snippet shows retrieving model objects from a collection and displaying their properties.
 

**Code Snippet:**  

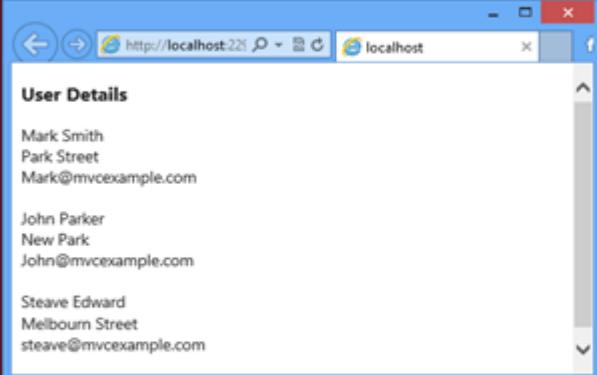
```
<!DOCTYPE html>
<html> <body>
<h3>User Details</h3>
@{ var user = ViewBag.user;
}
@foreach (var p in user)
{
  @p.name<br />
  @p.address<br />
  @p.email<br />
<br />
}
</body> </html>
```
- ◆ This code uses a `foreach` loop to iterate through the collection of model object stored in the `ViewBag` object and the `name`, `address`, and `email` properties are rendered as response.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      11

In slide 11, refer to the code displayed on the slide that shows how to retrieve model objects from a collection and displaying their properties. Explain them that the referred code uses a `foreach` loop to iterate through the collection of model object stored in the `ViewBag` object and the `name`, `address`, and `email` properties are rendered as response.

## Passing Model Data from Controller to View 7-10

♦ Following figure shows the output of retrieving model objects:



The screenshot shows a web browser window with the URL <http://localhost:221>. The page title is "User Details". The content displays three users with their names, addresses, and email addresses:

User	Name	Address	Email
1	Mark Smith	Park Street	Mark@mvceexample.com
2	John Parker	New Park	John@mvceexample.com
3	Steave Edward	Melbourn Street	steave@mvceexample.com

♦ You can use another approach to pass a collection of model objects from an action method to a view is to pass the collection directly as a parameter to the `View()` method.

♦ This code uses a foreach loop to iterate through the collection of model object stored in the `ViewBag` object and the name, address, and email properties are rendered as response.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 12

In slide 12, explain the figure displayed on the slide that shows the output of retrieving model objects.

Then, tell them that they can use another approach to pass a collection of model objects from an action method to a view is to pass the collection directly as a parameter to the `View()` method.

## Passing Model Data from Controller to View 8-10

- Following code snippet shows passing a collection of model objects to a view as a parameter to the `View()` method:

### Code Snippet:

```
<!DOCTYPE html>
<html> <body>
<h3>User Details</h3>
{@ var user = ViewBag.user; }
@foreach (var p in user) {
    @p.name<br />
    @p.address<br />
    @p.email<br />
}

<br /> } </body> </html>public ActionResult Index() {
var user = new List<User>(); var user1 = new User();
user1.name = "Mark Smith"; user1.address = "Park Street";
user1.email = "Mark@mvcexample.com";
var user2 = new User(); user2.name = "John Parker";
    user2.address = "New Park"; user2.email = "John@mvcexample.com";
var user3 = new User(); user3.name = "Steave Edward ";
user3.address = "Melbourn Street"; user3.email = "steave@mvcexample.com";
user.Add(user1); user.Add(user2); user.Add(user3); return View(user); }
```

Use slide 13 to refer to the code that shows how to pass a collection of model objects to a view as a parameter to the `View()` method.

## Passing Model Data from Controller to View 9-10

- ◆ The preceding code will create and initializes three objects of the model class, named User.
  - ◆ Then, a List<User> collection is created and the model objects are added to it.
  - ◆ Finally, the collection is passed to the view as a parameter to the View() method.

### Code Snippet:

```
<!DOCTYPE html>
<html> <body>
<h3>User Details</h3>
@{ var user = ViewBag.user;
}
@foreach (var p in user)
{
    @p.name<br />
    @p.address<br />
    @p.email<br />
<br />
}
</body> </html>
```

In slide 14, explain the code displayed on slide 13 that creates and initializes three objects of the model class, named User. Then, a List<User> collection is created and the model objects are added to it. Finally, the collection is passed to the view as a parameter to the View( ) method.

## Passing Model Data from Controller to View 10-10

- Following code snippet shows retrieving the user information in the view:

### Code Snippet:

```
<!DOCTYPE html>
<html> <body>
<h3>User Details</h3>
{
var user = Model;
}
@foreach(var p in user)
{
    @p.name <br />
    @p.address<br />
    @p.email<br />
<br/>
}
</body>
</html>
```

- This code shows how to retrieve the user information that has been passed to a view by passing a collection of objects as a parameter.

In slide 15, refer to the code displayed on the slide that shows how to retrieve the user information in the view.

Then, explain the students that this code shows how to retrieve the user information that has been passed to a view by passing a collection of objects as a parameter.

## Slides 16 to 21

Let us understand how to use strong typing.

### Using Strong Typing 1-6

- ♦ While passing model data from a controller to a view, the view cannot identify the exact type of the data.
- ♦ As a solution, you can typecast the model data to a specific type.
- ♦ Following code snippet shows how to typecast model data:

**Code Snippet:**

```
<html> <body>
<h3>User Details</h3>
{
    var user = Model as MVCModelDemo.Models.User;
}
@user.name <br/>
@user.address<br/>
@user.email<br/>
</body> </html>
```

♦ In this code, the Model object is cast to the type, `MVCModelDemo.Models.User`. As a result, the User object is created as an object of the type, `MVCModelDemo.Models.User`, and enables compile-time checking of code.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 16

In slide 16, explain to the students that while passing model data from a controller to a view, the view cannot identify the exact type of the data. As a solution, they can typecast the model data to a specific type.

Refer to the code that shows how to typecast model data.

Then, explain them that in this code, the Model object is cast to the type, `MVCModelDemo.Models.User`. As a result, the User object is created as an object of the type, `MVCModelDemo.Models.User`, and enables compile-time checking of code.

## Using Strong Typing 2-6

- ◆ You can also ignore explicit type casting of a model object, by creating a strongly typed view.
- ◆ A strongly typed view specifies the type of a model it requires by using the @model keyword.
- ◆ The general syntax of using the @model keyword is as follows:

**Syntax:**

```
@model <model_name>
```

where,

- ◆ `model_name`: is the fully qualified name of the model class.

- ◆ Once you use the @model keyword, you can access the properties of the model object in the view.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 17

Use slide 17 to explain to the students that they can also ignore explicit type casting of a model object, by creating a strongly typed view. A strongly typed view specifies the type of a model it requires by using the @model keyword.

Refer to the syntax of the @model keyword.

Also mention that once they use the @model keyword, they can access the properties of the model object in the view.

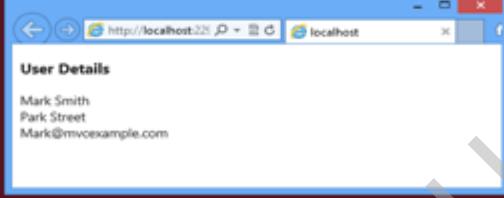
### Using Strong Typing 3-6

♦ Following code snippet shows accessing the properties of the model object by using the @model keyword:

**Code Snippet:**

```
@model MVCModelDemo.Models.User
<html><body>
<h3>User Details</h3>
@Model.name <br/> @Model.address<br/>
@Model.email<br/>
</body> </html>
```

♦ Following figure shows the output of accessing properties of the Model object:



The screenshot shows a Microsoft Internet Explorer window with the URL <http://localhost:224>. The title bar of the window says "User Details". The main content area of the browser displays the following text:  
Mark Smith  
Park Street  
Mark@mvceexample.com

In slide 18, refer to the code that shows how to access the properties of the model object by using the @model keyword.

Then, refer to the figure displayed on the slide that shows the output of accessing properties of the Model object.

**Additional Information:**

Tell the students that to avoid specifying a fully qualified type name for the model, they can use the @ using declaration.

## Using Strong Typing 4-6

- ◆ Sometime, you may need to pass a collection of objects to a view.
- ◆ In such situation, you can use the @model keyword.
- ◆ Following code snippet shows using the @model keyword to pass a collection of Model object:

### Code Snippet:

```
@model IEnumerable<MVCMvcDemo.Models.User>
```

- ◆ This code uses the @model keyword to indicate that it expects a collection of the User model objects.
- ◆ Once you pass a collection of the model objects, you can access it in a view.

In slide 19, explain that sometime, they may need to pass a collection of objects to a view. In such situation, they can use the @model keyword.

Then, refer to the code that shows how to use the @model keyword to pass a collection of Model object.

Then, explain them that this code uses the @model keyword to indicate that it expects a collection of the User model objects.

Mention that once they pass a collection of the model objects, they can access it in a view.

## Using Strong Typing 5-6

- Following code snippet shows accessing the collection of the User model in a view:

### Code Snippet:

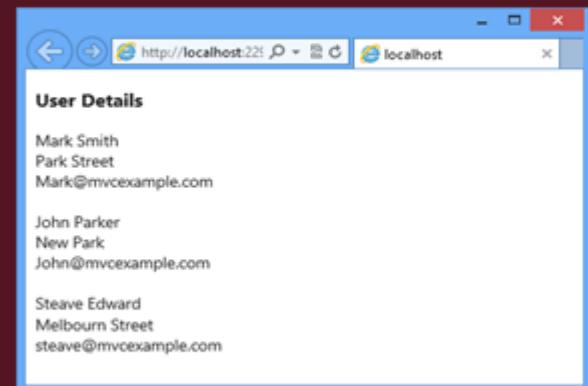
```
@model IEnumerable<MVCMODELDEMO.Models.User>
<html>
<body>
<h3>User Details</h3>
    @{
        var user = Model;
    }
    @foreach(var u in user)
    {
        @u.name <br/>
        @u.address<br/>
        @u.email<br/>
    <br/>
}
</body>
</html>
```

In slide 20, refer to the code that shows how to access the collection of the User model in a view.

Then, refer to the figure displayed on the next slide that shows the output of accessing collection of the User model.

## Using Strong Typing 6-6

- Following figure shows the output of accessing collection of the User model:



Using slide 21, show the result of code that demonstrates the use of User model. The Model sends data to the View and the View displays the same content to the user in the given format.

## Slides 22 to 26

Let us understand HTML helper methods in strongly type views.

### HTML Helper Methods in Strongly Types Views 1-5

- ◆ The MVC Framework:
  - ◆ Enables these helper methods to directly associate with model properties in a strongly types views.
  - ◆ Provides helper methods that you can use only in strongly typed views.
- ◆ Following table lists the helper methods that you can use only in strongly typed views:

Helper Method	Description
<code>Html.LabelFor()</code>	Is the stronglytyped version of the <code>Html.Label()</code> helper method that uses a lambda expression as its parameter, which provides compile time checking.
<code>Html.DisplayNameFor()</code>	Is used to display the names of model properties.
<code>Html.DisplayFor()</code>	Is used to display the values of the model properties.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      22

In slide 22, explain the students that the MVC Framework enables these helper methods to directly associate with model properties in a strongly types views. It also provides helper methods that they can use only in strongly typed views.

Then, refer to table displayed on the slide that lists the helper methods which can be used only in strongly typed views and explain it.

HTML Helper Methods in Strongly Types Views 2-5	
Helper Method	Description
Html.TextBoxFor()	Is the stronglytyped version of the Html.TextBox () helper method.
Html.TextAreaFor()	Is the stronglytyped version of the Html.TextArea () helper method that generates the same markup as that of the Html.TextArea () helper method.
Html.EditorFor ()	Is used to display an editor for the specified model property.
Html.PasswordFor()	Is the stronglytyped version of the Html.Password () helper method.
Html.DropDownListFor ()	Is the stronglytyped version of the Html.DropDownList () helper method that allows selection of a single item.

Using slide 23, explain the remaining helper methods which can be used only in strongly typed views. Refer to the table shown that lists all the methods supported by Html Helper Methods library.

HTML Helper Methods in Strongly Types Views 3-5	
◆ Following code snippet shows the HTML helper methods in a strongly typed view:	<p><b>Code Snippet:</b></p> <pre>@model MVCModelDemo.Models.User @{     ViewBag.Title = "User Form"; } &lt;h2&gt;User Form&lt;/h2&gt; @using (Html.BeginForm()) {     @Html.ValidationSummary(true)     &lt;div&gt;         @Html.LabelFor(model =&gt; model.name)     &lt;/div&gt;     &lt;div&gt;         @Html.EditorFor(model =&gt; model.name)     &lt;/div&gt;     &lt;div&gt;         @Html.LabelFor(model =&gt; model.address)     &lt;/div&gt;</pre>
<p>© Aptech Ltd. Models in ASP.NET MVC/Session 4 24</p>	

In slide 24, refer to the code that shows the HTML helper methods in a strongly typed view.

## HTML Helper Methods in Strongly Types Views 4-5

Code Snippet:

```
@Html.EditorFor(model =>model.address)
<div>
@Html.LabelFor(model =>model.email)
</div>
<div>
@Html.EditorFor(model =>model.email)
</div>
<p>
<input type="submit" value="Create" />
</p> }
```

- ◆ In this code:
  - + The `Html.LabelFor()` method is used to display labels based on the property names of the model.
  - + The `Html.EditorFor()` method is used to display editable fields for the properties of the model.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 25

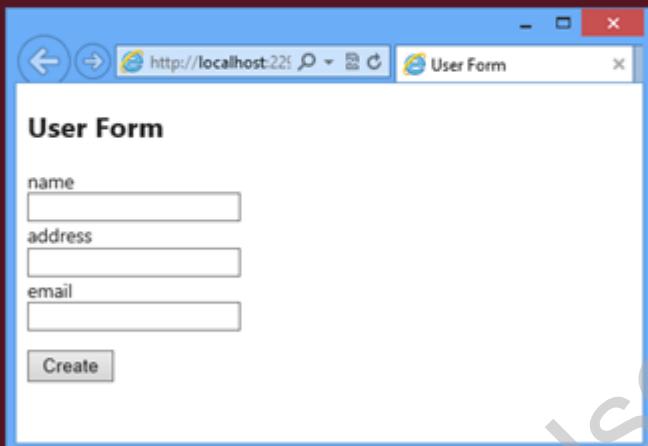
In slide 25, refer to the code displayed on the slide.

Then, explain them that in this code the `Html.LabelFor()` method is used to display labels based on the property names of the model. The `Html.EditorFor()` method is used to display editable fields for the properties of the model.

Next, refer to the figure displayed on the next slide that shows the output of using HTML helper methods.

## HTML Helper Methods in Strongly Types Views 5-5

- Following figure shows the output of using HTML helper methods:



© Aptech Ltd.

Models in ASP.NET MVC/Session 4

26

Using slide 26, show the output of the code given on slide 25. It shows the actual control that will be displayed when the code is executed.

## Slides 27 and 28

Let us understand role of model binder.

The slide has a dark red header bar with the title "Role of Model Binder 1-2" in white. The main content area is white with a dark red sidebar on the left. A large watermark "For Private Centre Use Only" is diagonally across the slide. The footer is dark red with white text: "© Aptech Ltd.", "Models in ASP.NET MVC/Session 4", and "27".

**Role of Model Binder 1-2**

- ◆ When a user submits information in a form within a strongly typed view, ASP.NET MVC automatically examines the `HttpRequest` object and maps the information sent to fields in the model object.
- ◆ The process of mapping the information in the `HttpRequest` object to the model object is known as model binding.
- ◆ Some of the advantages of model binding are as follows:
  - + Automatically extract the data from the `HttpRequest` object.
  - + Automatically converts data type.
  - + Makes data validation easy.

In slide 27, explain the students that when a user submits information in a form within a strongly typed view, ASP.NET MVC automatically examines the `HttpRequest` object and maps the information sent to fields in the model object. The process of mapping the information in the `HttpRequest` object to the model object is known as model binding.

Then, explain the following advantages of model binding:

- Automatically extracts the data from the `HttpRequest` object.
- Automatically converts data type.
- Makes data validation easy.

### Role of Model Binder 2-2

- ◆ The MVC Framework provides a model binder that performs model binding in application.
- ◆ The `DefaultModelBinder` class implements the model binder on the MVC Framework.
- ◆ The two most important roles of the model binder are as follows:
  - ◆ Bind request to primitive values
  - ◆ Bind request to objects

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      28

Use slide 28 to explain the students that the MVC Framework provides a model binder that performs model binding in application. The `DefaultModelBinder` class implements the model binder on the MVC Framework.

Then, explain the following two most important roles of the model binder:

- Bind request to primitive values
- Bind request to objects

## Slides 29 to 32

Let us understand binding to primitive values.

### Binding to Primitive Values 1-4

- ◆ To understand how the model binder binds request to primitive values, consider a scenario where you are creating a log in form that accepts log in details from user.
- ◆ For this, first you need to create a Login model in your application.
- ◆ Following code snippet shows the Login model class:

**Code Snippet:**

```
public class Login
{
    public string userName { get; set; }

    [DataType(DataType.Password)]
    public string password { get; set; }
}
```

- ◆ This code creates two properties named `userName` and `password` in the Login model.
- ◆ After creating the model class, you need to create an `Index.cshtml` view to display the login form.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 29

In slide 29, tell the students that to understand how the model binder binds request to primitive values, consider a scenario where they are creating a log in form that accepts log in details from user. For this, first they need to create a Login model in the application.

Then, refer to the code that shows the Login model class.

Explain them that the referred code creates two properties named `userName` and `password` in the Login model.

Tell them that after creating the model class, they need to create an `Index.cshtml` view to display the login form.

## Binding to Primitive Values 2-4

- Following code shows the content of the Index.cshtml file:

**Code Snippet:**

```
@model ModelDemo.Models.Login
@{ ViewBag.Title = "Index";
}
<h2>User Details</h2>
@using (Html.BeginForm()) {
@Html.ValidationSummary(true)
<div>
@Html.LabelFor(model =>model.userName)
</div> <div>
@Html.EditorFor(model =>model.userName)
</div> <div>
@Html.LabelFor(model =>model.password)
</div> <div>
@Html.EditorFor(model =>model.password) </div>
</div>
<input type="submit" value="Submit" />
</div> }
```

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      30

In slide 30, refer to the code displayed on the slide that shows the content of the Index.cshtml file.

## Binding to Primitive Values 3-4

- Once, you have created the view, you need to create a controller class that contains the Index() action method to display the view.
- Following code shows the HomeController controller:

**Code Snippet:**

```
public class HomeController : Controller {
    public ActionResult Index() {
        return View();
    }
    [HttpPost]
    public ActionResult Index(string userName, string password) {
        if (userName == "Peter" && password == "pass@123") {
            stringmsg = "Welcome " + userName;
            return Content(msg);
        }
        else {
            return View();
        }
    }
}
```

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      31

In slide 31, tell the students that once they have created the view, they need to create a controller class that contains the Index( ) action method to display the view.

Next, refer to the code displayed on the slide that shows the HomeController controller.

## Binding to Primitive Values 4-4

- ◆ In the preceding code:
  - ◆ The first `Index()` method returns the `Index.cshtml` view that displays a login form.
  - ◆ The second `Index()` method is marked with the `HttpPost` attribute. This method accepts two primitive as parameters. The `Index()` method compares the parameters with predefined values and returns a message if the comparison returns true. Else, the `Index()` method returns back the `Index.cshtml` view.
  - ◆ When a user submits the login data, the default model binder maps the values of the `userName` and `password` fields to the primitive type parameters of the `Index()` action method.
  - ◆ In the `Index()` action method, you can perform the required authentication and return a result.

In slide 32, refer to the code displayed on slide 31.

Next, explain them that in the code, the first `Index()` method returns the `Index.cshtml` view that displays a login form. The second `Index()` method is marked with the `HttpPost` attribute. This method accepts two primitive as parameters. The `Index()` method compares the parameters with predefined values and returns a message if the comparison returns true. Else, the `Index()` method returns back the `Index.cshtml` view.

Then, tell them that when a user submits the login data, the default model binder maps the values of the `userName` and `password` fields to the primitive type parameters of the `Index()` action method. In the `Index()` action method, they can perform the required authentication and return a result.

## Slides 33 to 37

Let us understand binding to objects.

**Binding to Object 1-5**

- ◆ To understand how the model binder binds requests to objects, consider the same scenario where you are creating a login form.
- ◆ For this, you have already created the Login model and the Index.cshtml view.
- ◆ To bind request to object, you need to update the controller class so that it accepts a Login object as a parameter, instead of an HttpRequest object.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 33

Using slide 33, tell the students that to understand how the model binder binds requests to objects, consider the same scenario where they were creating a login form. For this, they have already created the Login model and the Index.cshtml view.

Now, to bind request to object, they need to update the controller class, so that it accepts a Login object as a parameter, instead of an HttpRequest object.

**Binding to Object 2-5**

- ◆ Following code shows the updated controller class:

**Code Snippet:**

```
public class HomeController : Controller {
    public ActionResult Index() {
        return View();
    }

    [HttpPost]
    public ActionResult Index(Login login) {
        if (login.userName == "Peter" && login.password == "pass@123") {
            stringmsg = "Welcome " + login.userName;
            return Content(msg);
        }
        else {
            return View();
        }
    }
}
```

- ◆ In this code:
  - + The first Index() method returns the Index.cshtml view that displays a login form.
  - + The second Index() method automatically removes the data from the HttpRequest object and put into the Login object.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 34

Using slide 34, refer to the code that shows the updated controller class. Explain the students that in this code, the first Index() method returns the Index.cshtml view that displays a login form. The second Index() method automatically removes the data from the HttpRequest object and put into the Login object.

**Binding to Object 3-5**

- ◆ When a user submits the login data, the `Index()` method validates the `username` and `password` passed in the `Login` object.
- ◆ When the validation is successful, the view displays a welcome message.
- ◆ When you access the application from the browser the `Index.cshtml` view displays the login form.

1. Type Peter in the **User Name** text field and pass@123 in the **Password** field of the login form.

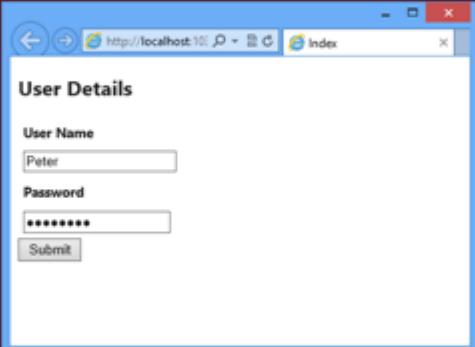
© Aptech Ltd.      Models in ASP.NET MVC/Session 4      35

In slide 35, explain the students that when a user submits the login data, the `Index()` method validates the username and password passed in the `Login` object. When the validation is successful, the view displays a welcome message.

Tell them that when they access the application from the browser, the `Index.cshtml` view displays the login form.

**Binding to Object 4-5**

♦ Following figure shows the login form with data specified in the **User Name** and **Password** fields:



2. Click **Submit**. The login form displays a 'Welcome Peter' message.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 36

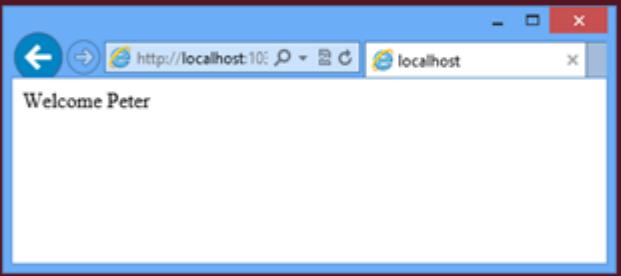
In slide 36, refer to the figure displayed on the slide that shows the login form with data specified in the User Name and Password fields.

Then, tell them that clicking the **Submit** button, the login form displays a 'Welcome Peter' message.

Refer to the figure displayed on the next slide that shows the output of the Index.cshtml view after submitting the user name and password.

**Binding to Object 5-5**

◆ Following figure shows the output of the `Index.cshtml` view after submitting the user name and password:



The screenshot shows a Microsoft Edge browser window. The address bar displays "http://localhost:1024" and the title bar says "localhost". The main content area of the browser contains the text "Welcome Peter". At the bottom of the slide, there is a watermark reading "For Aptech Centre Use Only".

Using slide 37, show the output after submitting the user name and password. The user name and password is accepted and sent as request to the server. In return, the server displays the message 'Welcome <user name>'. In the current example, the message shown is 'Welcome Peter'.

## Slide 38

Let us understand Visual Studio .NET scaffolding.

The slide has a dark red header with the title 'Visual Studio.NET Scaffolding'. The main content area contains a bulleted list describing the ASP.NET MVC Framework's scaffolding feature. It includes points about convention-based naming, code storage, and five template types: List, Create, Edit, Details, and Delete. It also mentions ViewData usage. The footer contains copyright information for Aptech Ltd., the slide title 'Models in ASP.NET MVC/Session 4', and the slide number '38'.

**Visual Studio.NET Scaffolding**

- ◆ The ASP.NET MVC Framework provides a feature called scaffolding that allows you to generate views automatically.
- ◆ By convention, scaffolding uses specific name for views.
- ◆ After creating a view it stores the auto-generated code in respective places for the application to work.
- ◆ Following are the five types of template that the scaffolding feature provides to create views:
  - ◆ **List:** Generates markup to display the list of model objects.
  - ◆ **Create:** Generates markup to add a new object to the list.
  - ◆ **Edit:** Generates markup to edit an existing model object.
  - ◆ **Details:** Generates markup to show the information of an existing model object.
  - ◆ **Delete:** Generates markup to delete an existing model object.
- ◆ In this code, `ViewData` is used to display the values of the `Message` and `CurrentTime` keys.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 38

In slide 38, explain to the students that the ASP.NET MVC Framework provides a feature called **scaffolding** that allows them to generate views automatically. By convention, scaffolding uses specific name for views. After creating a view, it stores the auto-generated code in respective places for the application to work.

Next, explain the following five types of template that the scaffolding feature provides to create views:

- **List:** Generates markup to display the list of model objects.
- **Create:** Generates markup to add a new object to the list.
- **Edit:** Generates markup to edit an existing model object.
- **Details:** Generates markup to show the information of an existing model object.
- **Delete:** Generates markup to delete an existing model object.

### Discussion:

Initiate a discussion by explaining the students that creating controllers manually and views to handle the information related to a product and for displaying the same to the user requires a lot of time. To overcome this problem, ASP.NET provides a feature called scaffolding that builds an application quickly.

## Slides 39 to 42

Let us understand the List template.

**List Template 1-4**

- ◆ ListTemplate allows you to create a view that displays a list of model objects.
- ◆ Following code shows an Index() action method that returns an ActionResult object through a call to the View() method of the controller class:

**Code Snippet:**

```
public ActionResult Index()
{
    var user = new List<User>();
    //Code to populate the user collection
    return View(user);
}
```

- ◆ This code shows the Index() action method of a controller that returns the result of a call to the View() method.
- ◆ The result of the View() method is an ActionResult object that renders a view.
- ◆ In this code, ViewData is used to display the values of the Message and CurrentTime keys.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      39

Using slide 39, explain to the students that the List template allows them to create a view that displays a list of model objects.

Then, refer to the code that shows an Index() action method that returns an ActionResult object through a call to the View() method of the controller class.

Explain them that this code shows the Index() action method of a controller that returns the result of a call to the View() method. The result of the View() method is an ActionResult object that renders a view.

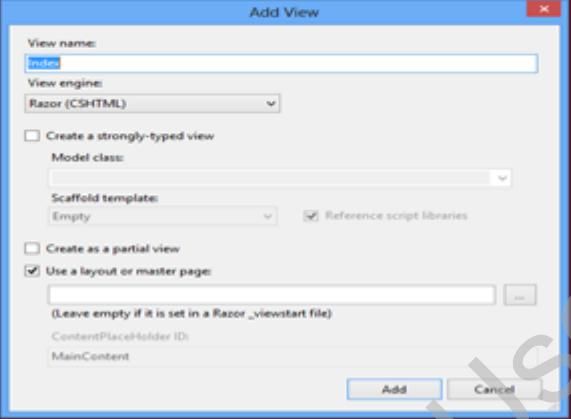
### Note:

Scaffolding templates give the basic code required to implement the create, read, update, and delete (CRUD) functionalities in an application. However, they may need to customize the code/markup as per the specific requirements.

Using scaffolding is not necessary. However, it can speed up the development process. If you do not like the scaffolding behavior, you can write the code/markup yourself from scratch.

### List Template 2-4

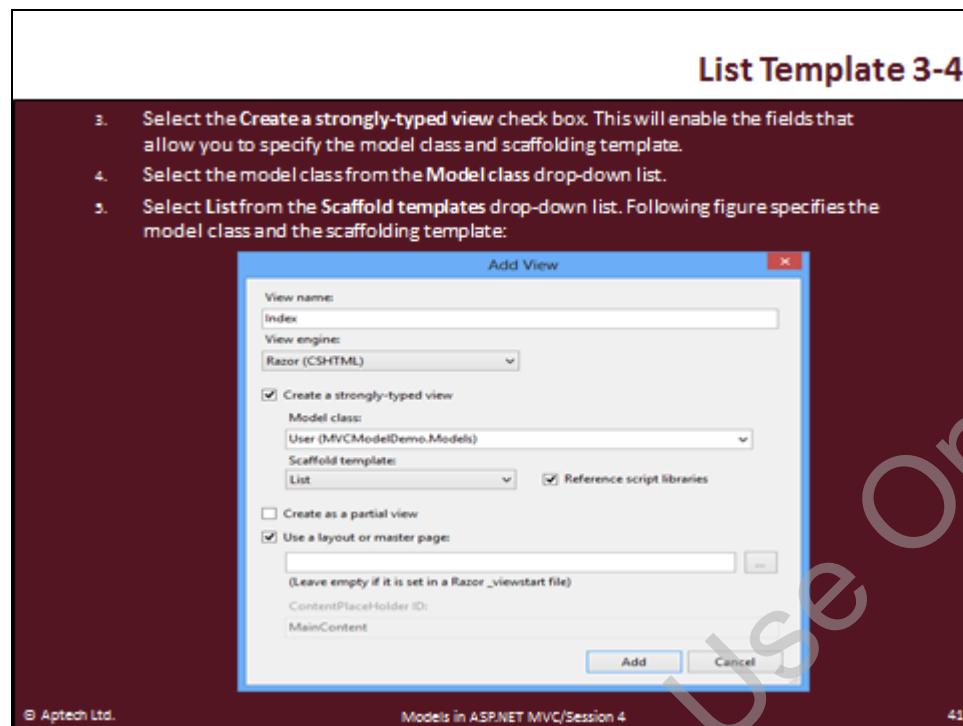
- ◆ To create a view using the List template, you need to perform the following steps:
  1. Right-click inside the action method for which you need to create a view.
  2. Select Add View from the context menu that appears. The Add View dialog box is displayed, as shown in the following figure:



© Aptech Ltd. Models in ASP.NET MVC/Session 4 40

Use slide 40 to explain the steps to create a view using the List template.

Refer to the figure displayed on the slide that shows the **Add View** dialog box.



Use slide 41 to explain the steps to create a view using the List template.

Refer to the figure displayed on the slide that shows how to specify the model class and the scaffolding template.

Then, refer to the figure displayed on the next slide that shows the output of the auto-generated markup.

**List Template 4-4**

6. Click Add. Visual Studio.NET automatically creates the appropriate directory structure and adds the view file to it.

- ◆ Following figure shows the output of the auto-generated markup:

The screenshot shows a web browser window with the URL <http://localhost:225> in the address bar. The page title is "Index". Below the title, there is a link "Create New". A table is displayed with three columns: "name", "address", and "email". The first row contains "Mark Smith", "Park Street", and "Mark@mvcexample.com" with links "Edit | Details | Delete". The second row contains "John Parker", "New Park", and "John@mvceexample.com" with links "Edit | Details | Delete". The third row contains "Steave Edward Melbourn Street" and "steave@mvceexample.com" with links "Edit | Details | Delete".

name	address	email
Mark Smith	Park Street	Mark@mvcexample.com
John Parker	New Park	John@mvceexample.com
Steave Edward Melbourn Street		steave@mvceexample.com

For Aptech Centre Use Only

Using slide 42, tell the students that the when the View is created the directory structure to store the same is automatically created by Visual Studio. Show them that how the auto generated mark-up shows the list of the directory structure which allows edit, details, and delete options also.

## Slides 43 to 48

Let us understand the Create template.

### Create Template 1-6

- ◆ The Create template allows you to generate a view that accepts the details of a new object to be stored in a data store.
- ◆ You need to create an action method, to display a view based on the Create template.
- ◆ Following code shows creating an action method named, `Create()` in the `HomeController` controller:

**Code Snippet:**

```
public ActionResult Create()  
{  
    return View();  
}
```

- ◆ This code creates the `Create()` action method that will invoke when a user clicks the Create link on the view generated using the List template.
- ◆ In this code, `ViewData` is used to display the values of the `Message` and `CurrentTime` keys.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      43

In slide 43, explain to the students that the Create template allows the students to generate a view that accepts the details of a new object to be stored in a data store. They need to create an action method, to display a view based on the Create template.

Then, refer to the code that shows how to create an action method named, `Create()` in the `HomeController` controller.

Then, explain them that this code creates the `Create()` action method that will invoke when a user clicks the Create link on the view generated using the List template.

### Create Template 2-6

- ◆ To create a view using the Create template, you need to perform the following steps:
  1. Right-click inside the `Create ()` action method.
  2. Select **Add View** from the context menu that appears. The **Add View** dialog box appears.
  3. Select the **Create a strongly-typed view** checkbox.
  4. Select the model class from the **Model class** drop-down list.
  5. Select **Create from the Scaffold templates** drop-down list.
  6. Click **Add**. Visual Studio.NET automatically creates a view named, Create in the appropriate directory structure.

Using slide 44, explain the steps to create a view using the Create template.

### Create Template 3-6

- Following code snippet shows the auto-generated markup using the Create template:

#### Code Snippet:

```
@model MVCModelDemo.Models.User
{
    ViewBag.Title = "Create";
}
<h2>Create</h2>
@using (Html.BeginForm()) {
    @Html.ValidationSummary(true)
    <fieldset>
        <legend>User</legend>
        <div class="editor-label">
            @Html.LabelFor(model => model.name)
        </div>
        <div class="editor-field">
            @Html.EditorFor(model => model.name)
            @Html.ValidationMessageFor(model => model.name)
        </div>
        <div class="editor-label">
            @Html.LabelFor(model => model.address)
        </div>
        <div class="editor-field">
```

In slide 45, explain the code that shows the auto-generated markup using the Create template.

### Create Template 4-6

**Code Snippet:**

```
@Html.EditorFor(model =>model.address)
@Html.ValidationMessageFor(model =>model.address)
</div>
<div class="editor-label">
@Html.LabelFor(model =>model.email)
</div>
<div class="editor-field">
@Html.EditorFor(model =>model.email)
@Html.ValidationMessageFor(model =>model.email)
</div>
<p>
<input type="submit" value="Create" />
</p>
</fieldset>
}
<div>
@Html.ActionLink("Back to List", "Index")
</div>
@section Scripts {
@Scripts.Render("~/bundles/jqueryval")
}
```

© Aptech Ltd. Models in ASP.NET MVC/Session 4 46

Using slide 46, show the continuation of previous slide code. It can be seen that the auto-generated code is completed now.

### Create Template 5-6

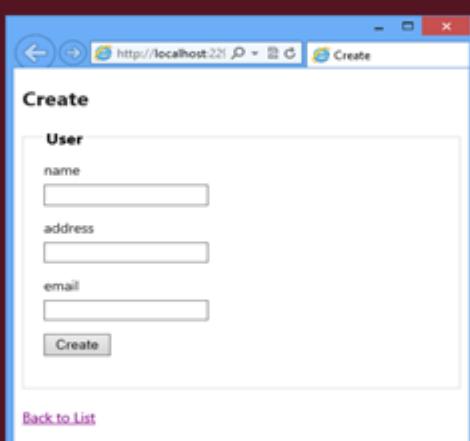
- ◆ In this code,
  - ◆ The `BeginForm` helper method starts a form.
  - ◆ The `Html.ValidationSummary()` validation helper displays the summary of all the error messages at one place.
  - ◆ The `Html.LabelFor()` helper method displays an HTML label element with the name of the property.
  - ◆ The `Html.EditorFor()` helper method displays a textbox that accepts the value of a model property.
  - ◆ The `Html.ValidationMessageFor()` validation helper method displays a validation error message for the associated model property.

In slide 47, explain the code displayed on slides 45 and 46, as follows:

- The `BeginForm` helper method starts a form.
- The `Html.ValidationSummary()` validation helper displays the summary of all the error messages at one place.
- The `Html.LabelFor()` helper method displays an HTML label element with the name of the property.
- The `Html.EditorFor()` helper method displays a textbox that accepts the value of a model property.
- The `Html.ValidationMessageFor()` validation helper method displays a validation error message for the associated model property.

**Create Template 6-6**

- ◆ Following figure shows the output of auto-generated markup using the Create template:



The screenshot shows a browser window with the URL `http://localhost:221`. The page title is "Create". The content is a "Create" form for a "User" model. It contains three text input fields: "name", "address", and "email", each with a label above it. Below these fields is a "Create" button. At the bottom of the form is a "Back to List" link.

- ◆ When the user enters data in the form and clicks the Create button, an HTTP POST request is sent to the `Create()` action method of the controller.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      48

In slide 48, refer to the figure displayed on the slide that shows the output of auto-generated markup using the Create template.

Tell them that when the user enters data in the form and clicks the Create button, an HTTP POST request is sent to the `Create()` action method of the controller.

## Slides 49 to 52

Let us understand the Edit template.

### Edit Template 1-4

- ♦ The Edit template can be used whenever you want to generate a view that required to be used for modifying the details of an existing object stored in a data store.
- ♦ To display a view based on the Edit template, you need to create an action method to pass the model object to be edited to the view.
- ♦ Following code shows the action method named `Edit()`:

**Code Snippet:**

```
public ActionResult Edit()
{
    return View();
}
```

- ♦ Once you have created the `Edit()` action method in the controller, you can use Visual Studio .NET to create the view using the Edit template. To create the view using the Edit template in Visual Studio .NET, you need to select **Edit** from the **Scaffold templates** drop-down list.
- ♦ After creating a view named, `Edit` for the `User` model using the Edit template, Visual Studio .NET generates the markup for the view.
- ♦ In this code, `ViewData` is used to display the values of the `Message` and `CurrentTime` keys.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 49

In slide 49, explain to the students that the Edit template can be used whenever they want to generate a view that required to be used for modifying the details of an existing object stored in a data store. To display a view based on the Edit template, they need to create an action method to pass the model object to be edited to the view.

Then, refer to the code that shows the action method named `Edit()`.

Next, tell them that once they have created the `Edit()` action method in the controller, they can use Visual Studio .NET to create the view using the Edit template. To create the view using the Edit template in Visual Studio .NET, they need to select **Edit** from the **Scaffold templates** drop-down list.

Then, tell them that after creating a view named, `Edit` for the `User` model using the Edit template, Visual Studio .NET generates the markup for the view.

## Edit Template 2-4

- Following figure shows the output of creating the view using the Edit template:

### Code Snippet:

```
@model MVCModelDemo.Models.User
@{
    ViewBag.Title = "Edit";
}
<h2>Edit</h2>
@using (Html.BeginForm()) {
    @Html.ValidationSummary(true)
    <fieldset>
        <legend>User</legend>
        @Html.HiddenFor(model => model.Id)
        <div class="editor-label">
            @Html.LabelFor(model => model.name)
        </div>
        <div class="editor-field">
            @Html.EditorFor(model => model.name)
            @Html.ValidationMessageFor(model => model.name)
        </div>
        <div class="editor-label">
            @Html.LabelFor(model => model.address)
        </div>
}
```

In slide 50, refer to the code that shows how to create the view using the Edit template.

## Edit Template 3-4

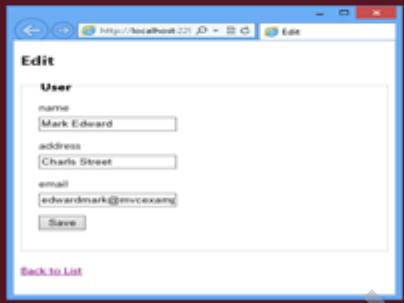
### Code Snippet:

```
<div class="editor-field">
    @Html.EditorFor(model => model.address)
    @Html.ValidationMessageFor(model => model.address)
</div>
<div class="editor-label">
    @Html.LabelFor(model => model.email)
</div>
<div class="editor-field">
    @Html.EditorFor(model => model.email)
    @Html.ValidationMessageFor(model => model.email)
</div>
<p>
    <input type="submit" value="Save" />
</p>
</fieldset>
}
<div>
    @Html.ActionLink("Back to List", "Index")
</div>
@section Scripts {
    @Scripts.Render("~/bundles/jqueryval")
}
```

Using slide 51, show the code that is continuation of the previous slide.

### Edit Template 4-4

- ◆ In this code,
  - ◆ The `Html.LabelFor()` helper method displays an HTML label element with the name of the property.
  - ◆ The `Html.EditorFor()` helper method displays a textbox to accept the value of a model property.
  - ◆ The `Html.ValidationMessageFor()` helper method displays a validation error message.
- ◆ Following figure shows the output of creating the view using the Edit template:



The screenshot shows a Windows application window titled "Edit". Inside, there's a form titled "User" with three text input fields: "name" containing "Mark Edward", "address" containing "Chats Street", and "email" containing "edwardmark@mvcexam...". Below the inputs is a "Save" button. At the bottom of the window is a link "Back to List".

In slide 52, refer to the code displayed on slides 50 and 51 and explain that in this code:

- The `Html.LabelFor()` helper method displays an HTML label element with the name of the property.
- The `Html.EditorFor()` helper method displays a textbox to accept the value of a model property.
- The `Html.ValidationMessageFor()` helper method displays a validation error message.

Then, refer to the figure displayed on the slide that shows the output of creating the view using the Edit template.

## Slides 53 to 55

Let us understand the Details template.

### Details Template 1-3

- ♦ The Details template allows you to create a view that displays details of the User model.
- ♦ Once you have created the Details() action method in the controller, and a view using the Details template in Visual Studio .NET, it generates the markup for the view.
- ♦ Following code snippet shows the auto-generated markup when you create the view using the Details template:

Code Snippet:

```
@model MVCModelDemo.Models.User
@{
    ViewBag.Title = "Details";
}
<h2>Details</h2>
<fieldset>
<legend>User</legend>
<div class="display-label">
    @Html.DisplayNameFor(model => model.name)
</div>
<div class="display-field">
    @Html.DisplayFor(model => model.name)
</div>
```

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      53

In slide 53, explain to the students that the Details template allows you to create a view that displays details of the User model. Once they have created the Details() action method in the controller, and a view using the Details template in Visual Studio .NET, it generates the markup for the view.

Refer to the code on the slide that shows the auto-generated markup when they create the view using the Details template.

### Details Template 2-3

**Code Snippet:**

```
<div class="display-label">
    @Html.DisplayNameFor(model =>model.address)
</div>
<div class="display-field">
    @Html.DisplayFor(model =>model.address)
</div>
<div class="display-label">
    @Html.DisplayNameFor(model =>model.email)
</div>
<div class="display-field">
    @Html.DisplayFor(model =>model.email)
</div>
</fieldset>
<p>
    @Html.ActionLink("Edit", "Edit", new { id=Model.Id }) |
    @Html.ActionLink("Back to List", "Index")
</p>
```

♦ In this code, the `Html.DisplayNameFor()` helper method displays the name of model properties and the `Html.DisplayFor()` helper method displays the values of the model properties.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 54

In slide 54, refer to the code that the auto-generated markup when you create the view using the Details template and explain the students that in this code, the `Html.DisplayNameFor()` helper method displays the name of model properties and the `Html.DisplayFor()` helper method displays the values of the model properties.

Refer to the figure displayed on the next slide that shows the output of auto-generated markup for Details template.

**Details Template 3-3**

◆ Following figure shows the output of auto-generated markup for Details template:

The screenshot shows a browser window with the URL <http://localhost:8221/> in the address bar. The page title is "Details". The main content area displays a "User" model with the following properties:

- name: Mark Smith
- address: Park Street
- email: Mark@mvcexample.com

At the bottom of the content area, there are two links: "Edit" and "Back to List".

Using slide 55, show the output of the auto generated markup. Explain to the students how the result is generated.

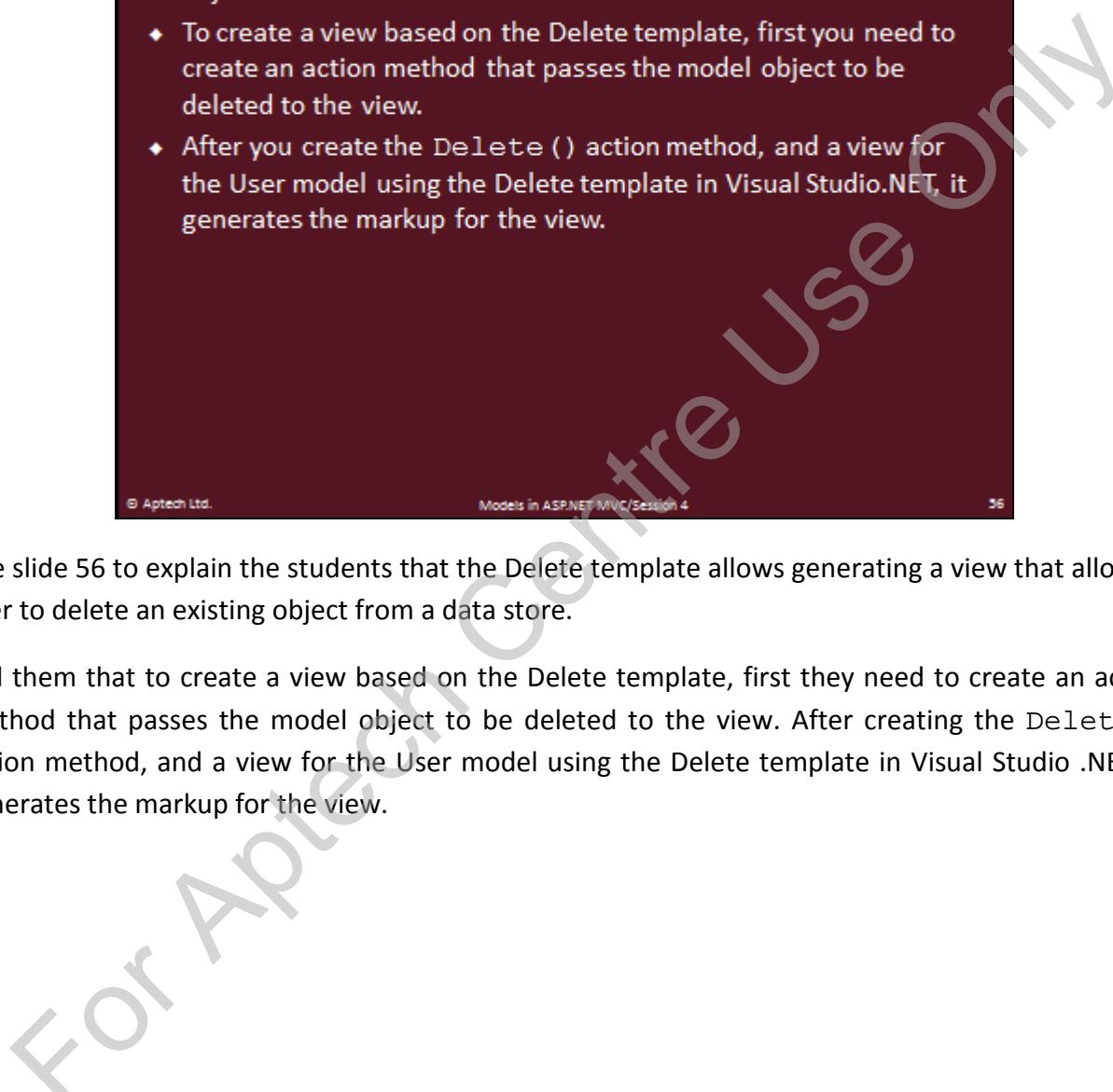
## Slides 56 to 59

Let us understand the Delete template.

**Delete Template 1-4**

- ◆ Allows generating a view that allows a user to delete an existing object from a data store.
- ◆ To create a view based on the Delete template, first you need to create an action method that passes the model object to be deleted to the view.
- ◆ After you create the `Delete()` action method, and a view for the User model using the Delete template in Visual Studio .NET, it generates the markup for the view.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      56



Use slide 56 to explain the students that the Delete template allows generating a view that allows a user to delete an existing object from a data store.

Tell them that to create a view based on the Delete template, first they need to create an action method that passes the model object to be deleted to the view. After creating the `Delete()` action method, and a view for the User model using the Delete template in Visual Studio .NET, it generates the markup for the view.

### Delete Template 2-4

♦ Following code snippet shows the auto-generated markup when you create a view using the Delete template:

**Code Snippet:**

```
@model MVCModelDemo.Models.User
@{
    ViewBag.Title = "Delete";
}
<h2>Delete</h2>
<h3>Are you sure you want to delete this?</h3>
<fieldset>
<legend>User</legend>
<div class="display-label">
    @Html.DisplayNameFor(model => model.name)
</div>
<div class="display-field">
    @Html.DisplayFor(model => model.name)
</div>
<div class="display-label">
    @Html.DisplayNameFor(model => model.address)
</div>
<div class="display-field">
    @Html.DisplayFor(model => model.address)
</div>
```

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      37

Using slide 57, refer to the code that shows the auto-generated markup when they create a view using the Delete template.

### Delete Template 3-4

**Code Snippet:**

```
<div class="display-label">
@Html.DisplayNameFor(model =>model.email)
</div>
<div class="display-field">
@Html.DisplayFor(model =>model.email)
</div>
</fieldset>
@using (Html.BeginForm()) {
<p>
<input type="submit" value="Delete" /> |
@Html.ActionLink("Back to List", "Index")
</p>
}
```

- ◆ In this code, the `Html.DisplayNameFor()` helper method displays the names of model properties and the `Html.DisplayFor()` helper method displays the values of the model properties.
- ◆ In addition, the `Html.Actionlink()` helper method is used to create a link to list the details for a product.

© Aptech Ltd.      Models in ASP.NET MVC/Session 4      58

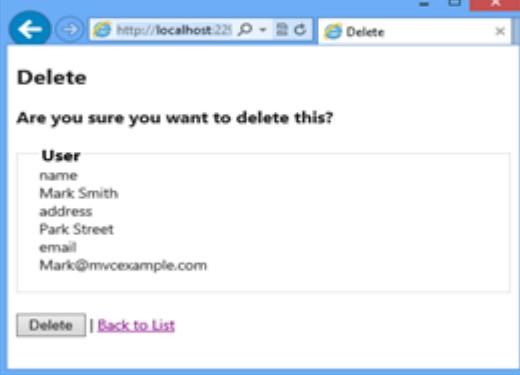
In slide 58, refer to the continuation of code that shows the auto-generated markup when the students create a view using the Delete template.

Explain them that in this code, the `Html.DisplayNameFor()` helper method displays the names of model properties and the `Html.DisplayFor()` helper method displays the values of the model properties. In addition, the `Html.Actionlink()` helper method is used to create a link to list the details for a product

Then, refer to the figure displayed on slide 59 that shows the output of creating view using the Delete template.

**Delete Template 4-4**

♦ Following figure shows the output of creating view using the Delete template:

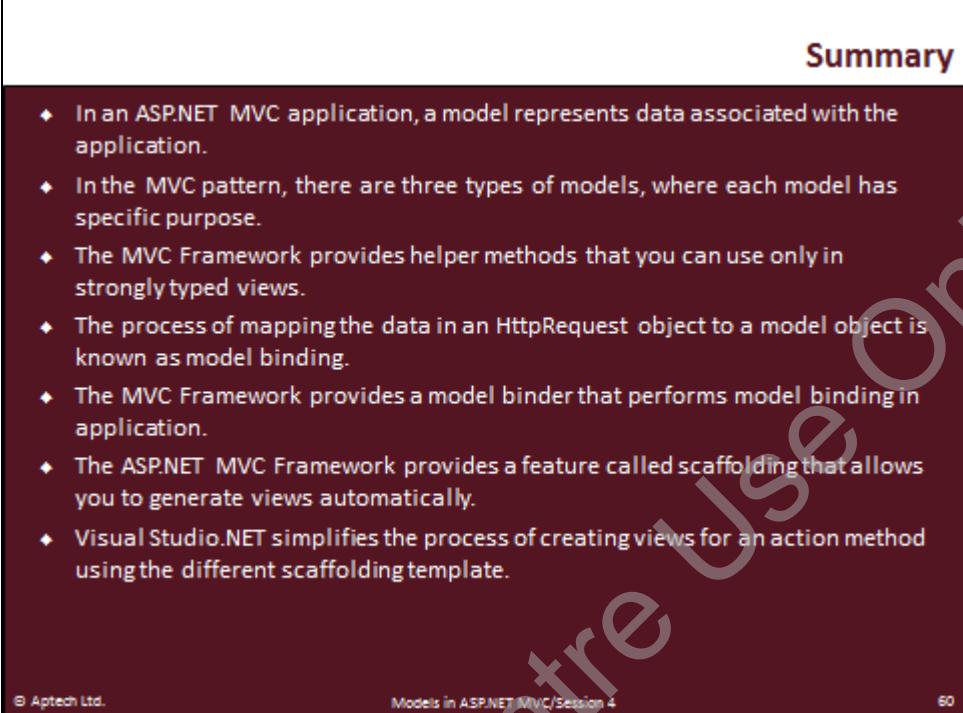


© Aptech Ltd. Models in ASP.NET MVC/Session 4 59

Using slide 59, show the output of creating view using the Delete template. When the 'Delete' template is clicked, it shows alert message 'Are you sure you want to delete this?' and waits for the response from the user.

## Slide 60

Let us summarize the session.



The slide has a dark blue header with the word "Summary" in white. The main content area is white with a dark blue border. It contains a bulleted list of nine points about ASP.NET MVC models. At the bottom, there is a footer bar with three items: "© Aptech Ltd.", "Models in ASP.NET MVC/Session 4", and "60". A large watermark reading "Aptech Centre for Reuse Only" is diagonally across the slide.

**Summary**

- ◆ In an ASP.NET MVC application, a model represents data associated with the application.
- ◆ In the MVC pattern, there are three types of models, where each model has specific purpose.
- ◆ The MVC Framework provides helper methods that you can use only in strongly typed views.
- ◆ The process of mapping the data in an HttpRequest object to a model object is known as model binding.
- ◆ The MVC Framework provides a model binder that performs model binding in application.
- ◆ The ASP.NET MVC Framework provides a feature called scaffolding that allows you to generate views automatically.
- ◆ Visual Studio.NET simplifies the process of creating views for an action method using the different scaffolding template.

© Aptech Ltd. Models in ASP.NET MVC/Session 4 60

In slide 60, summarize the session. End the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- In an ASP.NET MVC application, a model represents data associated with the application.
- In the MVC pattern, there are three types of models, where each model has specific purpose.
- The MVC Framework provides helper methods that you can use only in strongly typed views.
- The process of mapping the data in an HttpRequest object to a model object is known as model binding.
- The MVC Framework provides a model binder that performs model binding in application.
- The ASP.NET MVC Framework provides a feature called scaffolding that allows you to generate views automatically.
- Visual Studio.NET simplifies the process of creating views for an action method using the different scaffolding template.

### **4.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. How will you make the model data accessible to a view once you have accessed the model within a controller?
2. How will you ignore explicit type casting?
3. What option will you use to pass model data from a controller to a view?

## Session 5 -

### Data Validation and Annotation

#### **5.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

##### **5.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe how to validate data
- Explain how to use data annotation
- Explain and describe how to use ModelState

##### **5.1.2 Teaching Skills**

To teach this session successfully, you must know about data validation in an ASP.NET MVC. You should also be aware of how to use data annotation and ModelState in an ASP.NET MVC application.

You should teach the concepts in the theory class using slides and LCD projectors.

##### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

##### **In-Class Activities**

Follow the order given here during In-Class activities.

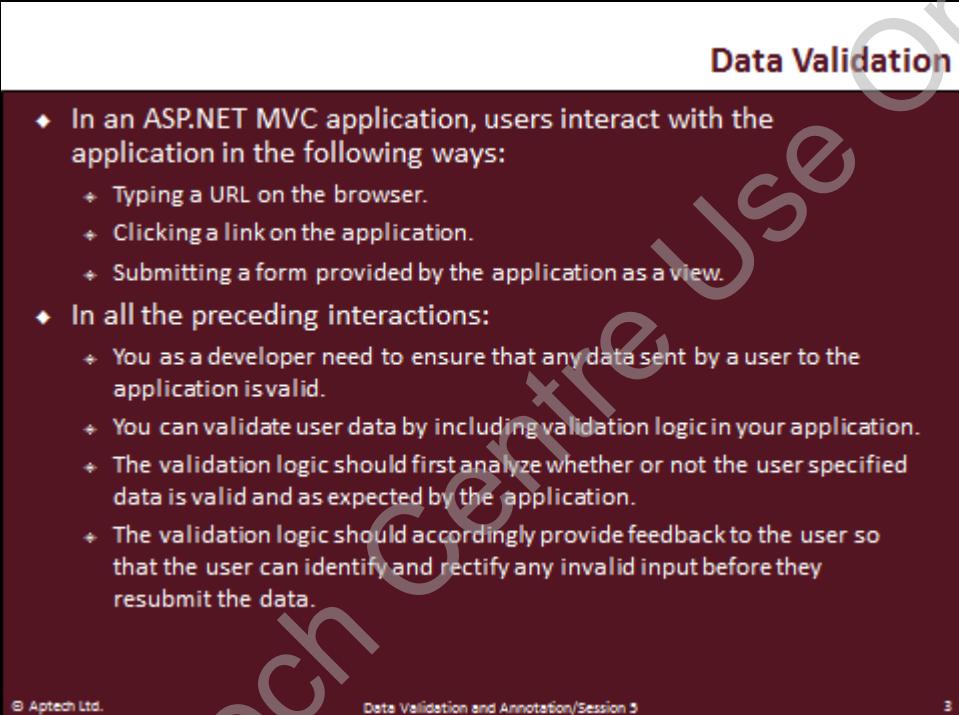
##### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the data validation in an ASP.NET MVC application. They will learn about how to use data annotations in an application. This session will also discuss about how to use ModelState.

## **5.2 In-Class Explanation**

### **Slide 3**

Let us understand data validation.



The slide has a dark red header bar with the title "Data Validation" in white. The main content area is white with black text. It contains two bullet points under a single heading:

- ◆ In an ASP.NET MVC application, users interact with the application in the following ways:
  - ◆ Typing a URL on the browser.
  - ◆ Clicking a link on the application.
  - ◆ Submitting a form provided by the application as a view.
- ◆ In all the preceding interactions:
  - ◆ You as a developer need to ensure that any data sent by a user to the application is valid.
  - ◆ You can validate user data by including validation logic in your application.
  - ◆ The validation logic should first analyze whether or not the user specified data is valid and as expected by the application.
  - ◆ The validation logic should accordingly provide feedback to the user so that the user can identify and rectify any invalid input before they resubmit the data.

At the bottom, there is a footer bar with the Aptech logo, the text "Data Validation and Annotation/Session 5", and a small number "3".

Use slide 3 to explain that in an ASP.NET MVC application, users interact with the application in three ways, such as typing a URL on the browser, clicking a link on the application, and submitting a form provided by the application as a view.

Tell them that in all the preceding interactions:

- They as a developer need to ensure that any data sent by a user to the application is valid.
- They can validate user data by including validation logic in your application.
- The validation logic should first analyze whether or not the user specified data is valid and as expected by the application.
- The validation logic should accordingly provide feedback to the user, so that the user can identify and rectify any invalid input before they resubmit the data.

## Slide 4

Let us understand data validation workflow.

### Data Validation Workflow

- ◆ The MVC Framework implements a validation workflow to validate user data.
- ◆ The validation workflow starts when a user specified data arrives in the server.
- ◆ The validation process:
  - ◆ First checks whether the request is some form of attack, such as Cross Site Scripting (CSS).
  - ◆ If the process identifies a request as an attack, it throws an exception.
  - ◆ Else, the process checks the request data against the validation requirements of an application.
  - ◆ If all data meets the validation requirements, the process forwards the request to the application for further processing.
  - ◆ If one or more data does not meet the validation requirements, the process sends back a validation error message as a response.

© Aptech Ltd.Data Validation and Annotation/Session 34

Using slide 4, explain the students that the MVC Framework implements a validation workflow to validate user data. The validation workflow starts when a user specified data arrives in the server.

Then, tell them that the validation process:

- First, checks whether the request is some form of attack, such as Cross Site Scripting (CSS).
- If the process identifies a request as an attack, it throws an exception.
- Else, the process checks the request data against the validation requirements of an application.
- If all data meets the validation requirements, the process forwards the request to the application for further processing.
- If one or more data does not meet the validation requirements, the process sends back a validation error message as a response.

## Slides 5 to 9

Let us understand manual validation.

**Manual Validation 1-5**

- ◆ In an ASP.NET MVC application, you can manually validate data in the controller action that accepts user data for some kind of processing.
- ◆ To manually validate data, you can implement simple validation routines, such as a routine to check that the length of a password submitted through a registration form exceeds more than seven characters.

© Aptech Ltd. Data Validation and Annotation/Session 5 5

Using slide 5, tell the students that in an ASP.NET MVC application, they can manually validate data in the controller action that accepts user data for some kind of processing.

Then, tell them that to manually validate data, they can implement simple validation routines, such as a routine to check that the length of a password submitted through a registration form exceeds more than seven characters.

## Manual Validation 2-5

- Following code snippet shows a manual validation implemented in an action method:

### Code Snippet:

```
public class HomeController : Controller
{
    Public ActionResult Index() {
        return View();
    }

    [HttpPost]
    Public ActionResult Index(User model) {
        String modelPassword = model.password;
        if (modelPassword.Length< 7) {
            return View();
        }
        else {
            /*Implement registration process*/
            return Content("You have successfully registered");
        }
    }
}
```

Using slide 6, explain the code that shows a manual validation implemented in an action method. In this case, when `HttpPost` call is made to this controller, the `Index` action is invoked which checks for the given password. If the password length is less than seven, then a View is returned otherwise registration process is implemented. Finally, the content 'You have successfully registered' is returned.

## Manual Validation 3-5

- The preceding code:
  - Creates a `HomeController` controller class with two `Index()` action methods.
  - The `firstIndex()` method returns the corresponding view.
  - When a user submits a form displayed by the view the `HttpPost` version of the action method receives a `User` model object that represents the user submitted data.
  - This action method validates whether or not the length of the password property is greater than seven.
  - If the password length is less than seven, the action method returns back the view, else the action method returns a success message.
  - You can also use regular expression while performing manual validations.
  - For example, you can use a regular expression to check whether the user submitted e-mail id is in the correct format, such as `abc@abc.com`.

In slide 7, refer to the code displayed on slide 6 and explain that the preceding code:

- Creates a `HomeController` controller class with two `Index()` action methods.
- The first `Index()` method returns the corresponding view.
- When a user submits a form displayed by the view, the `HttpPost` version of the action method receives a `User` model object that represents the user submitted data.
- This action method validates whether or not the length of the password property is greater than seven.
- If the password length is less than seven, the action method returns back the view, else the action method returns a success message.
- They can also use regular expression while performing manual validations. For example, they can use a regular expression to check whether the user submitted e-mail id is in the correct format, such as `abc@abc.com`.

### Manual Validation 4-5

♦ Following code snippet shows a manual validation implemented in an action method using regular expression:

**Code Snippet:**

```
public class HomeController : Controller {
    public ActionResult Index() {
        return View();
    }
    [HttpPost]
    public ActionResult Index(User model) {
        string modelEmailId = model.email;
        string regexPattern = @"[A-Za-z0-9._]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}";
        if (System.Text.RegularExpressions.Regex.IsMatch(modelEmailId,
            regexPattern)) {
            /*Implement registration process*/
            return Content("You have successfully registered");
        }
        else {
            return View();
        }
    }
}
```

© Aptech Ltd. Data Validation and Annotation/Session 3 8

In slide 8, refer to the code that displays a manual validation implemented in an action method using regular expression.

### Manual Validation 5-5

- ◆ The preceding code:
  - ◆ Uses the `regexPattern` variable to store a regular expression that specifies a valid email id format.
  - ◆ Then, uses the `IsMatch()` method of the `System.Text.RegularExpressions.Regex` class to validate whether or not the `email` property is in valid format. If the value of the `email` property is valid, the action method returns back the view else the action method returns a success message.

In slide 9, refer to the code displayed on slide 8 and explain that the preceding code uses the `regexPattern` variable to store a regular expression that specifies a valid email id format. Then, uses the `IsMatch()` method of the `System.Text.RegularExpressions.Regex` class to validate whether or not the `email` property is in valid format. If the value of the `email` property is valid, the action method returns back the view else the action method returns a success message.

## Slide 10

Let us understand data annotation.

**Data Annotation**

- ◆ The MVC Framework provides several data annotations that you can apply as attributes to a model.
- ◆ These data annotations implement tasks that are commonly required across applications.
- ◆ Some of the important annotations that you can use in the models of an ASP.NET MVC application are as follows:
  - ◆ Required
  - ◆ StringLength
  - ◆ RegularExpression
  - ◆ Range
  - ◆ Compare
  - ◆ DisplayName
  - ◆ ReadOnly
  - ◆ DataType
  - ◆ ScaffoldColumn

© Aptech Ltd. Data Validation and Annotation/Session 3 10

In slide 10, tell the students that the MVC Framework provides several data annotations that they can apply as attributes to a model. These data annotations implement tasks that are commonly required across applications.

Then, tell them about the following important annotations that they can use in the models of an ASP.NET MVC application:

- Required
- StringLength
- RegularExpression
- Range
- Compare
- DisplayName
- ReadOnly
- DataType
- ScaffoldColumn

**Additional Information:**

These data annotations are available in the `System.ComponentModel.DataAnnotations` namespace. Therefore, before using any data annotation, students need to add this namespace in the application.

**Slide 11**

Let us understand the Required annotation.

### Required Annotation 1-9

- ◆ The **Required** data annotation specifies that the property, with which this annotation is associated, is a required property.
- ◆ This attribute raises a validation error if the property value is null or empty.
- ◆ Following is the syntax of the `Required` data annotation:

**Syntax:**

```
[Required]
public string <property_name>;
```

where,

- ◆ `property_name`: Is the name of a model property.

© Aptech Ltd. Data Validation and Annotation/Session 3 11

In slide 11, explain the students that the **Required** data annotation specifies that the property, with which this annotation is associated, is a required property. This attribute raises a validation error if the property value is null or empty.

Then, refer to the syntax of the **Required** data annotation and explain it.

## Slide 12

Let us understand how to use Required annotation.

### Required Annotation 2-9

- Following code snippet shows using the Required annotation in the properties of a User model:

**Code Snippet:**

```
public class User {
    public long Id { get; set; }
    [Required]
    public string Name { get; set; }
    [Required]
    public string Password { get; set; }
    [Required]
    public string ReenterPassword { get; set; }
    [Required]
    public int Age { get; set; }
    [Required]
    public string Email { get; set; } }
```

- In this code, the Required attribute is specified for the Name, Password, ReenterPassword, Age, and Email properties of the User model.

© Aptech Ltd. Data Validation and Annotation/Session 3 12

In slide 12, refer to the code that shows using the Required annotation in the properties of a User model.

Then, explain that in this code, the Required attribute is specified for the Name, Password, ReenterPassword, Age, and Email properties of the User model.

### Slide 13

Let us understand how to set properties of Required validation control.

**Required Annotation 3-9**

- ◆ Once you use the `Required` attributes in the model properties, you should:
  - ◆ Use the `Html.ValidationSummary()` helper method passing true as the parameter.
  - ◆ This method is used to display validation messages on the page.
  - ◆ Use the `Html.ValidationMessageFor()` helper method passing the lambda expression to associate the method with a model property for each field.
  - ◆ This method returns the validation error message for the associated property as HTML.

© Aptech Ltd. Data Validation and Annotation/Session 3 13

Use slide 13 to explain the students that once they use the `Required` attributes in the model properties, they should use:

- The `Html.ValidationSummary()` helper method passing true as the parameter. This method is used to display validation messages on the page.
- The `Html.ValidationMessageFor()` helper method passing the lambda expression to associate the method with a model property for each field. This method returns the validation error message for the associated property as HTML.

## Slide 14

Let us understand the Helper methods.

**Required Annotation 4-9**

- Following code shows the view that uses helper methods to display validation messages:

**Code Snippet:**

```
@model MVCMODELDemo.Models.User @{
    ViewBag.Title = "User Form Validation";
}
<h2>User Form</h2>
@using (Html.BeginForm()) {
    @Html.ValidationSummary(true)
    <div>
        @Html.LabelFor(model =>model.Name)
    </div>
    <div>
        @Html.EditorFor(model =>model.Name)
        @Html.ValidationMessageFor(model =>model.Name)
    </div>
    <div>
        @Html.LabelFor(model =>model.Password)
    </div>
    <div>
        @Html.EditorFor(model =>model.Password)
        @Html.ValidationMessageFor(model =>model.Password)
    </div>
```

© Aptech Ltd. Data Validation and Annotation/Session 3 14

Use slide 14 to refer to the code that shows the view that uses helper methods to display validation messages.

## Slide 15

Let us understand the code generated by Helper methods.

### Required Annotation 5-9

**Code Snippet:**

```
<div>
    @Html.LabelFor(model =>model.ReenterPassword)
</div>
<div>
    @Html.EditorFor(model =>model.ReenterPassword)
    @Html.ValidationMessageFor(model =>model.ReenterPassword)
</div>
<div>
    @Html.LabelFor(model =>model.Age)
</div>
<div>
    @Html.EditorFor(model =>model.Age)
    @Html.ValidationMessageFor(model =>model.Age)
</div>
<div>
    @Html.LabelFor(model =>model.Email)
</div>
<div>
    @Html.EditorFor(model =>model.Email)
    @Html.ValidationMessageFor(model =>model.Email)
</div>
<p>
    <input type="submit" value="Submit" />
</p>
}
```

© Aptech Ltd. Data Validation and Annotation/Session 5 15

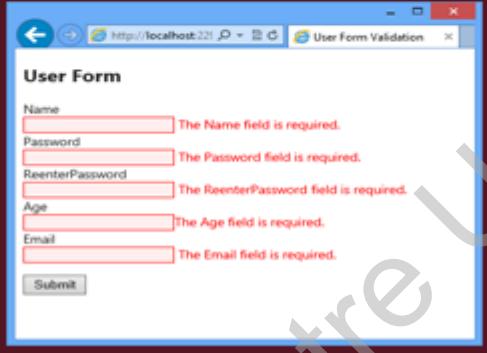
Use slide 15, show that it is the continuation of slide 14. Use both the slides to explain how to add annotations in the ASP.NET MVC applications.

## Slide 16

Let us understand how to use `Html.ValidationSummary` method.

### Required Annotation 6-9

- ◆ In the preceding code:
  - ◆ The `Html.ValidationSummary(true)` helper method displays validation messages as a list.
  - ◆ Then, for each UI field, the `Html.ValidationMessageFor()` helper method is used to validate the corresponding property of the model.
- ◆ Following figure shows the default validation error messages when a user clicks the **Submit** button without specifying any values in the fields:



The screenshot shows a Windows application titled "User Form Validation". It contains a "User Form" window with five text input fields: "Name", "Password", "ReenterPassword", "Age", and "Email". Each field has a red border indicating it is required. Below each field, there is a validation message: "The Name field is required.", "The Password field is required.", "The ReenterPassword field is required.", "The Age field is required.", and "The Email field is required.". A "Submit" button is at the bottom. The status bar at the bottom of the window displays "Data Validation and Annotation/Session 3".

In slide 16, refer to the code displayed on slides 14 and 15 and explain that in the preceding code, the `Html.ValidationSummary(true)` helper method displays validation messages as a list. Then, for each UI field, the `Html.ValidationMessageFor()` helper method is used to validate the corresponding property of the model.

Then, refer to the figure displayed on the slide that shows the default validation error messages when a user clicks the **Submit** button without specifying any values in the fields.

## Slide 17

Let us understand how to specify custom validation message.

### Required Annotation 7-9

- ◆ You can also specify a custom validation message for the Required annotation.
- ◆ The syntax to specify a custom validation message for the Required annotation is as follows:

**Syntax:**

```
[Required(ErrorMessage = <error-message>)]
```

where,

- + **error-message=**: Is the custom error message that you want to display.

© Aptech Ltd. Data Validation and Annotation/Session 3 17

Use slide 17 to explain to the students that they can also specify a custom validation message for the Required annotation.

Refer to the syntax to specify a custom validation message for the Required annotation and explain it.

## Slide 18

Let us understand the code when Required annotation is used with custom validation.

### Required Annotation 8-9

- Following code snippet shows a User model with the Required annotations with custom validation messages applied to its properties:

**Code Snippet:**

```
public class User {
    public long Id { get; set; }
    [Required(ErrorMessage = "Please enter your name.")]
    public string Name { get; set; }
    [Required(ErrorMessage = "Please enter your password.")]
    public string Password { get; set; }
    [Required(ErrorMessage = "Please re-enter your password.")]
    public string ReenterPassword { get; set; }
    [Required(ErrorMessage = "Please enter your age.")]
    public int Age { get; set; }
    [Required(ErrorMessage = "Please enter your Email-ID.")]
    public string Email { get; set; }
}
```

© Aptech Ltd. Data Validation and Annotation/Session 3 18

In slide 18, refer to the code that shows a User model with the Required annotations and custom validation messages applied to its properties.

Then, refer the figure displayed on the next slide that shows the custom validation messages when a user tries to submit the form without specifying any values.

## Slide 19

Let us understand the output when wrong input is given.

**Required Annotation 9-9**

- Following figure shows the custom validation messages when a user tries to submit the form without specifying any values:

The screenshot shows a Windows application window titled "User Form Validation". Inside, there's a "User Form" dialog box. It contains five text input fields with their respective validation error messages displayed next to them:

- Name: Please enter your name.
- Password: Please enter your password.
- ReenterPassword: Please re-enter your password.
- Age: Please enter your age.
- Email: Please enter your Email-ID.

A "Submit" button is at the bottom. The status bar at the bottom of the application window displays "© Aptech Ltd.", "Data Validation and Annotation/Session 3", and "19".

Using slide 19, show the result of application created using validations and annotations. In this example, show the students by entering invalid data and tell them what happens when invalid data is entered in the form. Explain to them that when invalid data is entered, the error messages are automatically shown using annotations.

## Slide 20

Let us understand the `StringLength` annotation.

### StringLength Annotation 1-3

- ◆ You can use the `StringLength` annotation to specify the minimum and maximum lengths of a string field.
- ◆ Following is the syntax for `StringLength` annotation:

Syntax:

```
[StringLength(<max_length>, MinimumLength= <min_length>)]
```

where,

- ◆ `max_length`: Is an integer value that specifies the maximum allowed length.
- ◆ `min_length`: Is an integer value that specifies the minimum allowed length.

© Aptech Ltd. Data Validation and Annotation/Session 3 20

In slide 20, explain the students that they can use the `StringLength` annotation to specify the minimum and maximum lengths of a string field.

Then, refer to the syntax for `StringLength` annotation displayed on the slide and explain it.

## Slide 21

Let us understand how to apply `StringLength` annotation to validate the length of given input.

### StringLength Annotation 2-3

- Following code shows a User model with the `StringLength` annotations applied to its `Password` and `ReenterPassword` properties

**Code Snippet:**

```
public class User { public long Id { get; set; }
    [Required(ErrorMessage = "Please enter your name.")]
    public string Name { get; set; }
    [StringLength(9, MinimumLength = 4)]
    [DataType(DataType.Password)]
    public string Password { get; set; }
    [StringLength(9, MinimumLength = 4)]
    [DataType(DataType.Password)]
    public string ReenterPassword { get; set; }
    [Required(ErrorMessage = "Please enter your age.")]
    public int Age { get; set; }
    [Required(ErrorMessage = "Please enter your Email-ID.")]
    public string Email { get; set; } }
```

- In this code, the `[StringLength]` annotation specifies that the maximum length of the `Password` and `ReenterPassword` properties is set to 9 and the minimum length is set to 4.

© Aptech Ltd. Data Validation and Annotations/Session 3 21

Using slide 21, refer to the code that shows a User model with the `StringLength` annotations applied to its `Password` and `ReenterPassword` properties.

Then, explain to them that in this code, the `[StringLength]` annotation specifies that the maximum length of the `Password` and `ReenterPassword` properties is set to 9 and the minimum length is set to 4.

## Slide 22

Let us understand how the Password and ReenterPassword validations work.

### StringLength Annotation 3-3

- ◆ Whenever a user specified values for these fields are out of this specified range a validation error message is displayed
- ◆ Following figure shows the validation messages when a user specified value in the Password and ReenterPassword fields is out of the specified range:

The screenshot shows a browser window titled "User Form Validation" with the URL "http://localhost:2290/". The form has fields for Name, Password, ReenterPassword, Age, and Email. The Password and ReenterPassword fields are highlighted in red, indicating validation errors. The error message for both fields is: "The field Password must be a string with a minimum length of 4 and a maximum length of 9." The Age field contains "25" and the Email field contains "mark@mvcexample.com". A "Submit" button is at the bottom. The browser status bar shows "Data Validation and Annotation/Session 3".

In slide 22, explain the students that whenever a user specified values for these fields are out of this specified range a validation error message is displayed.

Then, refer to the figure displayed on the slide that shows the validation messages when a user specified value in the Password and ReenterPassword fields is out of the specified range.

**Note:**

Each annotation attribute allows passing an optional parameter named `ErrorMessage` that can be used to specify a custom error message associated with the attribute.

## Slide 23

Let us understand the RegularExpression annotation.

### RegularExpression Annotation 1-4

- ◆ You can use the RegularExpression annotation to accept user input in a specific format.
- ◆ This annotation allows you to match a text string with a search pattern that contains one or more character literals, operators, or constructs.
- ◆ Following is the syntax for RegularExpression annotation:

**Syntax:**

```
[RegularExpression(<pattern>)]
```

where,

- ◆ <pattern>: is the specified format according to which you want user input.

© Aptech Ltd. Data Validation and Annotation/Session 3 23

In slide 23, explain the students that they can use the RegularExpression annotation to accept user input in a specific format. This annotation allows you to match a text string with a search pattern that contains one or more character literals, operators, or constructs.

Then, refer to the syntax for RegularExpression annotation and explain it.

## Slide 24

Let us understand how to apply Email annotation.

### RegularExpression Annotation 2-4

- Following code shows a User model with the RegularExpression annotations applied to its Email property:

**Code Snippet:**

```
public class User {
    public long Id { get; set; }
    [Required(ErrorMessage = "Please enter your name.")]
    public string Name { get; set; }
    [StringLength(9, MinimumLength = 4)]
    public string Password { get; set; }
    [StringLength(9, MinimumLength = 4)]
    public string ReenterPassword { get; set; }
    [Required(ErrorMessage = "Please enter your age.")]
    public int Age { get; set; }
    [RegularExpression(@"[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}",
        ErrorMessage = "Email is not valid.")]
    public string Email { get; set; }
}
```

© Aptech Ltd. Data Validation and Annotation/Session 3 24

In slide 24, refer to the code that shows a User model with the RegularExpression annotations applied to its Email property. First two annotations check for the minimum and maximum length of text entered. Third annotation mentions that this field is mandatory and cannot be left blank. The last annotation enters a regular expression that checks and validates the email entered by the user.

## Slide 25

Let us understand the Regular Expression in detail.

### RegularExpression Annotation 3-4

- ◆ In the preceding code:
  - ◆ The regular expression `A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}` defines the format of an e-mail address. It is divided in three parts where:
    - ◆ Part 1: `[A-Za-z0-9._%+-]+`
    - ◆ Part 2: `[A-Za-z0-9.-]+`
    - ◆ Part 3: `[A-Za-z]{2,4}`
  - ◆ The first part of the regular expression specifies the characters and it ranges within square brackets that can appear.
  - ◆ Then, the + sign between the first and second part indicates that these parts can consist of one or more characters of the types specified within the square brackets preceding the + sign.
  - ◆ The third part contains {2,4} at the end indicating that this part can include 2-4 characters.

© Aptech Ltd. Data Validation and Annotation/Session 3 25

In slide 25, refer to the code displayed on slide 24.

Then, explain them that in the preceding code, the regular expression `A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}` defines the format of an e-mail address. It is divided in three parts, they are as follows:

- Part 1 `[A-Za-z0-9._%+-]+` specifies the characters and it ranges within square brackets that can appear
- Part 2 `[A-Za-z0-9.-]+`, the + sign between the first and second part indicates that these parts can consist of one or more characters of the types specified within the square brackets preceding the + sign.
- Part 3 `[A-Za-z]{2,4}` contains {2,4} at the end indicating that this part can include 2-4 characters.

Next, refer to the figure displayed on slide 26 that shows the validation message that is displayed when a user specified value in the Email field is not in a valid format as specified using regular expression.

## Slide 26

Let us understand how to check Email validation.

**RegularExpression Annotation 4-4**

- Following figure shows the validation message that is displayed when a user specified value in the Email field is not in a valid format as specified using regular expression:

The screenshot shows a Windows application window titled "User Form". It contains several text input fields: Name (Mark), Password, ReenterPassword, Age (16), and Email (mark@com). Below the Email field, there is a red error message: "Email is not valid.". At the bottom of the window is a "Submit" button. The status bar at the bottom of the application window displays "Data Validation and Annotation/Session 3".

With the help of slide 26, explain the output and show the result after entering the valid and invalid text. Show the behavior of the annotations and draw their attention to how the error messages are shown by these annotations after validating the given input.

## Slide 27

Let us understand the Range annotation.

### Range Annotation 1-3

- ◆ You can use the Range annotation to specify the minimum and maximum constraints for a numeric value.
- ◆ Following is the syntax of the Range annotation:

Syntax:

```
[Range (<minimum_range>, <maximum_range>)]
```

where,

- + **minimum\_range**: Is a numeric value that specifies the minimum value for the range.
- + **maximum\_range**: Is a numeric value that specifies the maximum value for the range.

© Aptech Ltd. Data Validation and Annotation/Session 3 27

In slide 27, explain that the student can use the Range annotation to specify the minimum and maximum constraints for a numeric value.

Then, refer to the syntax of the Range annotation and explain it.

## Slide 28

Let us understand how to specify minimum and maximum limits using Range annotations.

### Range Annotation 2-3

- Following code shows a User model with the Range annotations applied to its Age property:

**Code Snippet:**

```
public class User {
    public long Id { get; set; }
    [Required(ErrorMessage = "Please enter your name.")]
    public string Name { get; set; }
    [StringLength(9, MinimumLength = 4)]
    public string Password { get; set; }
    [StringLength(9, MinimumLength = 4)]
    public string ReenterPassword { get; set; }
    [Range(18, 60, ErrorMessage = "The age should be between 18 and 60.")]
    public int Age { get; set; }
    [RegularExpression(@"[A-Za-z0-9._%+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}",
        ErrorMessage = "Email is not valid.")]
    public string Email { get; set; }
}
```

- This code shows using the Range annotations to the Age property.

© Aptech Ltd. Data Validation and Annotation/Session 3 28

In slide 28, refer to the code that shows a User model with the Range annotations applied to its Age property.

Then, refer to the figure displayed on slide 29 that shows the validation error message that is displayed when a user specified age is not in the specified range.

## Slide 29

Let us understand what happens when wrong input is given in Age field.

### Range Annotation 3-3

Following figure shows the validation error message that is displayed when a user specified age is not in the specified range:

The screenshot shows a web browser window titled "User Form Validation" with the URL "http://localhost:221". The form has fields for Name (Mark), Password, ReenterPassword, Age (containing "16"), and Email (containing "mark@mvcexample.com"). A "Submit" button is at the bottom. The "Age" field is highlighted in red, and a red error message "The age should be between 18 and 60." is displayed next to it. The browser's status bar shows "Data Validation and Annotation/Session 3".

© Aptech Ltd. Data Validation and Annotation/Session 3 29

With the help of slide 29, explain to students how the validation is done. Show them by entering wrong inputs. See how the error messages are displayed depending on the given input. One example is shown here where the user input 16 as age which is not acceptable because the condition given here is that, the age should be between 18 and 60. Since input given is wrong, the error message is displayed next to the input box. The color of the text is red in this example, but it can be changed as per our requirements.

## Slide 30

Let us understand the Compare annotation.

### Compare Annotation 1-3

- ◆ You can use the Compare annotation to compare values in two fields.
- ◆ The Compare annotation ensures that the two properties on a model object have the same value.
- ◆ Following code snippet shows a User model with the Compare annotation applied to its ReenterPassword property:

**Code Snippet:**

```
public class User {  
    public long Id { get; set; }  
    [Required(ErrorMessage = "Please enter your name.")]  
    public string Name { get; set; }  
    [StringLength(9, MinimumLength = 4)]  
    public string Password { get; set; }  
    [StringLength(9, MinimumLength = 4)]  
    [Compare("Password", ErrorMessage = "The specified passwords do not  
    match with the Password field.")]
```

© Aptech Ltd. Data Validation and Annotation/Session 5 30

In slide 30, tell the students that they can use the Compare annotation to compare values in two fields. The Compare annotation ensures that the two properties on a model object have the same value.

Then, refer to the code that shows a User model with the Compare annotation applied to its ReenterPassword property.

## Slide 31

Let us understand how to use the Compare annotation code.

### Compare Annotation 2-3

**Code Snippet:**

```
public string ReenterPassword { get; set; }
    [Range(18, 60, ErrorMessage = "The age should be between 18 and 60.")]
    public int Age { get; set; }
    [RegularExpression(@"[A-Za-z0-9._+-]+@[A-Za-z0-9.-]+\.[A-Za-z]{2,4}",
        ErrorMessage = "Email is not valid.")]
    public string Email { get; set; }
```

- ◆ This code uses the Compare annotation to check that the ReenterPassword field contains the same values as that of the Password field.

© Aptech Ltd. Data Validation and Annotation/Session 3 31

In slide 31, refer to the code that uses the Compare annotation to check that the ReenterPassword field contains the same values as that of the Password field.

Refer to the figure displayed on slide 32 that shows the validation error message that is displayed when a user does not enter the same value in both the Password and ReenterPassword fields.

## Slide 32

Let us understand the validation messages.

**Compare Annotation 3-3**

- Following figure shows the validation error message that is displayed when a user does not enter the same value in both the Password and ReenterPassword fields:

The screenshot shows a Windows application window titled "User Form". It contains several input fields: Name (Mark), Password, ReenterPassword, Age (25), and Email (mark@mvcexample.com). The "ReenterPassword" field is highlighted with a red border, indicating an error. A red message box displays the error message: "The specified passwords do not match with the Password field." Below the form is a status bar with the text "© Aptech Ltd.", "Data Validation and Annotation/Session 3", and "32".

With the use of slide 32, explain to students how the error messages are thrown if wrong input is given by the user. Enter wrong values and show the output. Then, enter the correct values and then show the output. Explain to student that it is possible to directly specify the conditions using Compare annotations.

**Slide 33**

Let us understand the `DisplayName` annotation.

### DisplayName Annotation 1-3

- ◆ When you use the `@Html.LabelFor()` helper method in a strongly typed view, the method displays a label with a text whose value is the corresponding property name.
- ◆ You can also explicitly state the text that the `@Html.LabelFor()` method should display using the `DisplayName` annotation on the model property.
- ◆ Following is the syntax for Range annotation:

Syntax:

```
[DisplayName(<text>)]
```

where,

- ◆ `text`: Is the text to be displayed for the property.

© Aptech Ltd. Data Validation and Annotation/Session 5 33

Using slide 33, tell the students that when they use the `@Html.LabelFor()` helper method in a strongly typed view, the method displays a label with a text whose value is the corresponding property name. They can also explicitly state the text that the `@Html.LabelFor()` method should display using the `DisplayName` annotation on the model property.

Then, refer to the syntax for `DisplayName` annotation and explain it.

**Slide 34**

Let us understand the User model with `DisplayName` annotations.

**DisplayName Annotation 2-3**

- Following code shows a User model with the `DisplayName` annotation applied to its `Name`, `ReenterPassword`, and `Email` properties:

**Code Snippet:**

```
public class User {  
    public long Id { get; set; }  
    [DisplayName("User Name")]  
    public string Name { get; set; }  
    public string Password { get; set; }  
    [DisplayName("Re-enter Password")]  
    public string ReenterPassword { get; set; }  
    public int Age { get; set; }  
    [DisplayName("Email- ID")]  
    public string Email { get; set; }  
}
```

- This code applies the `DisplayName` annotation to the `Name`, `ReenterPassword`, and `Email` properties.

34

Using slide 34, refer to the code that shows a User model with the `DisplayName` annotation applied to its `Name`, `ReenterPassword`, and `Email` properties.

Explain the students that this code applies the `DisplayName` annotation to the `Name`, `ReenterPassword`, and `Email` properties.

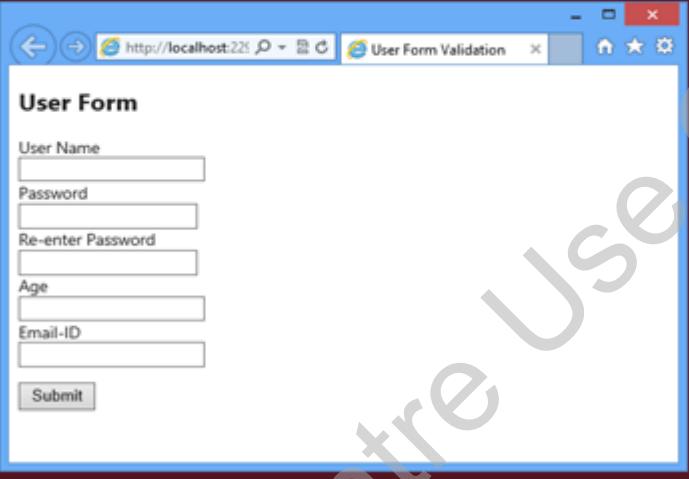
Then, refer to the figure displayed on slide 35 that shows the user friendly display name for the `Name`, `ReenterPassword`, and `Email` model properties.

## Slide 35

Let us understand the interface that uses `DisplayName` annotation.

**DisplayName Annotation 3-3**

- Following figure shows the user friendly display name for the `Name`, `Reenter Password`, and `Email` model properties:



The screenshot shows a web browser window titled "User Form Validation" with the URL "http://localhost:226". The page contains a form with the following fields and their display names:

- User Name
- Password
- Re-enter Password
- Age
- Email-ID

Below the form is a "Submit" button. The entire slide has a dark red background with a large watermark reading "For Aptech Centre Use Only" diagonally across it.

Using slide 35, explain to students how the `DisplayName` annotation works. Tell them that some of the controls will automatically show a prompt before or on the top of the current control. This can be added at runtime using `DisplayName` annotation.

## Slide 36

Let us understand the `ReadOnly` annotation.

### ReadOnly Annotation 1-2

- ◆ You can use the `ReadOnly` annotation to display read-only fields on a form.
- ◆ You can use this annotation to instruct the default model binder not to set the `DiscountedAmount` property with a new value from the request.
- ◆ Following is the syntax for `ReadOnly` annotation:

**Syntax:**

```
[ReadOnly(<boolean_value>)]
```

where,

- ◆ `boolean_value`: Is a boolean value which can be either `true` or `false`. A `true` value indicates that the default model binder should not set a new value for the property. A `false` value indicates that the default model binder should set a new value for the property based on the request.

© Aptech Ltd. Data Validation and Annotation/Session 5 36

In slide 36, explain the students that they can use the `ReadOnly` annotation to display read-only fields on a form. They can use this annotation to instruct the default model binder not to set the `DiscountedAmount` property with a new value from the request.

Then, refer to the syntax for `ReadOnly` annotation and explain it.

### Slide 37

Let us understand how to use `ReadOnly` annotation.

### ReadOnly Annotation 2-2

- Following code snippet shows how to use the `ReadOnly` annotation:

**Code Snippet:**

```
[ReadOnly(true)]
public int DiscountedAmount { get; set; }
```
- This code uses the `ReadOnly` annotation passing `true` as the value for the `DiscountedAmount` property.

© Aptech Ltd. Data Validation and Annotation / Session 3 37

In slide 37, refer to the code that shows how to use the `ReadOnly` annotation.

Tell them that this code uses the `ReadOnly` annotation passing `true` as the value for the `DiscountedAmount` property.

### Slide 38

Let us understand the `DataType` annotation.

### DataType Annotation 1-3

- ◆ You can use the `DataType` annotation to provide information about the specific purpose of a property at run time.
- ◆ Following is the syntax for `DataType` annotation:

Syntax:

```
[DataType(DataType.<value>)]
```

where,

- ◆ `value`: Is a member of `DataType` enumeration.

© Aptech Ltd. Data Validation and Annotation/Session 3 38

In slide 38, explain to the students that they can use the `DataType` annotation to provide information about the specific purpose of a property at run time.

Next, refer to the syntax for `DataType` annotation and explain it.

## Slide 39

Let us understand how to apply `DataType` annotation.

### DataType Annotation 2-3

- ◆ Following code shows applying the `DataType` annotation to the `Password` and `Address` properties of a model:

**Code Snippet:**

```
[DataType(DataType.Password)]
public string Password { get; set; }
[DataType(DataType.MultilineText)]
public string Address { get; set; }
```

- ◆ In this code:
  - ◆ The `DataType` annotation is first applied to the `Password` property. This instructs the view to display password field masks the user entered password.
  - ◆ Next, the `DataType` annotation is applied to the `Address` property. This instructs the view to display a multi-line text area for the `Address` property.

© Aptech Ltd. Data Validation and Annotation/Session 3 39

In slide 39, refer to the code that shows applying the `DataType` annotation to the `Password` and `Address` properties of a model.

Explain them that in this code, the `DataType` annotation is first applied to the `Password` property. This instructs the view to display password field masks the user entered password.

Next, the `DataType` annotation is applied to the `Address` property. This instructs the view to display a multi-line text area for the `Address` property.

Refer to the figure displayed on slide 40 that shows the view that displays the fields for the `Password` and `Address` properties.

## Slide 40

Let us understand how the annotation is applied to Password and Address properties.

**Data Type Annotation 3-3**

- Following figure shows the view that displays the fields for the Password and Address properties:

The screenshot shows a Windows application window titled "User Form". Inside, there is a "Password" label above a text input field containing "\*\*\*\*\*". Below it is an "Address" label above a text area containing "Street No: 12", "Park Street", and "California". At the bottom is a "Submit" button. The window title bar also says "User Form Validation".

© Aptech Ltd. Data Validation and Annotation/Session 3 40

Using slide 40, show the students how the annotation is applied to validate the given password and address. Explain to students what will happen if the wrong values are entered.

## Slide 41

Let us understand the `ScaffoldColumn` annotation.

### ScaffoldColumn Annotation

- When you use scaffolding using the Create template, the view by default will create UI fields for all the properties of the model.
- However, you might need to ensure that the template does not create UI fields for certain properties.
- In such scenarios, you can use the `ScaffoldColumn` annotation passing a false value.
- Following code shows using the `ScaffoldColumn` annotation:

**Code Snippet:**

```
[ScaffoldColumn (false)]  
public long Id { get; set; }
```

- In this code, the `ScaffoldColumn` annotation with a `false` value instructs the Create scaffolding template not to create a UI field in the view for the `Id` property.

© Aptech Ltd. Data Validation and Annotation/Session 3 41

Use slide 41 to explain that when the students use scaffolding, using the Create template, the view by default will create UI fields for all the properties of the model. However, they might need to ensure that the template does not create UI fields for certain properties. In such scenarios, users can use the `ScaffoldColumn` annotation passing a false value.

Then, refer to the code that shows how to use the `ScaffoldColumn` annotation.

Explain them that in this code, the `ScaffoldColumn` annotation with a `false` value instructs the Create scaffolding template not to create a UI field in the view for the `Id` property.

## Slide 42

Let us understand the ModelState validation.

### ModelState Validation 1-2

- ◆ **ModelState:**
  - ◆ Is a class in the `System.Web.Mvc` namespace that encapsulate the state of model binding in an application.
  - ◆ Object stores the values that a user submits to update a model and any validation errors.
  - ◆ You can check whether the state of a model is valid by using the `ModelState.IsValid` property.
  - ◆ If the model binding succeeds, the `ModelState.IsValid` property returns true and you can accordingly perform the required function with the model data.

© Aptech Ltd. Data Validation and Annotation/Session 5 42

In slide 42, explain to the students that ModelState is a class in the `System.Web.Mvc` namespace that encapsulate the state of model binding in an application. Here, the object stores the values that a user submits to update a model and any validation errors.

Tell them that they can check whether the state of a model is valid by using the `ModelState.IsValid` property. If the model binding succeeds, the `ModelState.IsValid` property returns `true`. They can accordingly perform the required function with the model data.

## Slide 43

Let us understand how to use `ModelState.IsValid` property.

### ModelState Validation 2-2

- Following code shows how to use the `ModelState.IsValid` property:  
**Code Snippet:**

```
public ActionResult Index(User model)
{
    if (ModelState.IsValid)
    {
        /*Perform required function with the model data*/
        return Content("You have successfully registered");
    }
    else
    return View();
}
```
- In this code, the `ModelState.IsValid` is used to check the state of the `User` model.
  - If the property returns true, a success message is displayed.
  - Else the view is returned so that the user can enter valid values.

© Aptech Ltd. Data Validation and Annotation/Session 3 43

In slide 43, refer to the code that shows how to use the `ModelState.IsValid` property.

Then, explain them that in this code, the `ModelState.IsValid` is used to check the state of the `User` model. If the property returns true, a success message is displayed.

Else, the view is returned, so that the user can enter valid values.

## Slide 44

Let us summarize the session.

**Summary**

- ◆ The MVC Framework implements a validation workflow to validate user data.
- ◆ In an ASP.NET MVC application, you can manually validate data in the controller action.
- ◆ The MVC Framework provides several data annotations that you can apply as attributes to the properties of a model.
- ◆ The Required annotation when applied to a property validates that a value is specified for that property.
- ◆ The RegularExpression annotation when applied to a property validates that a value specified for that property matches the specified regular expression.
- ◆ The @Html.LabelFor() helper method when used in a strongly typed view displays a label with a text whose value is the corresponding property name.
- ◆ ModelState is a class in the System.Web.Mvc namespace that encapsulates the state of model binding in an application.

© Aptech Ltd. Data Validation and Annotation/Session 3 44

In slide 44, summarize the session. End the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- The MVC Framework implements a validation workflow to validate user data.
- In an ASP.NET MVC application, you can manually validate data in the controller action.
- The MVC Framework provides several data annotations that you can apply as attributes to the properties of a model.
- The Required annotation when applied to a property validates that a value is specified for that property.
- The RegularExpression annotation when applied to a property validates that a value specified for that property matches the specified regular expression.
- The @Html.LabelFor() helper method when used in a strongly typed view displays a label with a text whose value is the corresponding property name.
- ModelState is a class in the System.Web.Mvc namespace that encapsulates the state of model binding in an application.

### **5.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. For what purpose will you use the `Required` annotation?
2. What is the purpose of the `RegularExpression` annotation?
3. What attribute will you use to match the value of two fields in the same form?

## Session 6

### Data Access

#### **6.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth. To check the knowledge of your students, you can even re-cap the last session. Ask some questions related to the last session and test their knowledge. Give necessary inputs and create a background so that the students are ready to perceive the knowledge you are about to give them.

#### **6.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe the Entity Framework
- Explain how to work with the Entity Framework
- Define and describe how to initialize a database with sample data
- Explain how to use LINQ queries to perform database operations

#### **6.1.2 Teaching Skills**

To teach this session successfully, you must know about the Entity Framework. You should know about how to work with Entity Framework and initialize a database with sample data. You should also be aware of how to use LINQ queries to perform database operations in an ASP.NET MVC application.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities:**

Follow the order given here during In-Class activities.

#### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students Slide 2 of the presentation. Tell them that they will be introduced to the Entity Framework. They will learn about how to work with the Entity Framework and initialize a database with sample data. This session will also discuss about how to use LINQ queries to perform database operations.

## **6.2 Introduction**

### **Slide 3**

Let us understand Entity Framework.

The slide has a dark red header bar with the title 'Entity Framework 1-2' in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following text:

- ◆ To address the data access requirements of ASP.NET MVC application, you can use an ORM framework.
- ◆ An ORM framework:
  - ◆ Simplifies the process of accessing data from applications.
  - ◆ Performs the necessary conversions between incompatible type systems in relational databases and object-oriented programming languages.
- ◆ The Entity Framework is an ORM framework that ASP.NET MVC applications can use.
- ◆ The Entity Framework is an implementation of the Entity Data Model (EDM), which is a conceptual model that describes the entities and the associations they participate in an application.
- ◆ EDM allows you to handle data access logic by programming against entities without having to worry about the structure of the underlying data store and how to connect with it.

At the bottom of the slide, there is a footer bar with the following text: © Aptech Ltd., Data Access / Session 6, and a small number 3.

Use slide 3 to explain that to address the data access requirements of ASP.NET MVC application, students can use an ORM framework. An ORM framework simplifies the process of accessing data from applications. It performs the necessary conversions between incompatible type systems in relational databases and object-oriented programming languages.

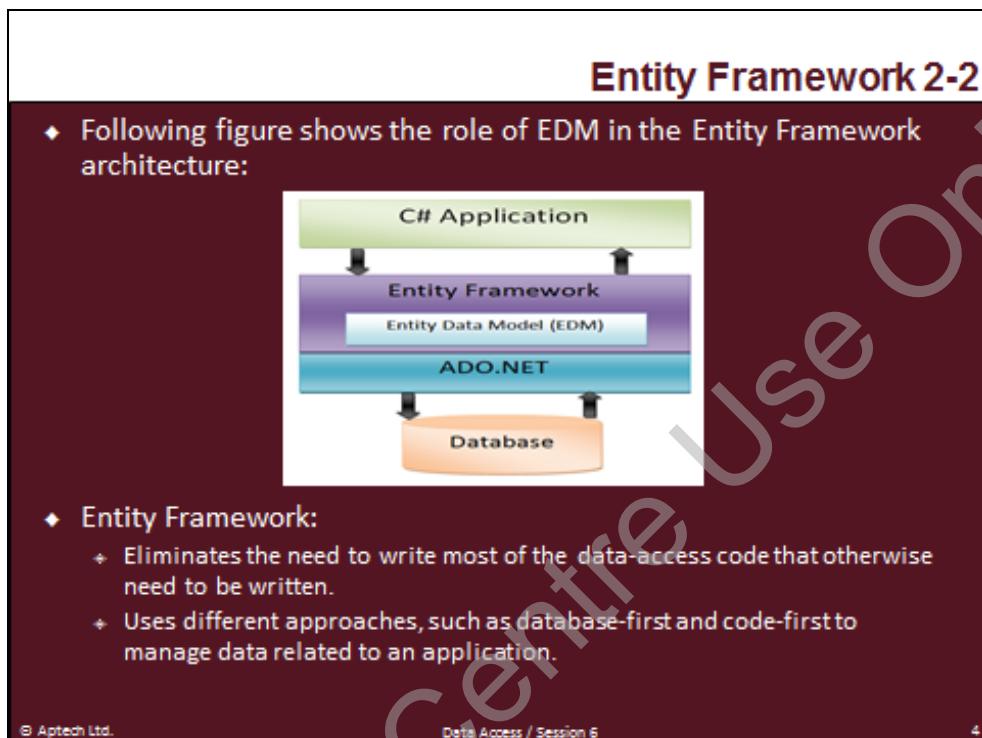
Tell them that the Entity Framework is an ORM framework that ASP.NET MVC applications can use. It is an implementation of the Entity Data Model (EDM), which is a conceptual model that describes the entities and the associations they participate in an application.

Also mention that EDM allows them to handle data access logic by programming against entities without having to worry about the structure of the underlying data store and how to connect with it.

Then you can refer to the figure displayed on the next slide that shows the role of EDM in the Entity Framework architecture.

#### Slide 4

Let us understand the EDM.



In slide 4, explain the students that the Entity Framework eliminates the need to write most of the data-access code that otherwise need to be written. It uses different approaches, such as database-first and code-first to manage data related to an application. In database-first model the database is designed and then the Entity Code is generated. In code-first model the database is generated after developing the code.

## Slide 5

Understand the database-first approach.

### Database-first Approach

- ◆ In the database-first approach the Entity Framework creates model classes and properties corresponding to the existing database objects, such as tables and columns.
- ◆ The database-first approach is applicable in scenario where a database already exists for the application.
- ◆ Following figure shows the database-first approach:

© Aptech Ltd. Data Access / Session 6 5

Display slide 5 and explain the students that in the database-first approach the Entity Framework creates model classes and properties corresponding to the existing database objects, such as tables and columns. The database-first approach is applicable in scenarios where a database already exists for the application.

Then you can refer to the figure displayed on the slide that shows the database-first approach.

## Slide 6

Understand the code-first approach.

### Code-first Approach

- ◆ In the code-first approach the Entity Framework creates database objects based on model classes that you create to represent application data.
- ◆ The code-first approach:
  - ◆ Is the most common approach implemented in ASP.NET MVC Framework.
  - ◆ Allows you to develop your application by coding model classes and properties and delegate the process of creating the database objects to the Entity Framework.
- ◆ Following figure shows the code-first approach:

The diagram illustrates the Code-first Approach. On the left, a blue rectangular box is labeled "Existing Class". An arrow points from this box to a blue cylindrical box on the right, which is labeled "Generated Database". The arrow is labeled "Code First" above it. Below the boxes, the text "Data Access / Session 6" is centered. At the bottom of the slide, there is copyright information: "© Aptech Ltd." on the left, "Data Access / Session 6" in the center, and the number "6" on the right.

In slide 6, you explain the students that in the code-first approach the Entity Framework creates database objects based on model classes that they create to represent application data.

Tell them that the code-first approach is the most common approach implemented in ASP.NET MVC Framework. It allows them to develop their application by coding model classes and properties and delegate the process of creating the database objects to the Entity Framework

Then you refer to the figure displayed on the slide that shows the code-first approach.

## Slide 7

Understand working with Entity Framework.

### Working with Entity Framework

- ◆ The Entity Framework uses different approaches to manage data related to objects.
- ◆ The code-first approach allows you to define your own model by creating custom classes.
- ◆ Then, you can create database based on the models.
- ◆ There are certain conventions that you should follow for the code-first approach.

© Aptech Ltd. Data Access / Session 6 7

In slide 7 you explain the students that the Entity Framework uses different approaches to manage data related to objects. The code-first approach allows them to define their own model by creating custom classes. Then, they can create database based on the models.

Mention that there are certain conventions that they should follow for the code-first approach.

## Slide 8

Let us understand how to implement Code-First approach.

### Implementing Code-first Approach 1-3

- ◆ The code-first approach allows you to provide the description of a model by using the C# classes.
- ◆ Based on the class definitions, the code-first conventions detect the basic structure for the model.
- ◆ Some of the code-first conventions that enable automatic configuration of a model are as follows:
  - + Table naming convention: When you have created an object of the User model and need to store its data in the database, Entity Framework by default creates a table named Users.
  - + Primary key convention: When you create a property named UserId in the User model, the property is accepted as a primary key. In addition, the Entity Framework sets up an auto-incrementing key column to hold the property value.
  - + Relationship convention: Entity Framework provides different conventions to identify a relationship between two models.

© Aptech Ltd. Data Access / Session 6 8

In slide 8, you explain the students that the code-first approach allows them to provide the description of a model by using the C# classes. Based on the class definitions, the code-first conventions detect the basic structure for the model.

Then explain following code-first conventions that enable automatic configuration of a model:

- Table naming convention: When they have created an object of the User model and need to store its data in the database, Entity Framework by default creates a table named Users.
- Primary key convention: When they create a property named UserId in the User model, the property is accepted as a primary key. In addition, the Entity Framework sets up an auto-incrementing key column to hold the property value.
- Relationship convention: Entity Framework provides different conventions to identify a relationship between two models.

## Slide 9

Let us understand how to implement Code-First approach.

### Implementing Code-first Approach 2-3

- ◆ Consider a scenario of an online shopping store where:
  - ◆ You have created two model classes named Customer and Order.
  - ◆ Then you need to declare properties in each class that allows navigating to the properties of another class.
  - ◆ Finally, define the relationship between these two classes.
- ◆ Following code snippet creates a Customer model class:

Snippet

```
public class Customer
{
    public int CustId { get; set; }
    public string Name { get; set; }
    // Navigation property
    public virtual ICollection<Order> Orders { get; set; }
}
```

- ◆ This code creates a model named Customer that contains two properties named, CustId and Name.

© Aptech Ltd. Data Access / Session 6 9

In slide 9, you ask the students to consider a scenario of an online shopping store where they have created two model classes named Customer and Order. Then they need to declare properties in each class that allows navigating to the properties of another class. Finally, define the relationship between these two classes.

Then you refer to the code that creates a Customer model class.

Explain them that this code creates a model named Customer that contains two properties named, CustId and Name.

## Slide 10

Let us understand how the model class is created.

### Implementing Code-first Approach 3-3

- ◆ Following code snippet creates a Order model class:

Snippet

```
public class Order {  
    public int Id { get; set; }  
    public string ProductName { get; set; }  
    public int Price { get; set; }  
    // Foreign key  
    public int CustId { get; set; }  
    // Navigation properties  
    public virtual Customer cust { get; set; }  
}
```

- ◆ In this code:
  - ◆ Orders is the navigational property in the Customer class.
  - ◆ cust is the navigational property in the Order class.
  - ◆ These two properties are known as navigational properties because they allow us to navigate to the properties of another class.

© Aptech Ltd. Data Access / Session 6 10

In slide 10, you refer to the code that creates an Order model class.

Then explain the students that in this code, Orders is the navigational property in the Customer class and cust is the navigational property in the Order class. These two properties are known as navigational properties because they allow us to navigate to the properties of another class.

**Slide 11**

Let us understand the DbContext Class.

## The DbContext Class 1-2

- ◆ The DbContext class:
  - ◆ Is provided by the System.Data.Entity namespace of the ASP.NET MVC Framework.
  - ◆ Can be used to define the database context class after creating a model class.
  - ◆ Coordinates with Entity Framework and allows you to query and save the data in the database.
  - ◆ Uses the DbSet <T> type to define one or more properties where, T represents the type of an object that needs to be stored in the database.

© Aptech Ltd. Data Access / Session 6 11

In slide 11, explain to the students that the DbContext class is provided by the System.Data.Entity namespace of the ASP.NET MVC Framework. It can be used to define the database context class after creating a model class. This class coordinates with Entity Framework and allows them to query and save the data in the database.

Tell them that this class uses the DbSet <T> type to define one or more properties where, T represents the type of an object that needs to be stored in the database.

## Slide 12

Let us understand the DbContext class.

### The DbContext Class 2-2

- Following code snippet shows how to use the DbContext class:

Snippet

```
public class OLShopDataContext : DbContext
{
    public DbSet<Customer> Customers { get; set; }
    public DbSet<Product> Products { get; set; }
}
```

- In this code:
  - A database context class named OLShopDataContext is created that derives from the DbContext class.
  - The DbContext class creates the DbSet property for both, the Customer class and the Product class.

© Aptech Ltd. Data Access / Session 6 12

In slide 12, explain you refer to the code that shows how to use the DbContext class.

Then explain that in this code, a database context class named OLShopDataContext is created that derives from the DbContext class. The DbContext class creates the DbSet property for both, the Customer class and the Product class.

### Slide 13

Let us understand how to initialize the database with test data.

**Initializing a Database with Test Data 1-6**

- ❖ While developing an ASP.NET MVC Web application, you might need to change the model classes to implement various new features.
- ❖ This requires maintaining the database related to the model based on the changes made in the model class.
- ❖ So while modifying the model classes, you should ensure that any changes in the model are reflected back in the database.
- ❖ To maintain the synchronization between the model classes and its associated database, you need to recreate databases.

© Aptech Ltd. Data Access / Session 6 13

Use slide 13 to explain the students that while developing an ASP.NET MVC Web application, they might need to change the model classes to implement various new features. This requires maintaining the database related to the model based on the changes made in the model class. So while modifying the model classes, they should ensure that any changes in the model are reflected back in the database.

Mention that to maintain the synchronization between the model classes and its associated database, they need to recreate databases.

## Slide 14

Let us understand how to initialize the database.

### Initializing a Database with Test Data 2-6

- ❖ The Entity Framework provides the following two classes under the System.Data.Entity namespace that allows recreating databases:
  - ❖ DropCreateDatabaseAlways: Allows recreating an existing database whenever the application starts.
  - ❖ DropCreateDatabaseIfModelChanges: Allows recreating an existing database whenever the associated model class changes.
- ❖ Based on your requirements, you can use one of these two classes in your application to recreate a database.
- ❖ You can use one of these two classes while calling the SetInitializer() method of the System.Data.Entity.Database namespace.

© Aptech Ltd. Data Access / Session 6 14

Use slide 14 to explain the students that the Entity Framework provides the following two classes under the System.Data.Entity namespace that allows recreating databases.

- DropCreateDatabaseAlways: Allows recreating an existing database whenever the application starts.
- DropCreateDatabaseIfModelChanges: Allows recreating an existing database whenever the associated model class changes.

Tell them that based on their requirements, they can use one of these two classes in an application to recreate a database. They can use one of these two classes while calling the SetInitializer() method of the System.Data.Entity.Database namespace.

## Slide 15

Let us understand the use of DropCreateDatabaseAlways class.

### Initializing a Database with Test Data 3-6

- Following code snippet shows using the DropCreateDatabaseAlways class inside the Application\_Start() method of the Global.asax.cs file:

```
Database.SetInitializer(new
DropCreateDatabaseAlways<ShopDataContext>());
```

- In this code, the DropCreateDatabaseAlways class is used while calling the SetInitializer() method to ensure that the existing database is recreated whenever the application starts.
- On the other hand, you can use the DropCreateDatabaseIfModelChanges class to recreate a database only when the model changes.

© Aptech Ltd. Data Access / Session 6 15

In slide 15, you refer to the code that shows using the DropCreateDatabaseAlways class inside the Application\_Start() method of the Global.asax.cs file.

Explain that in this code, the DropCreateDatabaseAlways class is used while calling the SetInitializer() method to ensure that the existing database is recreated whenever the application starts. On the other hand, they can use the DropCreateDatabaseIfModelChanges class to recreate a database only when the model changes.

#### **Note:**

Tell them that they need to ensure that the required namespaces are included. Otherwise, the code will not run.

## Slide 16

Let us understand the use of DropCreateDatabaseIfModelChanges class.

### Initializing a Database with Test Data 4-6

- Following code snippet shows using the DropCreateDatabaseIfModelChanges class inside the Application\_Start() method:

Database.SetInitializer(new DropCreateDatabaseIfModelChanges<ShopDataContext>());

- In this code, the DropCreateDatabaseIfModelChanges class is used while calling the SetInitializer() method to ensure that the existing database is recreated whenever the model changes.
- You can also instruct the MVC Framework to populate a database with sample data for an application.
- This can be achieved by creating a class that derives from either the DropCreateDatabaseIfModelChanges class or the DropCreateDatabaseAlways class.

Snippet

© Aptech Ltd. Data Access / Session 6 16

In slide 16, you refer to the code that shows using the DropCreateDatabaseIfModelChanges class inside the Application\_Start() method.

Then explain the students that in this code, the DropCreateDatabaseIfModelChanges class is used while calling the SetInitializer() method to ensure that the existing database is recreated whenever the model changes.

Tell them that they can also instruct the MVC Framework to populate a database with sample data for an application. This can be achieved by creating a class that derives from either the DropCreateDatabaseIfModelChanges class or the DropCreateDatabaseAlways class.

## Slide 17

Let us understand the use of MyDbInitializer model class.

### Initializing a Database with Test Data 5-6

Following code snippet shows the MyDbInitializer model class that uses the Seed() method to insert some sample data in the Customers database:

```
public class MyDbInitializer : DropCreateDatabaseIfModelChanges<OLShopDataContext>
{
    protected override void Seed(OLShopDataContext context)
    {
        context.Customers.Add(new Customer() { Name = "John Parker",
Address="Park Street",Email="john@mvcexample.com" });
        base.Seed(context);
    }
}
```

In this code:

- ◆ The MyDbInitializer class is derived from the DropCreateDatabaseIfModelChanges class.
- ◆ Then, the Seed() method is overridden to define the initial data for the Customer model.

Snippet

© Aptech Ltd. Data Access / Session 6 17

In slide 17 you refer to the code that shows a MyDbInitializer model class that uses the Seed( ) method to insert some sample data in the Customers database.

Explain them that in this code the MyDbInitializer class is derived from the DropCreateDatabaseIfModelChanges class. Then, the Seed( ) method is overridden to define the initial data for the Customer model.

## Slide 18

Let us understand how to use SetInitializer() method to register MyDbInitializer class.

### Initializing a Database with Test Data 6-6

- Once you have defined the initial data for the customers, you need to register the MyDbInitializer model class in the Global.asax.cs file by calling the SetInitializer() method.
- Following code snippet shows using the SetInitializer() method inside the Application\_Start() method:

Snippet

```
protected void Application_Start()
{
    System.Data.Entity.Database.SetInitializer(new
        MyDbInitializer());
}
```

- This code uses the SetInitializer() method to register the MyDbInitializer model class.

In slide 18, you explain the students that once they have defined the initial data for the customers, they need to register the MyDbInitializer model class in the Global.asax.cs file by calling the SetInitializer() method.

The you refer to the code that shows using the SetInitializer() method inside the Application\_Start() method.

Explain the students that this code uses the SetInitializer() method to register the MyDbInitializer model class.

## Slide 19

Let us understand LINQ query.

### LINQ Query

- ◆ LINQ:
  - ◆ Is a set of APIs that allows you to create data-source-independent queries to implement data access in an application.
  - ◆ Has a programming model that provides the standard query syntax to query different types of data sources.
  - ◆ Queries are written in any .NET Framework supported programming language, such as VB or C#.
  - ◆ Query acts on a strongly-typed collection of objects with the help of language keywords and common operators.
  - ◆ Helps you to quickly manipulate data from the various data sources without requiring them to learn a new query language.

© Aptech Ltd. Data Access / Session 6 19

In slide 19, you explain the students that LINQ is a set of APIs that allows them to create data-source-independent queries to implement data access in an application. It has a programming model that provides the standard query syntax to query different types of data sources.

LINQ queries are written in any .NET Framework supported programming language, such as VB or C#. These queries acts on a strongly-typed collection of objects with the help of language keywords and common operators.

Mention that LINQ query helps them to quickly manipulate data from the various data sources without requiring them to learn a new query language.

## Slide 20

Let us understand how to use a simple LINQ query.

### Using a Simple LINQ Query 1-3

- ◆ In an ASP.NET MVC application:
  - ◆ A LINQ query operation starts by creating a query.
  - ◆ Once the query is created, the next operation is to execute the query.
  - ◆ To execute a LINQ query, you need to first specify the data source.
  - ◆ LINQ queries operate on objects that implement either the `IEnumerable<T>` or `IQueryable<T>` interface.
  - ◆ Both these interfaces enable you to create queries to retrieve data from a specific data source.

© Aptech Ltd. Data Access / Session 6 20

In slide 20, you explain the students that in an ASP.NET MVC application a LINQ query operation starts by creating a query. Once the query is created, the next operation is to execute the query. To execute a LINQ query, they need to first specify the data source.

Tell them that LINQ queries operate on objects that implement either the `IEnumerable<T>` or `IQueryable<T>` interface. Both these interfaces enable them to create queries to retrieve data from a specific data source.

## Slide 21

Let us understand how to use select query in LINQ.

### Using a Simple LINQ Query 2-3

- Following code snippet shows creating a query that retrieves customer details from the Customer data source:

Snippet

```
IQueryable<Customer> q = from s in db.customers select s;
```

where,

- from: Clause specifies the data source that contains the data.
- db: Is an instance of the data context class provides access to the data source.
- s: Is the range variable.
- select: Clause specifies that each element in the result will consist of a customer object.

Using slide 21, refer to the code that shows creating a query that retrieves customer details from the Customer data source. This query is similar to SELECT query used in any database. It retrieves all the matching records as per the given criteria. In the given slide the LINQ statement 'from s in db.customers select s;' retrieves all the records from db.customers object and returns in a object called 'q'. You can iterate through this object to retrieve information store in the form of records in the object 'q'.

## Slide 22

Let us understand how to use advance LINQ queries.

### Using a Simple LINQ Query 3-3

- ♦ Once you have created a LINQ query:
  - ♦ You need to execute it to retrieve the data.
  - ♦ Then iterate over the result set using a foreach loop.
- ♦ Following code snippet shows using foreach loop to retrieve customer details:

```
string names = "";
IQueryable<Customer> q = from s in db.customers
                           select s;
foreach (var cust in q)
{
    names = names + " " + cust.Name;
}
ViewBag.Name = names;
```

Snippet

- ♦ In this code:
  - ♦ The foreach loop retrieves the query results and the cust variable stores each value one at a time.
  - ♦ Then, the name of each customer is appended to the string variable, names.
  - ♦ Finally, the names variable is assigned to the Name property using ViewBag to display the information in a view.

© Aptech Ltd. Data Access / Session 6 22

In slide 22, you explain the students that once they have created a LINQ query, they need to execute it to retrieve the data. Then iterate over the result set using a foreach loop.

Then refer to the code that shows using foreach loop to retrieve customer details.

Explain the students that in this code the foreach loop retrieves the query results and the cust variable stores each value one at a time. Then, the name of each customer is appended to the string variable, names. Finally, the names variable is assigned to the Name property using ViewBag to display the information in a view.

## Slide 23

Let us understand how to use advance LINQ queries.

### Using Advance LINQ Queries 1-7

- ◆ In addition to retrieve data from data sources, you can also use LINQ queries to perform various operations on data stored in a data source.
- ◆ Some of the common operations that you perform on data using LINQ queries include:
  - ◆ Forming Projections
  - ◆ Filtering the data
  - ◆ Sorting the data
  - ◆ Grouping the data

© Aptech Ltd. Data Access / Session 6 23

In slide 23, you explain the students that in addition to retrieving data from data sources, you can also use LINQ queries to perform various operations on data stored in a data source.

Explain about the following common operations that they perform on data using LINQ queries include:

- Forming Projections
- Filtering the data
- Sorting the data
- Grouping the data

## Slide 24

Let us understand how to use advanced queries in LINQ.

### Using Advance LINQ Queries 2-7

- ♦ Sometimes, you might only need to retrieve specific properties of a model from the data store, for example, only the Name property of the Customer model.
- ♦ You can achieve this by forming projections in the select clause.
- ♦ Following code snippet shows a LINQ query that retrieves only the customer names of the Customer model class:

Snippet

```
public static void DisplayCustomerNames() {
    using (Model1Container dbContext = new Model1Container()) {
        IQueryables<String> query = from c in dbContext.Customers
                                       select c.Name;

        Console.WriteLine("Customer Names:");
        foreach (String custName in query)
        {
            Console.WriteLine(custName);
        }
    }
}
```

© Aptech Ltd. Data Access / Session 6 24

In slide 24, you explain the students that sometimes, they might only need to retrieve specific properties of a model from the data store, for example, only the Name property of the Customer model. They can achieve this by forming projections in the select clause.

Then refer to the code that shows a LINQ query that retrieves only the customer names of the Customer model class.

## Slide 25

Let us understand how the selective data is retrieved using advanced LINQ queries.

### Using Advance LINQ Queries 3-7

- ❖ In the preceding code:
  - ❖ The select clause retrieves a sequence of customer names as an `IQueryable<String>` object.
  - ❖ The foreach loop iterates through the result to print out the names.
- ❖ Following is the output of the preceding code:

Customer Names:

Alex Parker

Peter Milne

In slide 25, you refer to the code displayed on slide 24.

Then explain them that in the preceding code the `select` clause retrieves a sequence of customer names as an `IQueryable<String>` object. Then the `foreach` loop iterates through the result to print out the names. The object `IQueryable` holds the data as collection of strings which can be retrieved and used by using `foreach` loop construct to iterate through the collection of strings.

## Slide 26

Let us understand how to filter records using where clause with LINQ.

### Using Advance LINQ Queries 4-7

- ◆ The where clause in a LINQ query enables filtering data based on a Boolean condition, known as the predicate.
- ◆ The where clause applies the predicate to the range variable that represents the source elements and returns only those elements for which the predicate is true.
- ◆ Following code snippet uses the where clause to filter customer records:

Snippet

```
public static void DisplayCustomerByName() {
    using (Model1Container dbContext = new Model1Container()) {
        IQueryable<Customer> query = from c in dbContext.Customers
        where c.Name == "Alex Parker" select c;
        Console.WriteLine("Customer Information:");
        foreach (Customer cust in query)
        {
            Console.WriteLine("Customer ID: {0}, Name: {1},
Address: {2}", cust.CustomerId, cust.Name, cust.Address);
        }
    }
}
```

© Aptech Ltd. Data Access / Session 6 26

In slide 26, you explain the students that the where clause in a LINQ query enables filtering data based on a Boolean condition, known as the predicate. The where clause applies the predicate to the range variable that represents the source elements and returns only those elements for which the predicate is true.

Then refer to the code that uses the where clause to filter customer records.

**Slide 27**

Let us understand how to filter records using where clause with LINQ

## Using Advance LINQ Queries 5-7

- ◆ In the preceding code:
  - ◆ The where clause to retrieve information of the customer with the name Alex Parker.
  - ◆ The foreach statement iterate through the result to print the information of the customer.
- ◆ Following is the output of the preceding code:

Customer Information:

Customer ID: 1, Name: Alex Parker, Address: 10th Park Street, Leo Mount

© Aptech Ltd. Data Access / Session 6 27

In slide 27, you refer to the code displayed on slide 26 and explain that in the preceding code the where clause to retrieve information of the customer with the name Alex Parker. The foreach statement iterate through the result to print the information of the customer.

## Slide 28

Let us understand how to use orderby clause in LINQ query.

### Using Advance LINQ Queries 6-7

- ❖ You can also use LINQ queries to retrieve data and then, display it in sorted manner by using the orderby clause.
- ❖ This clause specifies whether the result should be displayed in either ascending or descending order.
- ❖ While using the orderby clause the result is displayed in the ascending order by default.
- ❖ Following code snippet shows using the ascending keyword to display the customer name in ascending order:

#### Snippet

```
IQueryable<Customer> q = from s in dbContext.customers  
                           where s.City == "New Jersey"  
                           orderby s.Name ascending  
                           select s;
```

- ❖ This code will retrieve and then, displays the customer name who lives in New Jersey in the ascending order.

In slide 28, explain that they can also use LINQ queries to retrieve data and then, display it in sorted manner by using the orderby clause. This clause specifies whether the result should be displayed in either ascending or descending order. While using the orderby clause the result is displayed in the ascending order by default.

Then you refer to the code that shows using the ascending keyword to display the customer name in ascending order.

Explain them that this code will retrieve and displays the customer name that lives in New Jersey in the ascending order.

**Slide 29**

Let us understand how to use group by clause with LINQ query.

### Using Advance LINQ Queries 7-7

- ◆ You can also use LINQ query to retrieve and display the data as a group.
- ◆ For this, you need to use the group clause that allows you to group the results based on a specified key.
- ◆ Following code snippet shows how to group the customers according to their cities:

Snippet

```
var q = from s in dbContext.customers  
        groups by s.City;
```

- ◆ This code retrieves the customer records and groups these records based on the city where the customers lives.

In slide 29, explain that they can also use LINQ query to retrieve and display the data as a group. For this, they need to use the group clause that allows them to group the results based on a specified key.

Then refer to the code that shows how to group the customers according to their cities.

Explain the students that this code retrieves the customer records and groups these records based on the city where the customers lives.

**Slide 30**

Let us understand how to use LINQ method-based queries.

### **Using LINQ Method-Based Queries 1-3**

- ◆ The LINQ queries used so far are created using query expression syntax.
- ◆ Such queries are compiled into method calls to the standard query operators, such as select, where, and orderby.
- ◆ Another way to create LINQ queries is by using method-based queries where you can directly make method calls to the standard query operator, passing lambda expressions as the parameters.

© Aptech Ltd. Data Access / Session 6 30

In slide 30, tell the students that the LINQ queries used so far are created using query expression syntax. Such queries are compiled into method calls to the standard query operators, such as select, where, and orderby.

Tell them that another way to create LINQ queries is by using method-based queries where you can directly make method calls to the standard query operator, passing lambda expressions as the parameters.

## Slide 31

Let us understand how to use select clause of LINQ to retrieve specific properties.

### Using LINQ Method-Based Queries 2-3

- Following code snippet shows the use of Select clause to project the Name and Address properties of Customer model class into a sequence of anonymous types:

Snippet

```
public static void DisplayPropertiesMethodBasedQuery() {
    using (Model1Container dbContext = new Model1Container()) {
        var query =dbContext.Customers.Select(c=> new
        {
            CustomerName = c.Name,
            CustomerAddress = c.Address
        });
        Console.WriteLine("Customer Names and Addresses:");
        foreach (var custInfo in query)
        {
            Console.WriteLine("Name: {0}, Address: {1}",
            custInfo.CustomerName, custInfo.CustomerAddress);
        }
    }
}
```

© Aptech Ltd. Data Access / Session 6 31

In slide 31, you refer to the code that shows the use of Select clause to project the Name and Address properties of Customer model class into a sequence of anonymous types.

**Slide 32**

Let us see the output after using select clause.

### Using LINQ Method-Based Queries 3-3

- ◆ The preceding code uses the Select clause to project the Name and Address properties of Customer model class into a sequence of anonymous types.
- ◆ Following is the output of the preceding code:

Customer Names and Addresses:

Name: Alex Parker, Address: 10th Park Street, Leo Mount

Name: Peter Milne, Address: Lake View Street, Cheros Mount
- ◆ Similarly, you can use the other operators such as Where, GroupBy, Max, and so on through method-based queries.

In slide 32, you refer to the code displayed on slide 31 and explain them that this code uses the Select clause to project the Name and Address properties of Customer model class into a sequence of anonymous types.

Then tell them that similarly, they can use the other operators such as Where, GroupBy, Max, and so on through method-based queries.

**Slide 33**

Let us understand LINQ data providers.

**LINQ Data Providers**

- ◆ LINQ provides a consistent programming model to create standard query expression syntax to query different types of data sources.
- ◆ However, different data sources accept queries in different formats.
- ◆ To solve this problem, there are several LINQ providers, such as LINQ to SQL, LINQ to Objects, and LINQ to XML.
- ◆ The LINQ queries, that you have learned till now uses the LINQ to SQL provider.
- ◆ This provider allows you to access SQL-complaint databases and makes data available as objects in an application.

© Aptech Ltd. Data Access / Session 6 33

Display slide 33, and tell the students that LINQ provides a consistent programming model to create standard query expression syntax to query different types of data sources. However, different data sources accept queries in different formats. To solve this problem, there are several LINQ providers, such as LINQ to SQL, LINQ to Objects, and LINQ to XML.

Tell them that the LINQ queries, that they have learned till now uses the LINQ to SQL provider. This provider allows accessing SQL-complaint databases and makes data available as objects in an application.

## Slide 34

Let us understand LINQ to Objects.

### LINQ to Objects

- ◆ LINQ to Objects refers to the use of LINQ queries with enumerable collections, such as `List<T>` or arrays.
- ◆ To use LINQ to query an object collection, you need to declare a range variable.
- ◆ Following code snippet shows how to access data from a string array that contains the names of customers in a class:

Snippet

```
string products= "";
string[] arr = new string[] { "Laptop", "Mobile", "Jewellery"};
var query = from string product in arr
select product;
foreach (var p in query)
{
    products= products+ " " + p;
}
```

- ◆ This code accesses the data from a string array that contains the names of customers.

© Aptech Ltd. Data Access/ Session 6 34

In slide 34, explain to the students that LINQ to Objects refers to the use of LINQ queries with enumerable collections, such as `List<T>` or arrays. To use LINQ to query an object collection, they need to declare a range variable.

Then refer to the code displayed on the slide that shows how to access data from a string array that contains the names of customers in a class.

Explain the students that this code accesses the data from a string array that contains the names of customers.

## Slide 35

Let us understand LINQ to XML.

### LINQ to XML 1-2

- ◆ You can use the LINQ to XML queries to work with XML in an application.
- ◆ LINQ to XML query allows accessing data that is stored in XML files.
- ◆ Following code snippet shows the content of an XML file named, CustomersRecord.xml:

Snippet

```
<?xml version="1.0" encoding="utf-8"?>
<CustomerDetails>
<Customer>
<CustID>CId1001</CustID>
<Name>Peter Jones</Name>
<City>New York</City>
</Customer>
<Customer>
<CustID>CId1002</CustID>
<Name>Jessica Parker</Name>
<City>London</City>
</Customer>
</CustomerDetails>
```

- ◆ This code shows an XML document, named CustomersRecord.Xml that contains customer details.

© Aptech Ltd. Data Access / Session 6 33

In slide 35, explain the students that they can use the LINQ to XML queries to work with XML in an application. LINQ to XML query allows accessing data that is stored in XML files.

Then you refer to the code displayed on the slide that shows the content of an XML file named, CustomersRecord.xml.

Explain them that this code shows an XML document, named CustomersRecord.Xml that contains customer details.

**Slide 36**

Let us understand how to retrieve data from XML file using LINQ.

## LINQ to XML 2-2

- ◆ Now, to access the data from CustomersRecord.Xml you can use LINQ.
- ◆ Following code snippet shows how to retrieve data from an XML file by using LINQ query:

```
string result = "";
XDocument xmlDoc = XDocument.Load("E:\\CustomersRecord.xml");
var q = from c in xmlDoc.Descendants("Customer")
        select (string)c.Element("CustID") + "-" +
               (string)c.Element("Name") + "-" + (string)c.Element("City");
foreach (string entry in q)
{
    result += entry + " | ";
}
```

**Snippet**

- ◆ This code will retrieve all the customer details from the CustomersRecord.Xml file.
- ◆ The xmlDoc.Descendants() method returns a collection of the descendant elements for the specified element in the order they are present in the XML file.

© Aptech Ltd. Data Access / Session 6 36

In slide 36, explain the students that to access the data from CustomersRecord.Xml they can use LINQ. You then refer to the code displayed on the slide that shows how to retrieve data from an XML file by using LINQ query.

Explain the students that this code will retrieve all the customer details from the CustomersRecord.Xml file. The xmlDoc.Descendants() method returns a collection of the descendant elements for the specified element in the order they are present in the XML file.

## Slide 37

Let us summarize the session.

### Summary

- ◆ The Entity Framework is an ORM framework that ASP.NET MVC applications can use.
- ◆ In the database-first approach, the Entity Framework creates model classes and properties corresponding to the existing database objects.
- ◆ In the code-first approach, the Entity Framework creates database objects based on model classes that you create to represent application data.
- ◆ The System.Data.Entity namespace of the ASP.NET MVC Framework provides a DbContext class that coordinates with Entity Framework and allows you to query and save the data in the database.
- ◆ LINQ is a set of APIs that allows you to create data-source-independent queries to implement data access in an application.
- ◆ LINQ queries operate on objects that implement either the IEnumerable<T> or IQueryable<T> interface.
- ◆ LINQ queries can also be created as method-based queries where you can directly make method calls to the standard query operator, passing lambda expressions as the parameters.

© Aptech Ltd. Data Access / Session 6 37

In Slide 37, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- The Entity Framework is an ORM framework that ASP.NET MVC applications can use.
- In the database-first approach, the Entity Framework creates model classes and properties corresponding to the existing database objects.
- In the code-first approach, the Entity Framework creates database objects based on model classes that you create to represent application data.
- The System.Data.Entity namespace of the ASP.NET MVC Framework provides a DbContext class that coordinates with Entity Framework and allows you to query and save the data in the database.
- LINQ is a set of APIs that allows you to create data-source-independent queries to implement data access in an application.
- LINQ queries operate on objects that implement either the IEnumerable<T> or IQueryable<T> interface.
- LINQ queries can also be created as method-based queries where you can directly make method calls to the standard query operator, passing lambda expressions as the parameters.

### **6.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. Which class will you use to recreate an existing database whenever the application starts?
2. Which LINQ providers enable you to access relational databases?

## Session 7 -

# Consistent Styles and Layouts

### **7.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **7.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe how to use layout to maintain consistent look and feel
- Explain the process of creating a custom layout
- Explain how to implement styles in an application
- Explain and describe how to create adaptive styles

#### **7.1.2 Teaching Skills**

To teach this session successfully, you must know about the layout. You should know about how to create a custom layout. You should also be aware of how to implement styles and create adaptive styles in an ASP.NET MVC application.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

#### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students Slide 2 of the presentation. Tell them that they will be introduced to the layout in an application. They will learn about how to create a custom layout. This session will also discuss about how to implement styles in an application and create adaptive styles.

## 7.2 In-Class Explanation

### Slide 3

Understand consistent look and feel using layouts.

**Consistent Look and Feel Using Layouts**

- ◆ ASP.NET MVC Framework allows you to use layouts that simplify the process of maintaining consistent look and feel in an application.
- ◆ You can use layout that allows you to specify a style template to be used across the views in an application.
- ◆ In this layout, you can define the structure and common styles for the views.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 3

Use slide 3 to explain that ASP.NET MVC Framework allows them to use layouts that simplify the process of maintaining consistent look and feel in an application. They can use layout that allows specifying a style template to be used across the views in an application.

Tell them that in this layout, they can define the structure and common styles for the views.

#### Additional Information:

Explain the students that they can use a layout with the selected views in an application or with all the views in an application.

**Slide 4**

Understand the \_Layout.cshtml file.

**\_Layout.cshtml File**

- ◆ When you create an ASP.NET MVC Web application in Visual Studio 2013, a \_Layout.cshtml file is automatically added in the View/Shared folder.
- ◆ This file represents the layout that you can apply to the views of the application.
- ◆ This \_Layout.cshtml file contains the basic structure of a view.
- ◆ While developing an application, you can also modify the \_Layout.cshtml based on your requirements.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 4

In slide 4, explain the students that when they create an ASP.NET MVC Web application in Visual Studio 2013, a \_Layout.cshtml file is automatically added in the View/Shared folder. This file represents the layout that they can apply to the views of the application. This \_Layout.cshtml file contains the basic structure of a view.

Tell them that while developing an application, they can also modify the \_Layout.cshtml based on their requirements.

**Note:**

You can modify the \_Layout.cshtml file according to their need for developing an MVC application. For example, they can add add/delete hyperlinks, change the layout color, and include/exclude sections.

## Slide 5

Understand custom layout.

The slide has a dark purple background with a faint geometric pattern. At the top right, the title 'Custom Layout 1-6' is displayed in a blue font. Below the title is a bulleted list of instructions:

- ◆ While developing an ASP.NET MVC Web application, sometime you might require creating your custom layout.
- ◆ Creating a layout in Visual Studio 2013 is similar to creating a view.
- ◆ By default, the name of the layout file is preceded by the underscore (\_) character.
- ◆ The extension of a layout file is .cshtml.
- ◆ To create a custom layout using Visual Studio 2013, you need to perform the following steps:
  - + Right-click the Shared folder under the Views folder in the Solution Explorer window.
  - + Select Add → New Item from the context menu that appears. The Add New Item dialog box is displayed.
  - + Select MVC 5 Layout Page (Razor) template.
  - + Enter \_CustomLayout in the Name text field.

At the bottom left of the slide, there is a small watermark-like text '© Aptech Ltd.' and at the bottom center, it says 'Consistent Styles and Layouts / Session 7'. On the far right edge of the slide, there is a small number '5'.

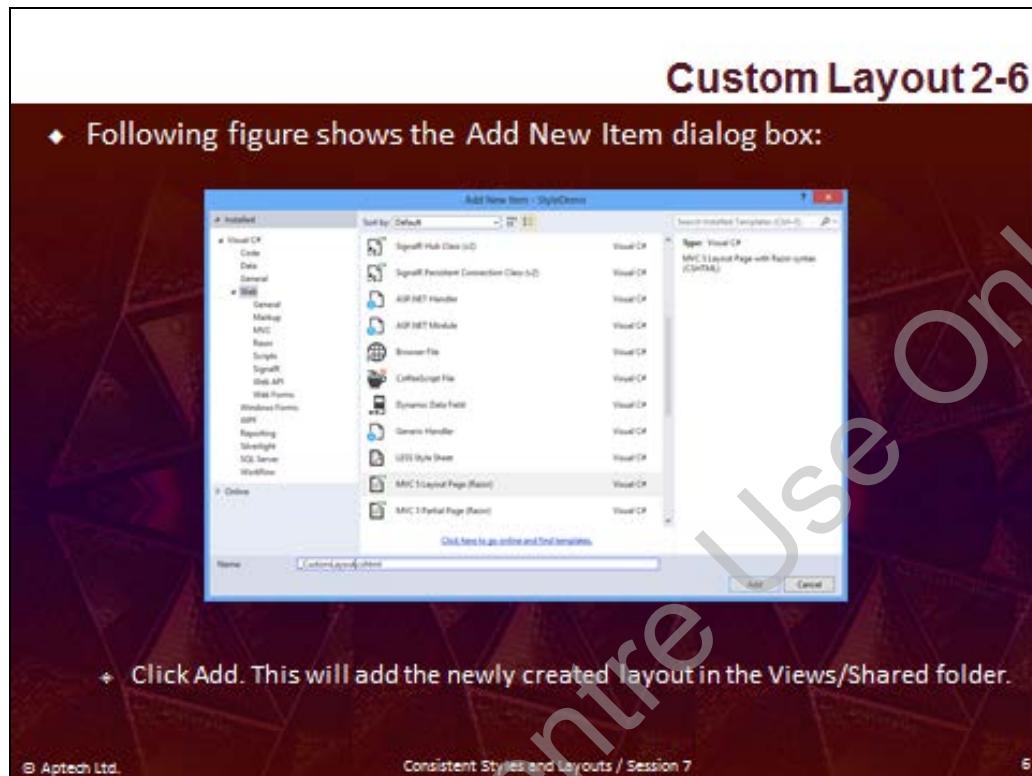
Display slide 5 and explain the students that while developing an ASP.NET MVC Web application, sometime they might require creating a custom layout. Creating a layout in Visual Studio 2013 is similar to creating a view. The extension of a layout file is .cshtml.

By default, the name of the layout file is preceded by the underscore (\_) character.

Then explain the students about the steps to create a custom layout using Visual Studio 2013 as displayed on slide 5.

## Slide 6

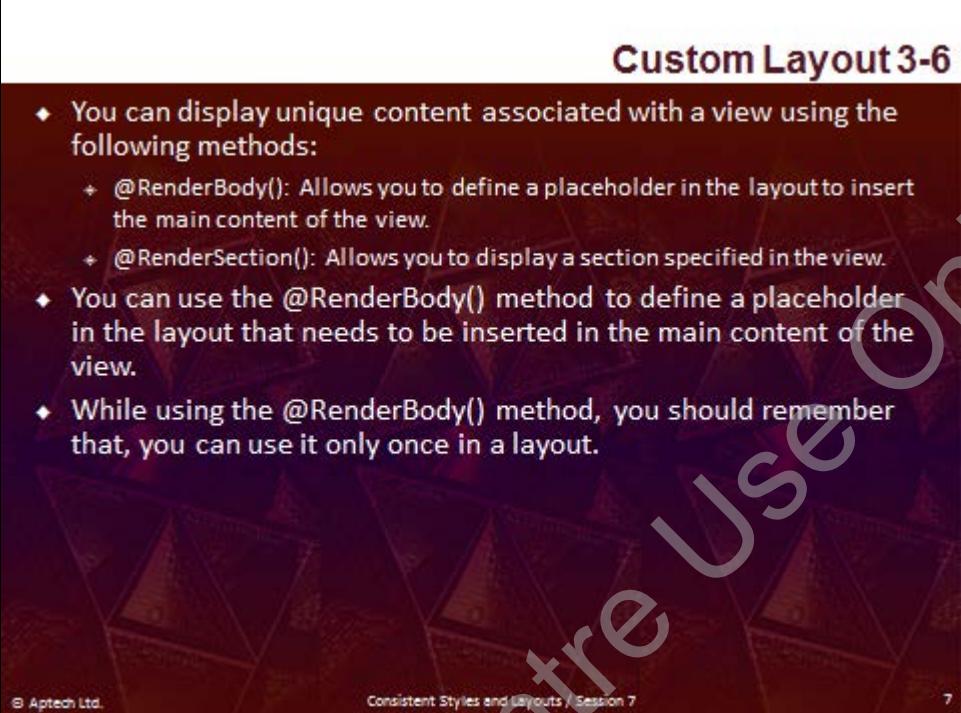
Understand custom layout.



In slide 6, you refer to the figure that shows the Add New Item dialog box. Show the students how to select the templates and how to add the new items into the current project.

## Slide 7

Understand custom layout.



The slide has a dark purple background with a faint geometric pattern of triangles. At the top center, the title "Custom Layout 3-6" is displayed in a white, bold font. Below the title, there is a bulleted list of four items, each preceded by a diamond symbol. The first item discusses @RenderBody() and @RenderSection(). The second item discusses using @RenderBody() to define a placeholder. The third item discusses the limitation of using @RenderBody() only once in a layout. The bottom left corner shows the copyright notice "© Aptech Ltd.", the bottom center says "Consistent Styles and Layouts / Session 7", and the bottom right corner has the number "7". A large, semi-transparent watermark reading "For Aptech Centre Only" is diagonally across the slide.

- ◆ You can display unique content associated with a view using the following methods:
  - + `@RenderBody()`: Allows you to define a placeholder in the layout to insert the main content of the view.
  - + `@RenderSection()`: Allows you to display a section specified in the view.
- ◆ You can use the `@RenderBody()` method to define a placeholder in the layout that needs to be inserted in the main content of the view.
- ◆ While using the `@RenderBody()` method, you should remember that, you can use it only once in a layout.

Using slide 7 explain to students that they can display unique content associated with a view using the following methods:

- `@RenderBody( )`: Allows defining a placeholder in the layout to insert the main content of the view.
- `@RenderSection( )`: Allows displaying a section specified in the view.

Tell them that they can use the `@RenderBody( )` method to define a placeholder in the layout that needs to be inserted in the main content of the view. While using the `@RenderBody( )` method, they can use it only once in a layout.

## Slide 8

Understand custom layout.

### Custom Layout 4-6

- Following code snippet shows a layout named \_CustomLayout.cshtml file that uses the @RenderBody() method:

Snippet

```
<!DOCTYPE html>
<html lang="en">
<head>
<title> ASP.NET MVC Application</title>
</head>
<body>
<p>
    @RenderBody()
</p>
</body>
</html>
```

- In this code, the @RenderBody() method inside the <p> tag inserts the content of the view.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 8

In slide 8, you refer to the code displayed on the slide that shows a layout named \_CustomLayout.cshtml file which uses the @RenderBody( ) method.

Then tell them that in this code, the @RenderBody( ) method inside the <p> tag inserts the content of the view.

## Slide 9

Understand custom layout.

### Custom Layout 5-6

- ◆ You can use the `@RenderSection()` method to display a section specified in the view.
- ◆ A layout allows you to add one or more sections in it.
- ◆ Following code snippet shows the `_CustomLayout.cshtml` layout that uses the `@RenderSection()` method:

Snippet

```
<!DOCTYPE html>
<html>
<head>
<title>@ViewBag.Title</title>
</head>
<body>
<div id="main-content">@RenderBody()</div>
<div>@RenderSection("TestSection")</div>
</body>
</html>
```

- ◆ In this code, the `@RenderSection()` method is used to add a section.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 9

In slide 9, you ask the students that they can use the `@RenderSection()` method to display a section specified in the view. A layout allows them to add one or more sections in it.

Then you refer to the code displayed on the slide that shows the `_CustomLayout.cshtml` layout that uses the `@RenderSection()` method.

Explain them that in this code the `@RenderSection()` method is used to add a section.

## Slide 10

Understand custom layout.

### Custom Layout 6-6

- ♦ Sometimes, you may not want to use a specified section in some of the views in your application.
- ♦ In such situation, you can use an overloaded method of the `@RenderSection()` method in your layout.
- ♦ This overloaded method uses the required attribute with the false value.
- ♦ Following code snippet shows how to use an overloaded method of the `@RenderSection()` method:

Snippet

```
<div>@RenderSection("TestSection", required: false)</div>
```

- ♦ In this code, the `@RenderSection()` method indicates that a view that uses this layout can provide content for the `TestSection` section. As this is not mandatory, a view using this layout may not contain any section named `TestSection`.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 10

In slide 10, you explain to the students that sometimes, they may not want to use a specified section in some of the views in their application. In such situation, they can use an overloaded method of the `@RenderSection()` method in the layout. This overloaded method uses the required attribute with the false value.

Then you refer to the code displayed on the slide that shows how to use an overloaded method of the `@RenderSection()` method.

Then explain the students that in this code, the `@RenderSection()` method indicates that a view that uses this layout can provide content for the `TestSection` section. As this is not mandatory, a view using this layout may not contain any section named `TestSection`.

**Slide 11**

Understand specifying a layout for a view.

## Specifying a Layout for a View 1-2

- ◆ While developing an ASP.NET MVC Web application, you can also specify a layout for either a particular view or for all the views.
- ◆ You can specify a layout for a view by using the Layout property.
- ◆ Following code snippet shows specifying a layout for a view:

@{  
 Layout = "~/Views/Shared/CustomLayout.cshtml";  
}

Snippet

- ◆ In this code, the CustomLayout.cshtml file is specified as the layout for a view available in the application.
- ◆ You can also specify a particular layout for all the views available in your application by using the Layout property in the \_ViewStart.cshtml file.
- ◆ This file is available under the Views folder.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 11

Display slide 11 and explain to the students that while developing an ASP.NET MVC Web application, they can also specify a layout for either a particular view or for all the views. They can specify a layout for a view by using the Layout property.

Then refer to the code displayed on the slide that shows specifying a layout for a view.

Explain the students that in this code, the CustomLayout.cshtml file is specified as the layout for a view available in the application.

Mention that they can also specify a particular layout for all the views available in an application by using the Layout property in the \_ViewStart.cshtml file. This file is available under the Views folder.

## Slide 12

Understand specifying a layout for a view.

### Specifying a Layout for a View 2-2

- ◆ When you create an ASP.NET MVC Web application in Visual Studio 2013, the \_ViewStart.cshtml file is created by default under the Views folder.
- ◆ This file specifies the default layout to be used for the views available.
- ◆ Following code snippet shows the default layout in the \_ViewStart.cshtml file:

```
@{  
    Layout = "~/Views/Shared/_Layout.cshtml";  
}
```

Snippet

- ◆ In this code, the default layout is specified as \_Layout.cshtml.
- ◆ When you run the application, the code available in the \_ViewStart.cshtml file is executed first.
- ◆ Then, the code of the available views in the same directory or within a sub-directory is executed.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 12

In slide 12, explain to the students that when they create an ASP.NET MVC Web application in Visual Studio 2013, the \_ViewStart.cshtml file is created by default under the Views folder. This file specifies the default layout to be used for the views available.

Then refer to the code that shows the default layout in the \_ViewStart.cshtml file.

Then explain that in this code, the default layout is specified as \_Layout.cshtml. When they run the application, the code available in the \_ViewStart.cshtml file is executed first. Then, the code of the available views in the same directory or within a sub-directory is executed.

## Slide 13

Understand nested layout.

### Nested Layout

- ◆ While developing Web applications, sometimes you might need to maintain some standard features.
- ◆ In such scenarios, you can use nested layouts.
- ◆ A nested layout:
  - + Refers to a layout that is derived from a parent layout and is responsible for defining the structure of a page.
  - + Uses the @RenderBody() method only once.
  - + It can have multiple sections that can be rendered by using the @RenderSection() method.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 13

Use slide 13 to explain the students that while developing Web applications, sometimes they might need to maintain some standard features. In such scenarios, they can use nested layouts.

Tell them that a nested layout refers to a layout that is derived from a parent layout and is responsible for defining the structure of a page. It uses the @RenderBody() method only once. It can have multiple sections that can be rendered by using the @RenderSection() method.

## Slide 14

Understand implementing styles.

## Implementing Styles

- ◆ Besides using the layouts, in an ASP.NET Web application, you can also use styles that allow you to apply consistent formatting across all the views available.
- ◆ Styles are used to define a set of formatting options, which can be reused to format different HTML helpers on a single view or on multiple views.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 14

Use slide 14 to explain the students that besides using the layouts, in an ASP.NET Web application, they can also use styles that allows applying consistent formatting across all the views available.

Tell them that styles are used to define a set of formatting options, which can be reused to format different HTML helpers on a single view or on multiple views.

### Discussion:

Initiate a discussion by explaining the students that the look and feel of a view depends upon the appearance and arrangement of the HTML elements contained in it. Therefore, they need to format the contents of a view to make it look attractive. However, formatting each HTML helper individually is a tedious task and it fails to bring consistency in the appearance of all the views on the Web application. Conclude the discussion by explaining that as a solution, they can use styles that enable a programmer to apply consistent formatting across the entire Web application.

## Slide 15

Understand CSS files.

### CSS Files 1-8

- ◆ A .css file in an ASP.NET Web application allows you to define a style that needs to be applied in the application.
- ◆ Whenever you create an ASP.NET MVC Web application in Visual Studio 2013, a .css file named Site.css is automatically created under the Content folder.
- ◆ Following code snippet shows using style definitions in the Site.css file:

```
H2 {  
    font-weight: bold;  
    font-size: medium;  
    color: red;  
    font-family: Times New Roman; }
```

Snippet

- ◆ This code contains a style definition that will be applied to all the <h2> tags in the views that references the test.css file.
- ◆ Here, <h2> tag acts as a selector that specifies the elements on which the specified style needs to be applied.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 15

In slide 15, explain the students that a .css file in an ASP.NET Web application allows defining a style that needs to be applied in the application. Whenever they create an ASP.NET MVC Web application in Visual Studio 2013, a .css file named Site.css is automatically created under the Content folder.

Then refer to the code that shows using style definitions in the Site.css file.

Explain that this code contains a style definition that will be applied to all the <h2> tags in the views that references the test.css file. Here, <h2> tag acts as a selector that specifies the elements on which the specified style needs to be applied.

## Slide 16

Understand CSS files.

The slide has a dark purple background with a faint graphic of a person working on a computer. At the top right, the title 'CSS Files 2-8' is displayed in a white, bold font. Below the title, there are two main bullet points:

- ◆ You can also define same style for more than one element by combining selectors as a single group.
- ◆ Following code snippet shows combining two elements in the Site.css file:

A blue button labeled 'Snippet' is positioned to the right of the second bullet point. Below the button, a code snippet is shown in a light gray box:

```
H1, H2 {COLOR: BLUE;}
```

Further down the slide, there are more bullet points:

- ◆ This code will display all the text that are specified inside the <h1> and <h2> elements in blue color.
- ◆ Besides applying styles using elements, you can also use the following types of CSS selectors:
  - + ID selector: Is used to identify an element that you need to style differently from the rest of the page. Each element on a view can have a unique Id, which contains a hash symbol (#) followed by the element Id.

At the bottom of the slide, there are three small pieces of text: '© Aptech Ltd.', 'Consistent Styles and Layouts / Session 7', and '16'.

In slide 16, tell them that they can also define same style for more than one element by combining selectors as a single group.

Then refer to the code that shows combining two elements in the Site.css file.

Next, explain the students that this code will display all the text that are specified inside the <h1> and <h2> elements in blue color.

Then tell them that besides applying styles using elements, they can also use the ID CSS selector. This selector is used to identify an element that they need to style differently from the rest of the page. Each element on a view can have a unique Id, which contains a hash symbol (#) followed by the element Id.

## Slide 17

Understand CSS files.

The slide has a dark purple background with a faint geometric pattern. At the top right, the title 'CSS Files 3-8' is displayed in a white, bold font. Below the title, there are two sections, each labeled 'Snippet' in a blue button-like box. The first snippet shows an HTML code block: <p id="uname">Mobile Phone Users</p>. The second snippet shows a CSS code block: #uname { color:green; font-size:20pt; font-weight:bold; }. To the left of these snippets, there are two bulleted lists. The first list, under the first snippet, states: 'Following code snippet specifies an id for the <p> tag:' followed by a bullet point: 'Now, you can define style separately for the element, whose Id is uname.' The second list, under the second snippet, states: 'Following code snippet shows defining style specifications to be applied to the element, whose Id is uname:' followed by a bullet point: 'This code selects the element with the id named, uname and applies the specified styles on it.' At the bottom left of the slide, the text '© Aptech Ltd.' is visible. In the bottom center, the text 'Consistent Styles and Layouts / Session 7' is displayed. On the far right, the number '17' indicates the slide number.

In slide 17, you refer to the first code that specifies an id for the <p> tag. Now, they can define style separately for the element, whose Id is uname.

Then you refer to the second code that shows defining style specifications to be applied to the element, whose Id is uname.

Explain them that this code selects the element with the id named, uname and applies the specified styles on it.

## Slide 18

Understand CSS files.

The slide has a dark purple background with a faint geometric pattern. At the top right, the title 'CSS Files 4-8' is displayed in white. Below the title, there are two bullet points:

- ◆ Class selector:
  - + Is used to specify styles for a group of elements.
  - + Allows applying a style on several elements in a view.
- ◆ Following code snippet shows how to use class selector to define styles:

A blue button labeled 'Snippet' is positioned to the right of the code block. The code block contains the following HTML:

```
<p class="red"> First paragraph </p>
<p class="red"> Second paragraph </p>
<p class="green"> Third paragraph </p>
<p class="green"> Fourth paragraph </p>
<p class="blue"> Fifth paragraph </p>
```

Below the code, another bullet point states: 'This code uses the class selector to define styles for multiple elements.'

At the bottom left of the slide, it says '© Aptech Ltd.' and at the bottom center 'Consistent Styles and Layouts / Session 7'. The bottom right corner shows the number '18'.

In slide 18, explain the students that class selector is used to specify styles for a group of elements. It allows applying a style on several elements in a view.

Then you refer to the code that shows how to use class selector to define styles.

Explain the students that this code uses the class selector to define styles for multiple elements.

## Slide 19

Understand CSS files.

The slide has a dark background with a geometric pattern. At the top right, the title 'CSS Files 5-8' is displayed in a maroon font. Below the title, there is a blue button-like element containing the word 'Snippet'. To the left of the snippet button, there is a bulleted list:

- Following code snippet shows applying different style to the five paragraphs:

```
.red
{
    color:red;
}
.green
{
    color:green;
}
.blue
{
    color:blue
}
```

- This code will display the first two paragraphs in red color, the next two paragraphs in green color, and the last paragraph is in blue color.

At the bottom left of the slide, there is a small copyright notice: '© Aptech Ltd.'. At the bottom center, it says 'Consistent Styles and Layouts / Session 7'. On the far right, the number '19' is visible.

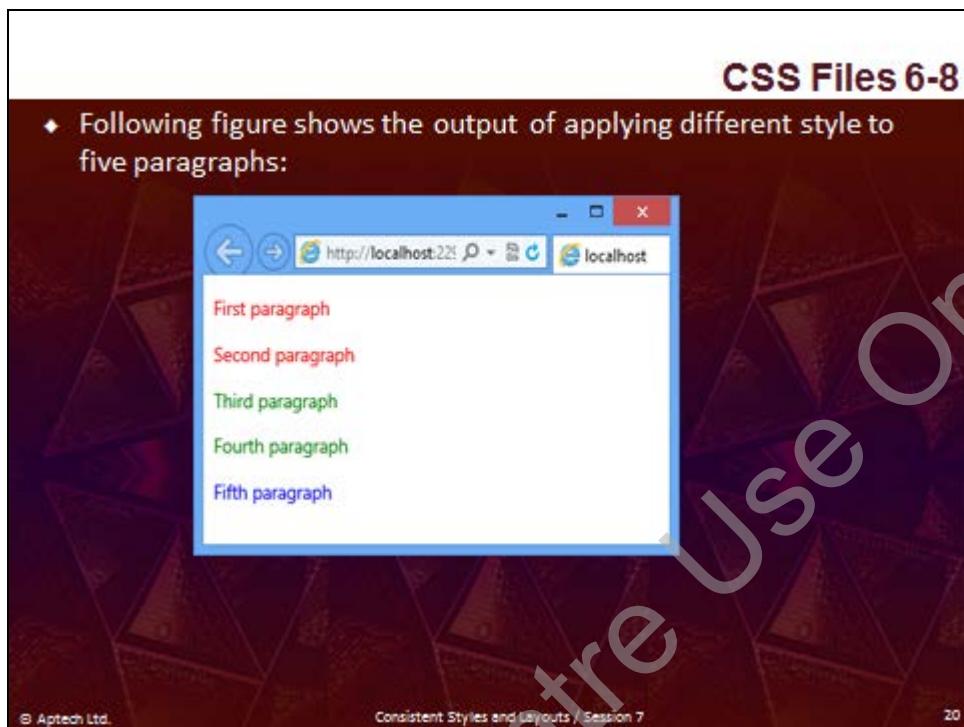
In slide 19, you refer to the code displayed on the slide that shows applying different style to the five paragraphs.

Explain the students that this code will display the first two paragraphs in red color, the next two paragraphs in green color, and the last paragraph is in blue color.

You can refer to the figure displayed on the slide 20, that shows the output of applying different style to five paragraphs.

## Slide 20

Understand CSS files.



Using slide 20, show the students about the result after applying various styles to paragraphs. Use the figure given on this slide to make it more clear to the students as to what is the effect of using CSS styles. Give them explanation about what is CSS and how it is helpful in enhancing the look and feel of the web pages. Also tell them that we can override almost any style using CSS.

**Slide 21**

Understand CSS files.

## CSS Files 7-8

- ◆ Universal selector is used to specify styles for all available elements in a particular area of a page or the whole page.
- ◆ To define universal selector, you should use the \* symbol.
- ◆ Following code snippet shows the <div> tag that contains some elements:

Snippet

```
<div class="Customers">
<p id="category1">Mobile users</p>
<p id="category2">Laptop users</p>
<p id="category3">Desktop users</p>
</div>
```

- ◆ This code creates a <div> tag that contains three <p> tags with data inside it.
- ◆ Now, to apply styles to all the elements in the preceding <div> tag, you can use the universal selector.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 21

Display slide 21 and explain the students that universal selector is used to specify styles for all available elements in a particular area of a page or the whole page. To define universal selector, they should use the \* symbol.

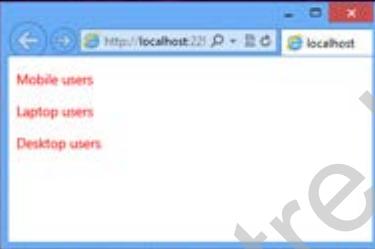
Then you refer to the code that shows the <div> tag that contains some elements.

## Slide 22

Understand CSS files.

### CSS Files 8-8

- Following code snippet shows applying styles using the universal selectors:  
**Snippet**

```
div.Customers *  
{  
color:red;  
}
```
- This code defines styles using the universal selector in the Site.css file.
- Following figure shows the output of applying styles using the universal selector:  


In slide 22, you refer to the code that shows applying styles using the universal selectors.

Explain the students that this code defines styles using the universal selector in the Site.css file.

Then refer to the figure displayed on the slide that shows the output of applying styles using the universal selector.

## Slide 23

Understand importing styles in views.

### Importing Styles in Views

- ◆ Once you have defined styles in the Site.css file, you need to link it with HTML documents on which the styles are to be implemented.
- ◆ For this, you can use the <link> tag that enables linking the views along with style sheet together.
- ◆ Following code snippet shows how to use the <link> tag:

**Snippet**

```
<link href("~/content/Site.css" rel="stylesheet" type="text/css" />
```
- ◆ In this code:
  - + The Site.css file is referred to by using the href attribute of the <link> tag.
  - + The value of the rel attribute specifies that the document will use a style sheet.
  - + Finally, the value of the type attribute specifies that the MIME type of the linked document is text/css.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 23

In slide 23, explain the students that once they have defined styles in the Site.css file, they need to link it with HTML documents on which the styles are to be implemented. For this, they can use the <link> tag that enables linking the views along with style sheet together.

Then refer to the code displayed on the slide that shows how to use the <link> tag.

Explain them that in this code the Site.css file is referred to by using the href attribute of the <link> tag. The value of the rel attribute specifies that the document will use a style sheet. Finally, the value of the type attribute specifies that the MIME type of the linked document is text/css.

## Slide 24

Understand adaptive user interface.

**Adaptive User Interface 1-11**

- ◆ At recent time it is possible to access Web application from various devices, such as tabs, laptops, or mobile phones.
- ◆ Due to the different screen sizes and browser capability, the page size and visual appearance of a Web application may change accordingly.
- ◆ Adaptive rendering enables automatic scaling down of a page and redrawing the page according to a device screen.
- ◆ You can create an adaptive UI in order to implement adaptive rendering using one of the following mechanisms:
  - + Viewport meta tag
  - + CSS media queries

© Aptech Ltd. Consistent Styles and Layouts / Session 7 24

In slide 24, explain the students that at recent time it is possible to access Web application from various devices, such as tabs, laptops, or mobile phones. Due to the different screen sizes and browser capability, the page size and visual appearance of a Web application may change accordingly. Adaptive rendering enables automatic scaling down of a page and redrawing the page according to a device screen.

Then tell them that they can create an adaptive UI in order to implement adaptive rendering using one of the following mechanisms:

- Viewport meta tag
- CSS media queries

## Slide 25

Understand adaptive user interface.

The slide has a dark purple background with a faint geometric pattern. At the top center, the title "Adaptive User Interface 2-11" is displayed in a bold, white, sans-serif font. Below the title, there is a bulleted list of four items, each preceded by a diamond symbol. The list discusses the rendering of mobile pages, the concept of a viewport, and how it is typically wider than the device's width. It also mentions controlling the size and scaling using the `<meta name="viewport" content="width=250">` tag. A blue rectangular button labeled "Snippet" is positioned to the right of the list. Below the button, a code block shows the `<meta name="viewport" content="width=250">` tag. Further down, another bullet point explains that setting the width to 250 pixels results in the page appearing on a 250-pixel wide canvas and being scaled properly within the screen area. The bottom of the slide contains copyright information: "© Aptech Ltd.", "Consistent Styles and Layouts / Session 7", and a small number "25".

**Adaptive User Interface 2-11**

- ◆ The browsers of a mobile device can render pages of a Web application and scale them in a manner so that they can properly appear in the screen area of the device.
- ◆ These pages are rendered in a virtual window known as a viewport.
- ◆ Viewport is typically wider than the actual width of a mobile device.
- ◆ You can control the size and scaling of pages of a Web application using the viewport `<meta>` tag in the head section of the view, as shown in the following code snippet:

`<meta name="viewport" content="width=250">`

- ◆ In this code, the width property is set to 250 pixels. As a result, the pages of the Web application will be appear on a canvas of width 250 pixels. It will also be scaled properly inside the screen area.

In slide 25, explain the students that the browsers of a mobile device can render pages of a Web application and scale them in a manner so that they can properly appear in the screen area of the device. These pages are rendered in a virtual window known as a viewport.

Tell them that viewport is typically wider than the actual width of a mobile device. They can control the size and scaling of pages of a Web application using the viewport `<meta>` tag in the head section of the view.

Refer to the code displayed on the slide.

Then explain them that in this code, the width property is set to 250 pixels. As a result, the pages of the Web application will be appear on a canvas of width 250 pixels. It will also be scaled properly inside the screen area.

## Slide 26

Understand adaptive user interface.

### Adaptive User Interface 3-11

- ◆ You can also use the device-width keyword to set the width of a viewport to the native screen size of the browser.
- ◆ This allows you to create the content, which can be adjusted according to the device specifications.
- ◆ Following code snippet shows how to use the device-width keyword:

Snippet

```
<meta name="viewport" content="width=device-width">
```

- ◆ In this code, the width property is set to the device-width value. This specifies the width of the device screen.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 26

In slide 26, explain the students that they can also use the device-width keyword to set the width of a viewport to the native screen size of the browser. This allows them to create the content, which can be adjusted according to the device specifications.

Then refer to the code that shows how to use the device-width keyword.

Then explain the students that in this code, the width property is set to the device-width value. This specifies the width of the device screen.

## Slide 27

Understand adaptive user interface.

### Adaptive User Interface 4-11

- ◆ You can use CSS media queries to enable your application to support different browsers and devices.
- ◆ To use CSS media queries you need to use the @media keyword.
- ◆ Using these queries you can apply different conditional CSS styles to support different device conditions or browser capabilities.
- ◆ Following code snippet shows the content of an Index.cshtml view:

Snippet

```
<html>
<head>
<title>Sample Article</title>
<link href="~/content/Site.css" rel="stylesheet" type="text/css" />
</head>
<body>
<div id="container">
<div id="article1">
<h1>Cricket</h1> Cricket is a bat-and-ball game played between two teams of 11 players each on a field at the center of which is a rectangular 22-yard long pitch. Each team takes its turn to bat, attempting to score runs, while the other team fields. Each turn is known as an innings. There are several formats of the cricket game.
</div>
```

© Aptech Ltd. Consistent Styles and Layouts / Session 7 27

In slide 27, explain to the students that they can use CSS media queries to enable an application to support different browsers and devices. To use CSS media queries they need to use the @media keyword. Using these queries they can apply different conditional CSS styles to support different device conditions or browser capabilities.

Then refer to the code displayed on slide 27 that shows the content of an Index.cshtml view.

**Slide 28**

Understand adaptive user interface.

## Adaptive User Interface 5-11

♦ Following code snippet shows the content of an Index.cshtml view:

Snippet

```
<div></div>
<div></div>
<div id="article2">
<h1>Football</h1>Football refers to a number of sports that involve,
to varying degrees, kicking a ball with the foot to score a goal. The
most popular of these sports worldwide is association football, more
commonly known as just "football" or "soccer". The variations of
football are known as football codes.
</div>
</div>
</body>
```

♦ This code creates the Index.cshtml view that contains sample articles on two topics.  
♦ Once you have created the Index.cshtml view, then, you can apply styles in the Site.css file to design the Index.cshtml view.

© Aptech Ltd.      Consistent Styles and Layouts / Session 7      28

Display slide 28 and refer to the code that shows the content of an Index.cshtml view.

Explain the students that this code creates the Index.cshtml view that contains sample articles on two topics. Once they have created the Index.cshtml view, then, they can apply styles in the Site.css file to design the Index.cshtml view.

**Slide 29**

Understand adaptive user interface.

## Adaptive User Interface 6-11

Following code snippet shows applying styles to design the Index.cshtml view:

Snippet

```
h1 {  
    text-align: center;  
    color:blue;  
}  
  
#container {  
    float: left;  
    width: 830px;  
}  
  
#article1 {  
    float: right;  
    width: 410px;  
    background: lightblue;  
}  
  
#article2 {  
    float: left;  
    width: 410px;  
    background: lightblue;  
}
```

© Aptech Ltd. Consistent Styles and Layouts / Session 7 29

Display slide 29 and refer to the code that shows applying styles to design the Index.cshtml view.

## Slide 30

Understand adaptive user interface.

### Adaptive User Interface 7-11

- ◆ The preceding code applies styles in the Site.css file to design the look and feel of the Index.cshtml view.
- ◆ Following figure shows the Index.cshtml view when access it on a browser:

The screenshot shows a web browser window with two cards displayed side-by-side. The left card is titled 'Football' and contains text about the sport. The right card is titled 'Cricket' and contains text about the sport. The browser address bar shows 'http://localhost:2290/'. The bottom of the slide includes copyright information: '© Aptech Ltd.' and 'Consistent Styles and Layouts / Session 7'.

In slide 30, refer to the code displayed on slide 29 and explain that the preceding code applies styles in the Site.css file to design the look and feel of the Index.cshtml view.

Then refer to the figure displayed on slide 30 that shows the Index.cshtml view when access it on a browser.

## Slide 31

Understand adaptive user interface.

The slide has a dark background with a geometric pattern. At the top center, the title "Adaptive User Interface 8-11" is displayed in a maroon font. Below the title, there are three bullet points in white:

- ◆ When you reduce the dimensions of the browser window, the look and feel of the view is affected.
- ◆ As a solution, you can use media queries in the Site.css file to add styles that allow reducing the dimensions of a view.
- ◆ Following code shows how to use media queries:

A blue button labeled "Snippet" is positioned to the right of the bullet points. Below the button, the CSS code is shown in white:

```
@media screen and (max-width:700px) {  
    #container {  
        float: left;  
        width: 500px; }  
    #article1 {  
        float: left;  
        width: 500px;  
        background: lightblue; }  
    #article2 {  
        float: none;  
        width: 500px;  
        background: lightblue; }  
}
```

At the bottom of the slide, there are three small pieces of text: "© Aptech Ltd.", "Consistent Styles and Layouts / Session 7", and "31".

In slide 31, explain to the students that when they reduce the dimensions of the browser window, the look and feel of the view is affected. As a solution, they can use media queries in the Site.css file to add styles that allow reducing the dimensions of a view.

Then you refer to the code that shows how to use media queries.

## Slides 32 and 33

Understand adaptive user interface.

### Adaptive User Interface 9-11

◆ Following code shows how to use media queries:

```
@media screen and (max-width:600px) {  
    #container {  
        float: left;  
        width: 400px; }  
    #article1 {  
        float: left;  
        width: 400px;  
        background: lightblue;  
    }  
    #article2 {  
        float: none;  
        width: 400px;  
        background: lightblue;  
    }  
}
```

Snippet

© Aptech Ltd. Consistent Styles and Layouts / Session 7 32

## Adaptive User Interface 10-11

- ◆ This code defines two media queries by using the @media syntax:
  - ◆ The first query defines the styles for the screens that have maximum width of 700px.
  - ◆ The second query defines the style for the screens that have maximum width of 500px.
- ◆ When you reduce the width of the browser, the view will appear according to the styles specified for reduced screen width.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 33

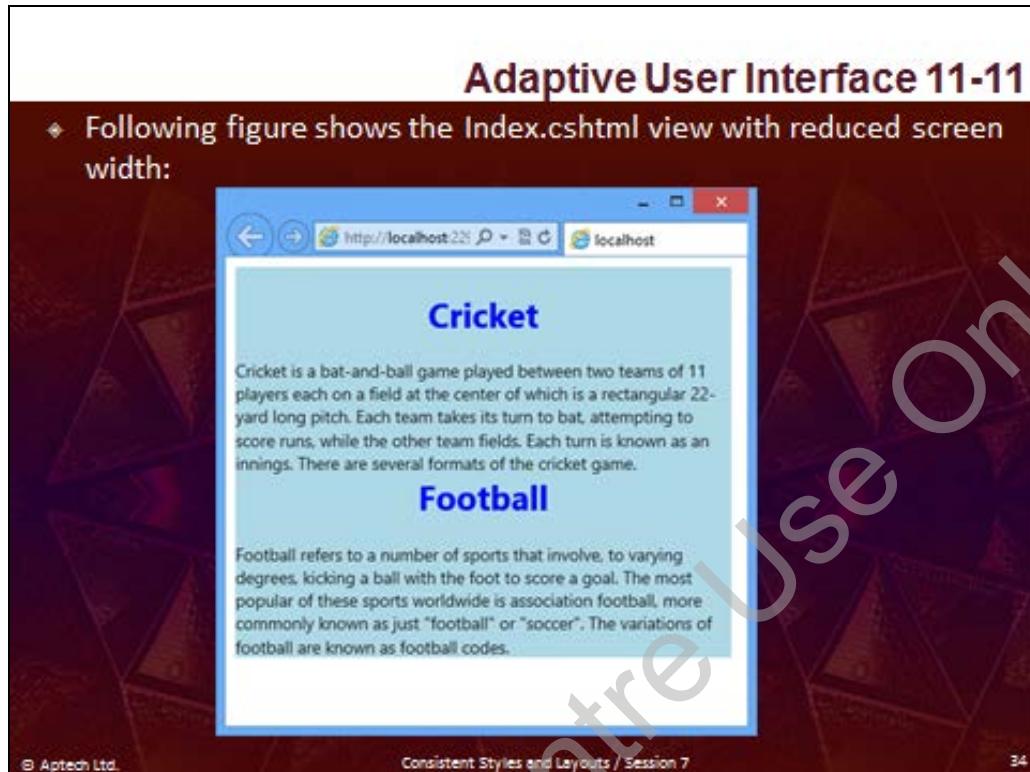
Use slide 33 to refer to the code displayed on slides 31 and 32 and explain the students that this code defines two media queries by using the @media syntax. The first query defines the styles for the screens that have maximum width of 700px and the second query defines the style for the screens that have maximum width of 500px.

Tell them that when they reduce the width of the browser, the view will appear according to the styles specified for reduced screen width.

Then you can refer to the figure displayed on the next slide that shows the Index.cshtml view with reduced screen width.

**Slide 34**

Understand adaptive user interface.



Using slide 34, show the students how the outputs of Index.cshtml file is shown using the reduced screen width.

## Slide 35

Let us summarize the session.

**Summary**

- ◆ ASP.NET MVC Framework allows you to use layouts to simplify the process of maintaining consistent look and feel in an application.
- ◆ The \_Layout.cshtml file represents the layout that you can apply to the views of an application.
- ◆ You can specify a layout for a view by using the Layout property.
- ◆ A nested layout refers to a layout that is derived from a parent layout and is responsible for defining the structure of a page.
- ◆ Styles are used to define a set of formatting options, which can be reused to format different HTML helpers on a single view or on multiple views.
- ◆ The Site.css file allows you to define styles for a particular view or for all the views available in the application.
- ◆ Adaptive rendering enables automatic scaling down of a page and redrawing the page according to a device screen.

© Aptech Ltd. Consistent Styles and Layouts / Session 7 35

In slide 35, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- ASP.NET MVC Framework allows you to use layouts to simplify the process of maintaining consistent look and feel in an application.
- The \_Layout.cshtml file represents the layout that you can apply to the views of an application.
- You can specify a layout for a view by using the Layout property.
- A nested layout refers to a layout that is derived from a parent layout and is responsible for defining the structure of a page.
- Styles are used to define a set of formatting options, which can be reused to format different HTML helpers on a single view or on multiple views.
- The Site.css file allows you to define styles for a particular view or for all the views available in the application.
- Adaptive rendering enables automatic scaling down of a page and redrawing the page according to a device screen.

### **7.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. How many times can the @RenderBody() method be invoked in a layout?
2. Which one of the following CSS selectors is prefixed with #?
3. Which options can be used to specify the layout to be used for all the views of the website?

# Session 8 -

## Responsive Pages

### **8.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **8.1.1 Objectives**

After the session, the learners will be able to:

- 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

#### **8.1.2 Teaching Skills**

To teach this session successfully, you must know about the layout. You should know about JavaScript, JQuery, and AJAX. You should also be aware of how work with AJAX.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

#### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students Slide 2 of the presentation. Tell them that they will be introduced to the technologies, such as JavaScript, JQuery, and AJAX. This session will also discuss about how to use these technologies in an ASP.NET MVC application.

## 8.2 In-Class Explanation

### Slide 3

Understand JavaScript.

The slide has a dark background with a faint geometric pattern. At the top right, the title 'Using JavaScript 1-3' is displayed in a light blue font. Below the title, the word 'JavaScript:' is in bold. A bulleted list follows, with the last item being a callout to a code snippet. The code snippet is highlighted with a blue box and labeled 'Snippet'. The code itself is standard HTML with a script tag containing a function that alerts 'Welcome user' when a button is clicked.

**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>
```

© Aptech Ltd.      Responsive Pages / Session 8      3

Use slide 3 to explain that JavaScript is a client-side scripting language that allows developing dynamic and interactive Web applications. This technology allows Web applications to respond to user requests without interacting with a Web server. It reduces the response time to deliver Web pages faster.

Then refer to the code displayed on slide 3 the slide that shows a JavaScript in an HTML page.

### Additional Information:

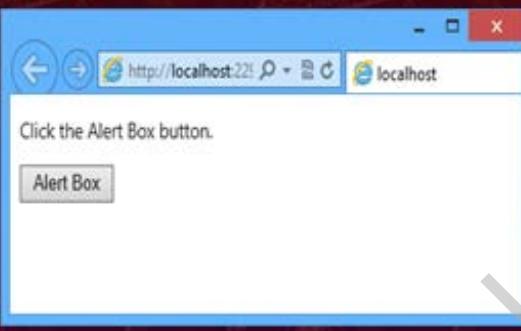
To develop a Web application that provides quick user interactions using JavaScript, users need to write a good amount of code because the JavaScript functions vary from browser to browser. Thus, to simplify the task of developing such Web applications with fewer lines of code, JavaScript provides various libraries, such as jQuery and jQuery UI.

**Slide 4**

Understand JavaScript.

### 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:



The screenshot shows a Microsoft Edge browser window with the URL <http://localhost:221>. A modal alert dialog box is displayed in the center of the screen. The dialog box has a white background and a blue border. Inside, there is a message: "Click the Alert Box button." Below the message is a single button labeled "Alert Box". The browser's address bar shows the URL, and the status bar at the bottom indicates "Responsive Pages / Session 8".

In slide 4, you refer to the code displayed on slide 3 and explain that the preceding code uses JavaScript to display a view that contains an Alert Box button.

Then you can refer to the figure displayed on slide 4 that shows the **Index.cshtml** view that displays the Alert Box button.

## Slide 5

Understand JavaScript.

### 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:

Display slide 5 and explain the students that when a user clicks the Alert Box button, the JavaScript code will execute on the browser, and display an alert box.

Then you refer to the figure displayed on slide 5 that shows the alert box that appears when a user clicks the Alert Box button.

## Slide 6

Understand unobtrusive JavaScript.

The slide has a dark background with a geometric pattern of triangles. At the top center, the title 'Unobtrusive JavaScript 1-3' is displayed in a white, bold, sans-serif font. Below the title is a bulleted list of six items, each preceded by a diamond-shaped bullet point. The list discusses the traditional use of JavaScript with HTML, the recommendation against mixing them, the solution provided by unobtrusive JavaScript, and its benefits. At the bottom left, there is small text that reads '© Aptech Ltd.' and 'Responsive Pages / Session 3'. On the right side, there is a small number '6'.

- ◆ 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.

In slide 6, explain the students that 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, they should always separate the presentation from the business logic code. As a solution, they can use unobtrusive JavaScript to separate the JavaScript code from HTML markups.

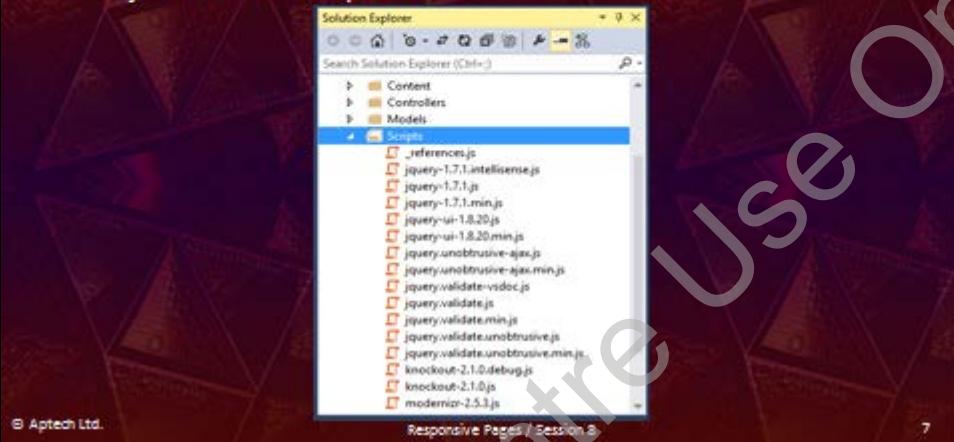
Tell them that unobtrusive JavaScript is an approach to use JavaScript separately from HTML markups in Web applications. In this approach, code is stored in a file with the .js extension.

## Slide 7

Understand unobtrusive JavaScript.

### 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:



The screenshot shows the Visual Studio Solution Explorer window. The 'Scripts' folder is expanded, displaying numerous JavaScript files including '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', and 'modernizr-2.5.3.js'. The title bar of the window reads 'Unobtrusive JavaScript 2-3'.

In slide 7, explain the students that when they use Visual Studio 2013 to create an ASP.NET MVC application, it automatically creates a Scripts folder that contains various JavaScript files.

Then you can refer to the figure that shows the Solution Explorer window that displays the .js files in the Scripts folder.

## Slide 8

Understand unobtrusive JavaScript.

### 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:  

```
<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.

Snippet

© Aptech Ltd.

Responsive Pages / Session 8

8

In slide 8, explain the students that once they have a JavaScript file with .js file extension, they can refer it in a view using the <script> element.

Then refer to the general syntax of using the <script> element in a view as displayed on slide 8 and explain it.

#### Additional Information:

Keeping all scripts in a separately downloadable file can give an application a performance boost because the browser can cache the script files locally. In addition, Unobtrusive JavaScript allows user to use a strategy known as progressive enhancement for an application. Progressive enhancement is a technique that allows users to deliver the content progressively. It uses a layered approach to deliver the content that allows everyone to access the basic content and functionality of a view, while providing an enhanced version of the view to users whose device or browser supports features, such as scripts and style sheets.

## Slide 9

Understand JavaScript libraries.

The screenshot shows a presentation slide with a dark background featuring a geometric pattern of triangles. At the top center, the title 'JavaScript Libraries 1-2' is displayed in a white, bold, sans-serif font. Below the title, there is a bulleted list of five points, each preceded by a blue diamond symbol. The list discusses the creation of a Script folder in Visual Studio 2013, referring to libraries in views, default library files, and the difference between main and minified versions. At the bottom left, there is a small copyright notice: '© Aptech Ltd.' and at the bottom right, 'Responsive Pages / Session 8' followed by a small number '9'.

- ◆ 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.

In slide 9, explain the students that when they create an ASP.NET MVC application in Visual Studio 2013, a Script folder is automatically created with several JavaScript libraries. They can refer one of those libraries in a view and then, use the functionalities provided by the library.

Then tell them that 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 other version is the minified version that 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.

### Additional Information:

Apart from the JavaScript library, the Scripts folder also contains jQuery plug-ins, such as jQuery UI and jQuery Validation. These plug-ins add additional capabilities to the core jQuery library. Additional plug-ins such as jQuery Mobile can also be downloaded and used in an application.

## Slide 10

Understand JavaScript libraries.

The slide has a dark purple background with a geometric pattern of triangles. At the top center, the title 'JavaScript Libraries 2-2' is displayed in a white, bold, sans-serif font. Below the title, there is a list of bullet points in white text:

- ◆ 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.

At the bottom left of the slide, there is a small copyright notice: '© Aptech Ltd.' In the bottom center, it says 'Responsive Pages / Session 8'. On the far right, the number '10' is visible.

In slide 10, explain to the students that the minified versions provide the same functionalities as their corresponding main versions. The advantage of using the minified versions is that it helps to reduce the amount of data to be packaged in an application.

Then you tell them that 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.

## Slide 11

Understand jQuery.

The slide has a dark purple background with a faint geometric pattern. At the top center, the title 'Introducing jQuery 1-5' is displayed in a white, bold, sans-serif font. Below the title, the word 'jQuery:' is in a smaller white font. A bulleted list follows, with each item preceded by a blue diamond symbol. The list details the features of jQuery, including its cross-browser compatibility, ability to manipulate HTML elements, and support for events and animation. A callout box labeled 'Snippet' in white text is positioned over a code block. The code block contains a single line of HTML-like syntax: '<script type="text/javascript" src="@Url.Content("~/Scripts/jquery-1.7.1.js")"></script>'. The bottom of the slide includes a footer with the text '© Aptech Ltd.', 'Responsive Pages / Session 8', and the number '11'.

## 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.

Display slide 11 and explain to the students that jQuery is a cross-browser JavaScript library that you can use to perform various functionalities, such as finding, traversing, and manipulating HTML elements. It enables animating HTML elements, handle events to make views of an application more dynamic and interactive.

Tell them that they can use the jQuery library in a view of an application by using the `<script>` element in the head section of the view.

Then refer to the code displayed on slide 11 the slide that shows using the `<script>` element.

Explain the students that this code uses the `<script>` element to use jQuery library in a view.

## Slide 12

Understand jQuery.

### 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.

© Aptech Ltd.      Responsive Pages / Session 8      12

Using slide 12, explain to the students that jQuery provides the document .ready( ) function to ensure that a view of the Web application has been fully loaded before executing the jQuery code.

Then refer to the code that shows the general syntax of using the document .ready( ) function and explain it.

## Slide 13

Understand jQuery.

The slide has a dark purple background with a faint geometric pattern. At the top center, the title "Introducing jQuery 3-5" is displayed in a white, sans-serif font. Below the title, there is a bulleted list of instructions:

- ◆ 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.

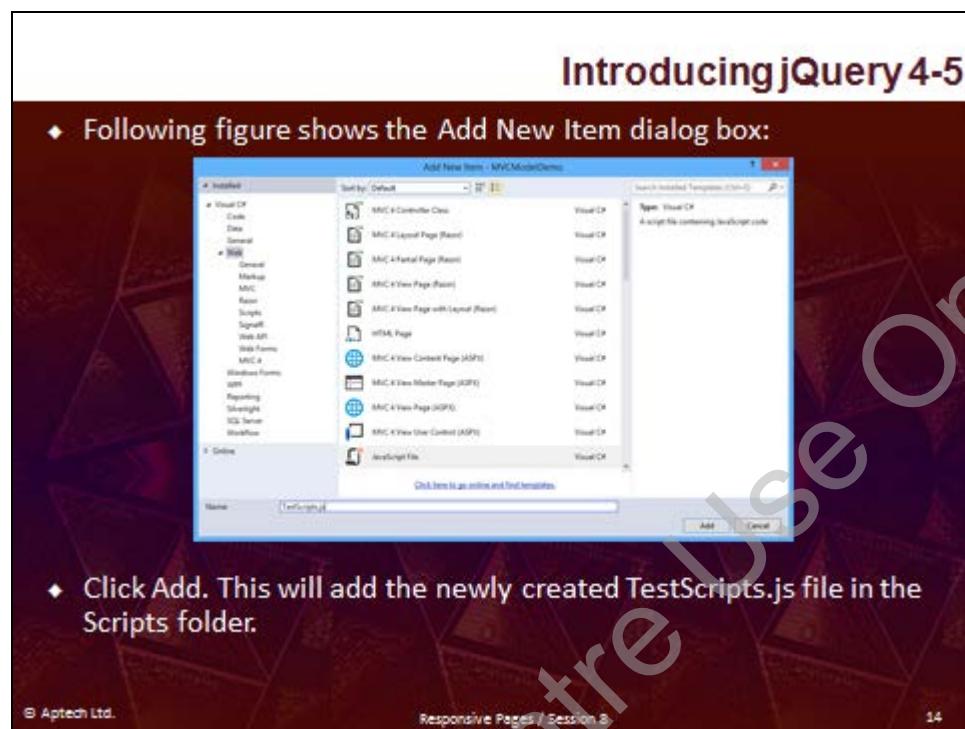
At the bottom of the slide, there are three small text elements: "© Aptech Ltd.", "Responsive Pages / Session 8", and "13". A large, diagonal watermark reading "For Aptech Centre Use Only" is overlaid across the slide content.

Use slide 13 to explain the students that sometimes, they might require adding customized client-side functionality in an application. In such situation, they can add your jQuery code in a file with .js extension and then, add it to the Scripts folder of the application directory.

Then explain them the steps to create a JavaScript file in Visual Studio 2013 as displayed on slide 13.

## Slide 14

Understand jQuery.



Use slide 14 to refer to the figure displayed on the slide that shows the Add New Item dialog box.

## Slide 15

Understand jQuery.

The slide has a dark purple background with a geometric pattern. At the top center, the title "Introducing jQuery 5-5" is displayed in a white, bold font. Below the title, there are two bulleted lists:

- 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:

A blue button labeled "Snippet" is positioned above a code block. The code block contains the following HTML:

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

Below the code block, another list of bullet points continues:

- 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.

At the bottom of the slide, there are three small pieces of text: "© Aptech Ltd.", "Responsive Pages / Session 8", and "15".

In slide 15, explain the students that once they have created and added the jQuery code in a .js file, they need to invoke it by using the <script> element.

Then refer to the code that shows referring to a customized TestScripts.js file.

Explain that this code uses the <script> element that refers to the TestScripts.js file.

Mention that apart from this, they can also use jQuery to select and manipulate HTML elements, handle events related to HTML elements, and add effects to HTML elements.

## Slide 16

Understand selecting and manipulating 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.

© Aptech Ltd.      Responsive Pages / Session 8      16

In slide 16, tell them that to select HTML elements and then manipulating them, jQuery allows to use some specific syntax.

Then refer to the code that shows the general syntax to select HTML elements and then perform the required actions on them.

**Slide 17**

Understand selecting and manipulating HTML elements.

## 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").

© Aptech Ltd.      Responsive Pages / Session 8      17

In slide 17, explain the students that they can use one of the following jQuery selectors, as per their 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, they can use \$( "h2" ).
- ID selector: Allows searching elements on the basis of their ids. For example, to search an element with the id, green, they 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 they can use \$( ".red" ).

## Slide 18

Understand handling events.

### 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
<code>\$(document).ready(function)</code>	Allows executing a function once a document completely loads.
<code>\$(selector).click(function)</code>	Allows executing a function on the click event of the selected elements.

© Aptech Ltd.      Responsive Pages / Session 8      18

In slide 18, explain the students that in jQuery, to handle events they can use functions or built-in event methods. They can use an event method in order to select an event and then, trigger a function when that event occurs.

Tell them that jQuery provides various event methods that they can use to handle events on HTML element.

Then you refer to the table that describes some of the event methods of jQuery.

**Slide 19**

Understand handling events.

<b>Handling Events 2-6</b>	
◆ Following table describes some of the event methods of jQuery:	
Event Method	Description
<code>\$(selector).dblclick(function)</code>	Allows executing a function on the double-click event of the selected elements.
<code>\$(selector).mouseover(function)</code>	Allows executing a function when the mouse pointer is moved over the selected elements.
<code>\$(selector).mouseout(function)</code>	Allows executing a function when the mouse pointer is moved out of the selected elements.
<code>\$(selector).keydown(function)</code>	Allows executing a function when a key is pressed on the selected elements.

Using slide 19, explain to the students the various event methods of JQuery.

## Slide 20

Understand handling events.

### 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>
```

© Aptech Ltd.      Responsive Pages / Session 3      20

Display slide 20 and ask them to consider a scenario, where they 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.

Then refer to the code displayed on slide 20 that shows the code of the `Index.cshtml` view.

**Slide 21**

Understand handling events.

## 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.

© Aptech Ltd.      Responsive Pages / Session 8      21

Display slide 21 and refer to the code displayed on slide 20 and explain that 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.

Tell them that after creating the Index.cshtml view, they need to add the jQuery code in the TestScripts.js file to change the text inside the <h1> element.

Then you refer to the code that shows the content of the TestScripts.js file.

Explain them that 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

## Slide 22

Understand handling events.



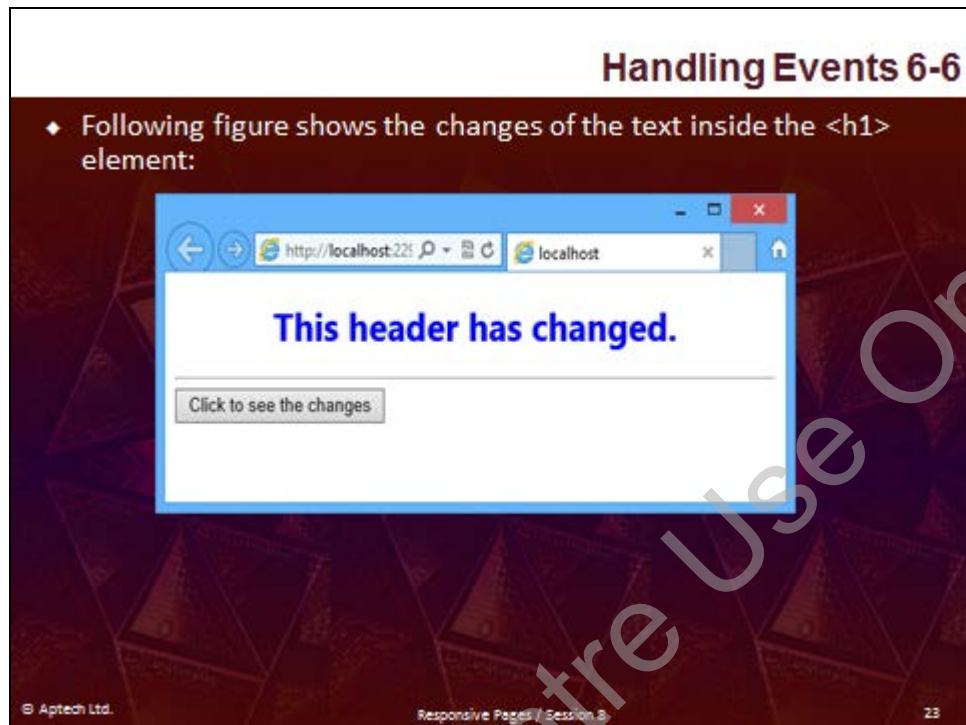
In slide 22, refer to the figure displayed on the slide that shows the Index.cshtml view, when a user access the application.

Explain the students that when a user click the Click to see the changes button the text inside the `<h1>` element that is "This is a header." should be changed.

Then you can refer to the figure displayed on the next slide that shows the changes of the text inside the `<h1>` element.

## Slide 23

Understand handling events.



Using slide 23, show the output as shown in the image given on the slide. Also point out the changes made in `<h1>` tag.

## Slide 24

Understand adding different effects.

### 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

© Aptech Ltd.      Responsive Pages / Session 8      24

In slide 24, explain the students that using jQuery, they 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 them to enrich the browsing experience of the user.

Then refer to the general syntax of using the hide( ) function to hide the selected elements and explain it as displayed on slide 24.

## Slide 25

Understand adding different effects.

### 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.

© Aptech Ltd.      Responsive Pages / Session 8      25

In slide 25, explain them that when they specify the values for the speed attribute as duration in milliseconds, they should ensure that this value should not enclosed within quotes. On the other hand, when they specify values for the speed attribute as slow or fast, this value should be within the quotes.

Then ask them to consider a scenario, where they want to slowly hide a text when a user clicks a button in a view. In such scenario, they can use the `hide( )` function.

## Slide 26

Understand adding different effects.

### 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.

© Aptech Ltd. | Responsive Pages / Session 3 | 26

In slide 26, you refer to the code that shows the Index.cshtml view that contains a Click to hide the text button.

Then explain the students that 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 they need to use the hide( ) function in the .js file.

## Slide 27

Understand adding different effects.

### 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.

© Aptech Ltd.      Responsive Pages / Session 8      27

In slide 27, you refer to the code displayed on slide that shows the jQuery code in the TestScripts.js file to hide a text.

Explain them that 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.

## Slide 28

Understand adding different effects.

The slide has a dark purple background with a faint geometric pattern of triangles. At the top, the title 'Adding Different Effects 5-7' is displayed in a maroon header bar. Below the title is a bulleted list of six points in white text. At the bottom left is the copyright notice '© Aptech Ltd.', at the bottom center is 'Responsive Pages / Session 3', and at the bottom right is the slide number '28'. A large, semi-transparent watermark reading 'For Aptech Centre Use Only' is diagonally across the slide.

- ◆ 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.

Display slide 28 and explain the students that they can use the fade effect on a selected element using jQuery. This effect enables gradually reducing the opacity of the selected elements. Some of the common fade function that they can use on selected elements are `fadeOut()` and `fadeIn()`.

Ask the students to consider a scenario, where they need to fade out the text “Text to be faded out” when a user clicks over it. Thereafter, they need 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.

**Slide 29**

Understand adding different effects.

## 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.

© Aptech Ltd.      Responsive Pages / Session 8      29

Display slide 29 and refer to the code that shows the Index.cshtml file that contains the text where the fade effect is to be applied and a Restore Text button.

Explain them that 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.

## Slide 30

Understand adding different 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:

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

Snippet

- ◆ 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.

© Aptech Ltd.      Responsive Pages / Session 8      30

In slide 30, explain the students that once they have created the view, they can define the jQuery code to implement fade effect in the TestScripts.js file.

Then refer to the code displayed on slide 30 that shows the jQuery code to perform fade in and fade out effects.

Explain them that 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.

## Slide 31

Understand JQuery UI.

The slide has a dark background with a red header bar containing the text "JQuery UI". Below the header, there is a bulleted list under the heading "◆ jQuery UI:". The list includes the following points:

- ◆ 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:

A blue box labeled "Snippet" contains the following code:

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

At the bottom of the slide, there are three small text elements: "© Aptech Ltd.", "Responsive Pages / Session 3", and "31".

In slide 31, explain to the students that jQuery UI is a set of features that allows developing highly interactive Web applications. It contains interface interactions, effects, widgets, and themes built on top of the jQuery JavaScript library.

Tell them that library provides a set of widgets that they can use to design UIs and apply various effects to the UI elements. It can be used in a view by including a reference to the jQuery UI script file in the view.

Then you refer to the code that shows adding reference to the jQuery UI script file in a view.

## Slide 32

Understand applying jQuery UI effects.

### 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>
```

© Aptech Ltd.      Responsive Pages / Session 8      32

Use slide 32 to explain the students that while applying style animations, jQuery UI provides a technique that they use to add, remove, or toggle class for the HTML elements.

Then refer to the code displayed on slide 32 that shows the Index.cshtml view that contains three buttons.

**Slide 33**

Understand applying jQuery UI effects.

## 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 : 250px;  
    height: 250px;  
    background-color:purple;  
}
```

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

© Aptech Ltd.      Responsive Pages / Session 8      33

Use slide 33 to refer to the code displayed on slide 32 and explain the students that 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 stylesheet of the application named Site.css.

Then you can refer to the code displayed on slide 33 that shows the defined styles in the Site.css file.

**Slide 34**

Understand applying jQuery UI effects.

### 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);
    });
});
```

© Aptech Ltd.      Responsive Pages / Session 3      34

Use slide 34 to explain the students that once they define the style in the Site.css file, they can define jQuery code to perform these effects in the script file named, TestScripts.js.

Then you can refer to the code displayed on slide 34 that shows the TestScripts.js that contains jQuery code to perform these effects.

**Note:**

Tell them that if the view uses the default \_Layout.cshtml layout included in most ASP.NET MVC project templates, they need to add the script elements within the scripts section.

**Slide 35**

Let us learn how to apply jQuery Effects.

**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.

© Aptech Ltd.      Responsive Pages / Session 8      35

Use slide 35 to refer the code displayed on slide 34 and explain the students that 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.

**Slide 36**

Understand AJAX.

## 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.

© Aptech Ltd.      Responsive Pages / Session 2      36

Use slide 36 to explain the students that AJAX is a Web development technique that allows making requests to the server in the background using client-side code. It enables them to update a view without reloading it completely.

Tell them that it also improves the performance of an application and the user experience. To understand the concept of AJAX, they should understand about asynchronous communication.

Then explain them that Asynchronous communication is the ability of a Web application to send multiple requests and receive responses from the server simultaneously. It enables them to work on the application without being affected by the responses received from the server.

### Slide 37

Understand AJAX helpers.

The slide has a dark purple background with a faint geometric pattern of triangles. At the top right, the title 'AJAX Helpers 1-2' is displayed in a white, bold, sans-serif font. Below the title is a list of eight bullet points, also in a white sans-serif font. The footer of the slide contains three small pieces of text: '© Aptech Ltd.', 'Responsive Pages / Session 8', and '37'.

**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.

© Aptech Ltd.      Responsive Pages / Session 8      37

Use slide 37 to explain the students that ASP.NET MVC application also uses a set of AJAX helpers. These helpers enable creating forms and links that point to controller actions. The only difference is that AJAX helpers behave asynchronously.

Tell them that while using AJAX helpers, they should not explicitly write JavaScript code make the asynchrony work. To make use of it, they should ensure that the jquery.unobtrusive-ajax script file is present in the Scripts folder of an application directory.

Mention that however, if they want to include the file manually, they need to use the <script> element.

**Slide 38**

Let us learn about AJAX Helpers.

## 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/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.

© Aptech Ltd.      Responsive Pages / Session 8      38

Use slide 38 to refer to the code that shows adding reference to the jquery.unobtrusive-ajax script file manually in the default \_Layout.cshtml file.

Explain them that this code adds reference to the jquery.unobtrusive-ajax script file in the default \_Layout.cshtml file.

**Slide 39**

Understand Unobtrusive AJAX.

### 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.

© Aptech Ltd.      Responsive Pages / Session 2      39

Use slide 39 to explain the students that the ASP.NET MVC Framework provides built-in support for unobtrusive AJAX. It enables them to use AJAX helpers to define AJAX features instead of writing code in views.

Tell them that to enable unobtrusive AJAX feature in an application, they need to configure the Web.config file.

Then you explain them that to enable unobtrusive AJAX feature in the Web.config file, they need to set the UnobtrusiveJavaScriptEnabled property to true under the <appSettings> element of the Web.config.

**Slide 40**

Let us learn about Unobtrusive AJAX

### Unobtrusive AJAX 2-3

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

<appSettings>

        <add key="webpages:Enabled" value="false" />

        <add key="UnobtrusiveJavaScriptEnabled" value="true" />

    </appSettings>

Snippet

- ◆ 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.

© Aptech Ltd.      Responsive Pages / Session 2      40

In slide 40, you refer to the code displayed on the slide that shows enabling the unobtrusive AJAX feature in the Web.config file.

Then you explain them that 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, they need to add references to the jQuery JavaScript libraries that implement the unobtrusive AJAX functionality.

**Slide 41**

Let us understand how to apply jQuery UI effects.

### 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.

© Aptech Ltd.      Responsive Pages / Session 3      41

Use slide 41 to refer to the code displayed on the slide that shows adding references to jQuery JavaScript libraries in a view.

Explain them that 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.

**Slide 42**

Understand working with AJAX.

**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.

© Aptech Ltd.      Responsive Pages / Session 8      42

In slide 42, explain the students that 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.

Tell them that 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.

**Slide 43**

Understand AJAX ActionLinks.

### 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>
```

© Aptech Ltd.      Responsive Pages / Session 8      43

Use slide 43 to explain the students that similar to the HTML helper methods, most of the methods of AJAX property are extension methods. The Ajax.ActionLink( ) helper method enables them to create an anchor tag with asynchronous behavior.

Then refer to the code that shows using the Ajax.ActionLink( ) method in the Index.cshtml view.

**Slide 44**

Let us understand the AJAX ActionLinks.

The slide has a dark red background with a faint geometric pattern of triangles. The title 'AJAX ActionLinks 2-2' is at the top right. Below it is a bulleted list:

- ◆ 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.

At the bottom left is the text '© Aptech Ltd.', in the center is 'Responsive Pages / Session 2', and at the bottom right is the number '44'.

Use slide 44 to refer to the code displayed on slide 43 and explain them that 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.

**Slides 45 to 48**

Understand AJAX Forms.

The slide has a dark red background with a faint watermark of a pyramid pattern. At the top right, the title 'AJAX Forms 1-4' is displayed in a white, bold font. Below the title, there is a list of four bullet points:

- ◆ 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.

At the bottom left of the slide, the text '© Aptech Ltd.' is visible. At the bottom center, it says 'Responsive Pages / Session 8'. On the far right, the number '45' is present.

Display slide 45 and ask the students to consider a scenario, where they need to enable a user to search for product. They want that 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, they 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" />
    
}
```

© Aptech Ltd.      Responsive Pages / Session 8      46

Use slide 46 to refer to the code displayed on the slide that shows using an asynchronous form element.

## 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.

© Aptech Ltd.      Responsive Pages / Session 8      47

Use slide 47 to refer to the code displayed on slide 46 and explain them that 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 ActionResult ProductSearch(string q)
{
    ProductDBContext db = 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.

© Aptech Ltd.      Responsive Pages / Session 8      48

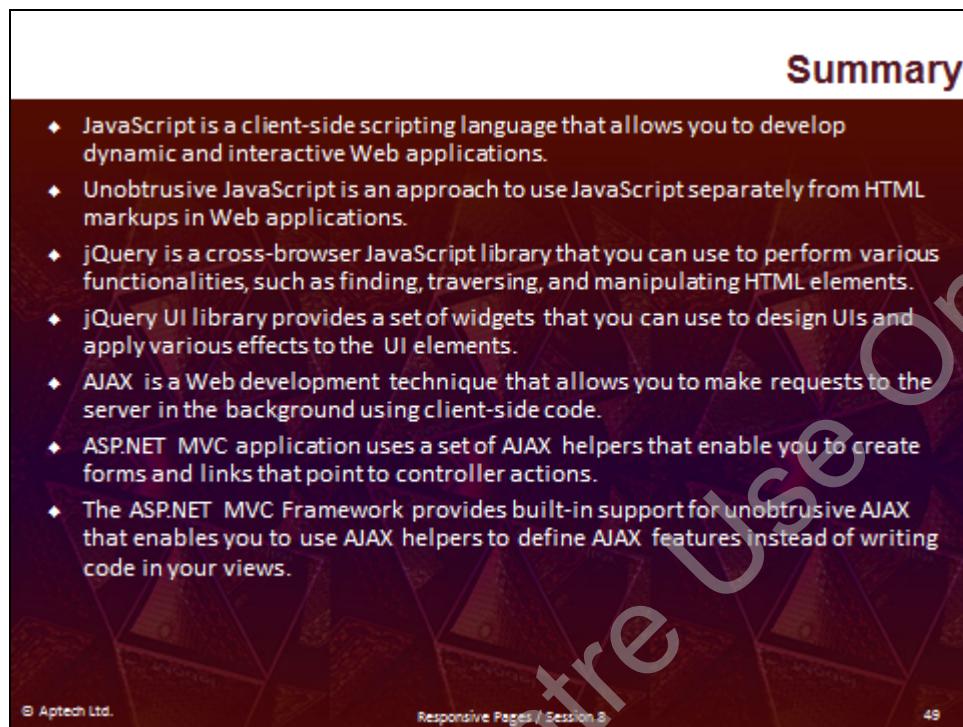
Use slide 48 to explain the students that when the user clicks the Submit button, the browser sends an asynchronous GET request to the ProductSearch() action method of the Product controller.

Then refer to the code displayed on slide 48 that shows defining the ProductSearch() action method.

Explain them that 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 search results element on the view.

## Slide 49

Let us summarize the session.



The slide has a dark purple background with a faint geometric pattern of triangles. At the top right, the word "Summary" is written in a white, bold, sans-serif font. Below it is a bulleted list of nine points, each preceded by a diamond symbol. The list covers various topics related to web development, including JavaScript, Unobtrusive JavaScript, jQuery, jQuery UI, AJAX, ASP.NET MVC, and the ASP.NET Framework's support for unobtrusive AJAX. At the bottom left, there is a small watermark-like text "© Aptech Ltd." and at the bottom center, another watermark-like text "Responsive Pages / Session 8". The bottom right corner shows the number "49".

- ◆ 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.

In slide 49, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- 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.

### **8.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. Which option will you use to search the HTML elements on the basis of their names?
2. Which jQuery UI functions allows you to style UI elements by adding and removing a class attribute to the UI elements alternatively?
3. Which event method is used to execute a function after the document has finished loading?

## Session 9 -

# State Management and Optimization

### **9.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **9.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe state management
- Explain and describe how to use cookies and application and session state
- Explain how to use caches to improve performance of an application
- Explain the process of bundling and minification

#### **9.1.2 Teaching Skills**

To teach this session successfully, you must know about the layout. You should know about state management and cookies in an ASP.NET MVC application. You should know how to use caches to improve performance of an application. You should also be aware of bundling and minification process.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

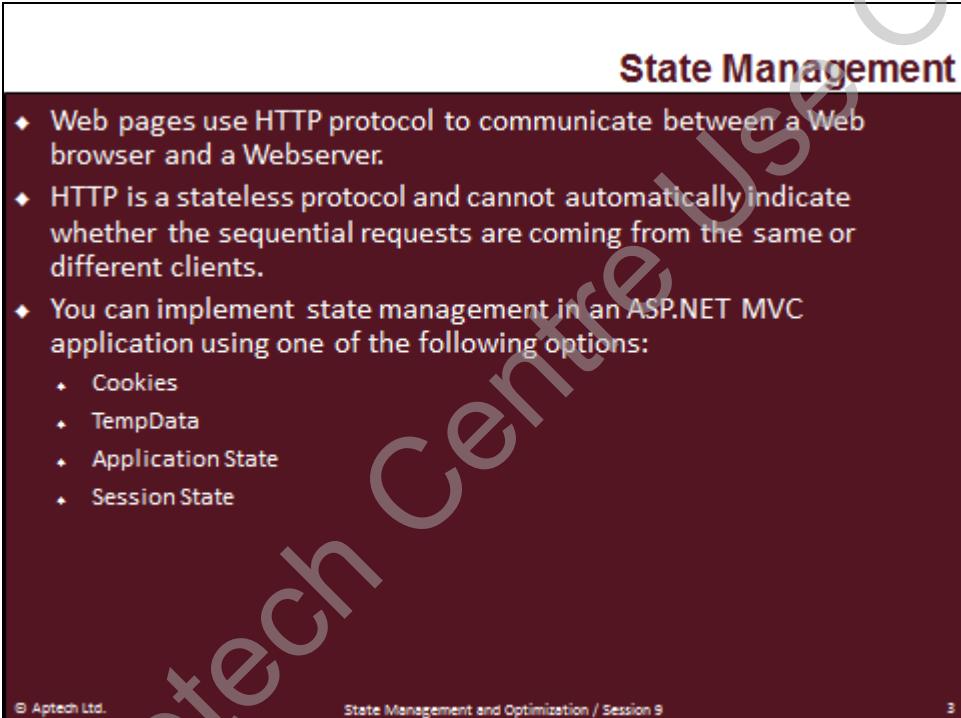
## Overview of the Session

Give the students a brief overview of the current session in the form of session objectives. Show the students Slide 2 of the presentation. Tell them that they will be introduced to state management and cookies. They will learn about how to use caches to improve performance of an application. This session will also discuss about the process of bundling and minification.

## 9.2 Introduction

### Slide 3

Understand State Management.



The slide has a dark red header bar with the title "State Management" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains four bullet points:

- ◆ Web pages use HTTP protocol to communicate between a Web browser and a Webserver.
- ◆ HTTP is a stateless protocol and cannot automatically indicate whether the sequential requests are coming from the same or different clients.
- ◆ You can implement state management in an ASP.NET MVC application using one of the following options:
  - Cookies
  - TempData
  - Application State
  - Session State

At the bottom of the slide, there is a footer bar with the text "© Aptech Ltd.", "State Management and Optimization / Session 9", and the number "3".

Use slide 3 to explain that Web pages use HTTP protocol to communicate between a Web browser and a Webserver. HTTP is a stateless protocol and cannot automatically indicate whether the sequential requests are coming from the same or different clients.

Then explain them that they can implement state management in an ASP.NET MVC application using one of the following options:

- Cookies
- TempData
- Application State
- Session State

**Slides 4 to 8**

Understand cookies.

## Cookies 1-5

- ◆ Cookies are used to store small pieces of information related to a user's computer, such as its IP address, browser type, operating system, and Web pages last visited.
- ◆ The purpose of storing this information is to offer a personalized experience to the user.
- ◆ Cookies are sent to a client computer along with the page output.
- ◆ These cookies are stored on the client's computer.
- ◆ When a browser requests the same page the next time, it sends the cookie along with the request information.
- ◆ The Web server reads the cookie and extracts its value.
- ◆ It then, process the Web page according to the information contained in the cookie and renders it on the Web browser.

© Aptech Ltd. State Management and Optimization / Session 9 4

Display slide 4 and explain the students that cookies are used to store small pieces of information related to a user's computer, such as its IP address, browser type, operating system, and Web pages last visited. The purpose of storing this information is to offer a personalized experience to the user. Cookies are sent to a client computer along with the page output. These cookies are stored on the client's computer.

Tell them that when a browser requests the same page the next time, it sends the cookie along with the request information. The Web server reads the cookie and extracts its value. It then, process the Web page according to the information contained in the cookie and renders it on the Web browser.

**Discussion:**

Initiate a discussion by explaining the advantages and disadvantages of cookies.

Advantages:

- A cookie is stored on the client computer and can be read by the server after a request for a page is posted.
- A cookie is a text-based data structure that contains key-value pairs. Therefore, it is easy to create and manipulate cookies.
- A cookie can expire when the browser session ends or can exist for a specified time on the client computer.

Disadvantages:

- Cookies that are stored on the client computer have a limited size. Therefore, you cannot store a large amount of data in a cookie.
- A user can disable cookies to prevent them from being stored on the hard disk of the computer. If a user denies permission for cookies, an ASP.NET Web application cannot store cookies on the client computer.
- Users can tamper with cookies because cookies are stored on the client computer.

## Cookies 2-5

- Following figure shows how cookies are transmitted between browser and application:

The diagram illustrates the transmission of a cookie between a Browser and an Application. It shows three main components: a blue vertical rectangle labeled 'Browser', a green vertical rectangle labeled 'Application', and a central blue box labeled 'Name = Joyee'. A red-bordered box labeled 'First request' points from the Browser to the Application. A blue-bordered box labeled 'Response with cookie' points from the Application back to the Browser, containing the 'Name = Joyee' box. A blue-bordered box labeled 'Subsequent request with cookie' points from the Browser to the Application, also containing the 'Name = Joyee' box.

- There are two types of cookies:
  - The first one is the session cookies that are stored in the browser's memory that are transmitted through the header during every request.
  - The other type of cookie is the persistent cookies that are stored in text files on a user's computer. This type of cookie is useful when you need to store information for a longtime.

© Aptech Ltd. State Management and Optimization/ Sessions 5

In slide 5, you refer to the figure displayed on the slide that shows how cookies are transmitted between browser and application.

Then tell them that there are two types of cookies. The first one is the session cookies that are stored in the browser's memory that are transmitted through the header during every request. The other type of cookie is the persistent cookies that are stored in text files on a user's computer. This type of cookie is useful when they need to store information for a longtime.

## Cookies 3-5

♦ Following code snippet shows how to create persistent cookies:

**Snippet**

```
Response.Cookies["UserName"].Value = "John";
Response.Cookies["UserName"].Expires = DateTime.Now.AddDays(2);
Response.Cookies["LastVisited"].Value = DateTime.Now.ToString();
Response.Cookies["LastVisited"].Expires = DateTime.Now.AddDays(2);
```

- ♦ This code creates the UserName and LastVisited cookies. The UserName cookie stores the name of a user and the LastVisited cookie stores the date and time when the user last visited the page. The expiry period for both the cookies is set as 2 days.
- ♦ Once you have a cookie, you can access the value of the cookie by using the built-in Request object.
- ♦ On the other hand to modify a cookie, you need to use the built-in Response object.

© Aptech Ltd. State Management and Optimization / Session 9 6

Display slide 6 and refer to the code displayed on the slide that shows how to create persistent cookies.

Explain them that this code creates the UserName and LastVisited cookies. The UserName cookie stores the name of a user and the LastVisited cookie stores the date and time when the user last visited the page. The expiry period for both the cookies is set as 2 days.

Tell them that once they have a cookie, they can access the value of the cookie by using the built-in Request object. On the other hand to modify a cookie, they need to use the built-in Response object.

## Cookies 4-5

- Following code snippet shows how to access a cookie that stores a single value:

**Snippet**

```
if (Request.Cookies["UserName"].Value != null)
{
    string Name = Request.Cookies["UserName"].Value;
}
```

- This code specifies that if the value of the UserName cookie is not null, then, the value is assigned to the Name string.

© Aptech Ltd. State Management and Optimization/ Session 5 7

In slide 7, you refer to the code that shows how to access a cookie that stores a single value.

Then explain the students that this code specifies that if the value of the UserName cookie is not null, then, the value is assigned to the Name string.

## Cookies 5-5

- ◆ You can also access the values from a cookie that store multiple values.
- ◆ Following code snippet shows how to access a cookie that stores multiple values:

Snippet

```
if (Request.Cookies["UserInfo"] != null)
{
    string Name = Request.Cookies["UserInfo"]["UserName"];
    string VisitedOn = Request.Cookies["UserInfo"]["LastVisited"];
}
```

- ◆ In this code:
  - ◆ The UserInfo cookie contains two sub-keys named, UserName and LastVisited.
  - ◆ If the value of the UserInfo cookie is not null, the value of the two sub-keys are assigned to the Name and VisitedOn strings respectively.

© Aptech Ltd.

State Management and Optimization / Session 9

8

In slide 8, explain the students that they can also access the values from a cookie that store multiple values.

Then refer to the code that shows how to access a cookie that stores multiple values.

Then explain them that in this code, the UserInfo cookie contains two sub-keys named, UserName and LastVisited. If the value of the UserInfo cookie is not null, the value of the two sub-keys are assigned to the Name and VisitedOn strings, respectively.

## Slides 9 to 11

Understand TempData.

The slide has a dark red header bar with the title "TempData 1-3" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following bullet points:

- ◆ **TempData:**
  - ◆ Is a dictionary that stores temporary data in the form of key-value pairs.
  - ◆ Allows store values between requests.
- ◆ You can add values to TempData by adding them to TempData collection.
- ◆ The data stored in TempData is available during the current and subsequent requests.
- ◆ TempData is useful in situations when you need to pass data to a view during a page redirect.

At the bottom of the slide, there is a footer bar with the following text: "© Aptech Ltd.", "State Management and Optimization / Session 9", and "9".

In slide 9, explain the students that TempData is a dictionary that stores temporary data in the form of key-value pairs. It allows store values between requests.

Tell them that they can add values to TempData by adding them to TempData collection. The data stored in TempData is available during the current and subsequent requests.

Mention that TempData is useful in situations when you need to pass data to a view during a page redirect.

### Discussion:

Initiate a discussion by explaining the advantages and disadvantages of TempData.

Advantages:

- TempData can be used to pass an error message to an error page easily.
- TempData is used to pass the data from the current request to the subsequent request in case of redirection.

**Disadvantages:**

- Its life is short and it lives only until the time the target view is fully loaded.
- It can be used to store only one time messages, such as error messages and validation messages.

### TempData 2-3

♦ Following code snippet shows the content of a controller class named HomeController:

**Snippet**

```
public class HomeController : Controller
{
    public ActionResult Index()
    {
        TempData["tempText"] = "Welcome to MVC";
        return RedirectToAction("Browse");
    }

    public ActionResult Browse()
    {
        return View();
    }
}
```

♦ In this code, when you send request for the Index() action method, the value, Welcome to MVC, is stored in the TempData key, named tempText. Hence, you are automatically redirected to the Browse() action method that will render a view.

© Aptech Ltd.      State Management and Optimization / Session 9      10

Display slide 10 and refer to the code that shows the content of a controller class named HomeController.

Explain that in this code, when they send request for the Index( ) action method, the value, Welcome to MVC, is stored in the TempData key, named tempText. Hence, they are automatically redirected to the Browse( ) action method that will render a view.

### TempData 3-3

- Following code snippet shows the content of the Browse() action method:

**Snippet**

```
@{var msg = TempData["tempText"] as String;}  
@msg
```

- The code renders a view that access the data stored for it in TempData and then display it to the user.

© Aptech Ltd. State Management and Optimization/ Session 5 11

Display slide 11 and refer to the code displayed on the slide that shows the content of the Browse( ) action method.

Explain the students that this code renders a view that access the data stored for it in TempData and then display it to the user.

**Slides 12 to 16**

Understand application state.

**Application State 1-5**

- ◆ Application state enables storing application-specific information as key-value pairs.
- ◆ When a user accesses any URL, application state is created for the first time.
- ◆ After that it stores the application-specific information.
- ◆ This information can be shared with all the pages and user sessions of the application.
- ◆ For this, you need to use the `HttpApplicationState` class.
- ◆ This class is accessed by using the `Application` property of the `HttpContext` object.

© Aptech Ltd.      State Management and Optimization / Session 9      12

In slide 12, explain to the students that application state enables storing application-specific information as key-value pairs. When a user accesses any URL, application state is created for the first time. After that it stores the application-specific information.

Tell them that this information can be shared with all the pages and user sessions of the application. For this, they need to use the `HttpApplicationState` class. This class is accessed by using the `Application` property of the `HttpContext` object.

**Note:**

Application state variables are global for an ASP.NET application. However, the values stored in the application state variables are accessible only from the code running within the context of the originating application. Other applications running on the system cannot access or modify the values.

## Application State 2-5

- ◆ Whenever any application events occur, the application state is initialized and manipulated.
- ◆ These application events include the following:
  - ◆ Application.Start: Event is raised when an application receives a request for a page for the first time, and when the server or the application is restarted. Event handler of this event contains the code for initializing the application variables.
  - ◆ Application.End: Event is raised when the server or the application is stopped or restarted. Event handler of this event contains the code to clear the resources that the application has already used.
  - ◆ Application.Error: Event is raised when an unhandled error occurs.
- ◆ The handlers for these three types of events are defined in the Global.asax file.
- ◆ You can write the code to manipulate the application state in one of these event handlers.

Use slide 13 to explain the students that whenever any application events occur, the application state is initialized and manipulated. These application events include the following:

- Application.Start: Event is raised when an application receives a request for a page for the first time, and when the server or the application is restarted. Event handler of this event contains the code for initializing the application variables.
- Application.End: Event is raised when the server or the application is stopped or restarted. Event handler of this event contains the code to clear the resources that the application has already used.
- Application.Error: Event is raised when an unhandled error occurs.

Tell them that the handlers for these three types of events are defined in the Global.asax file. They can write the code to manipulate the application state in one of these event handlers.

## Application State 3-5

- ◆ To store application-specific information in application state, you need to create variables and objects.
- ◆ Then, you can add these variables and objects to the application state.
- ◆ These variables and objects are accessible for any components of an ASP.NET MVC application.
- ◆ You need to write the code for initializing these application variables and objects inside the Application\_Start() function available in the Global.asax file.
- ◆ Following code snippet shows creating a variable named TestVariable and then, stores it in the application state:

Snippet

```
HttpContext.Application["TestVariable"] = "Welcome to MVC";
```

- ◆ This code creates a variable named, TestVariable. All the pages and the user sessions of the application can access the value stored in this variable.

Use slide 14 to explain the students that to store application-specific information in application state, they need to create variables and objects. Then, they can add these variables and objects to the application state. These variables and objects are accessible for any components of an ASP.NET MVC application.

Then tell them that they need to write the code for initializing these application variables and objects inside the Application\_Start( ) function available in the Global.asax file.

Next, refer to the code that shows creating a variable named TestVariable and then, stores it in the application state.

Explain them that this code creates a variable named, TestVariable. All the pages and the user sessions of the application can access the value stored in this variable.

## Application State 4-5

- ◆ When you run the application that includes the newly created variable, any pages of this application can retrieve the value of this variable.
- ◆ Following code snippet shows how to access the value of TestVariable:

Snippet

```
stringval = (string)HttpContext.Application["TestVariable"];
```

- ◆ This code access the value included in the TestVariable and then, stores it in a local variable named val.
- ◆ Once you have created and added a variable or object to an application state, you can also remove it from the application state using the Remove() method.

© Aptech Ltd. State Management and Optimization / Session 9 15

In slide 15, explain the students that when they run the application that includes the newly created variable, any pages of this application can retrieve the value of this variable.

Then refer to the code that shows how to access the value of TestVariable.

Explain that this code access the value included in the TestVariable and then, stores it in a local variable named val.

Tell them that once they have created and added a variable or object to an application state, they can also remove it from the application state using the Remove( ) method.

## Application State 5-5

♦ Following code snippet shows how to remove an application variable named, TestVariable, from the application state:

**Snippet**

```
HttpContext.Application.Remove("TestVariable");
```

♦ This code uses the Remove() method that will remove the variable named, TestVariable from the application state.

♦ You can also use the RemoveAll() method to remove all the available variables present in the application state.

♦ Following code snippet shows using the RemoveAll() method:

**Snippet**

```
HttpContext.Application.RemoveAll();
```

♦ This code will remove all the existing variables or objects from the application state.

© Aptech Ltd. State Management and Optimization / Session 9 16

In slide 16, you refer to the code that shows how to remove an application variable named, TestVariable, from the application state.

Explain that this code uses the Remove( ) method that will remove the variable named, TestVariable from the application state.

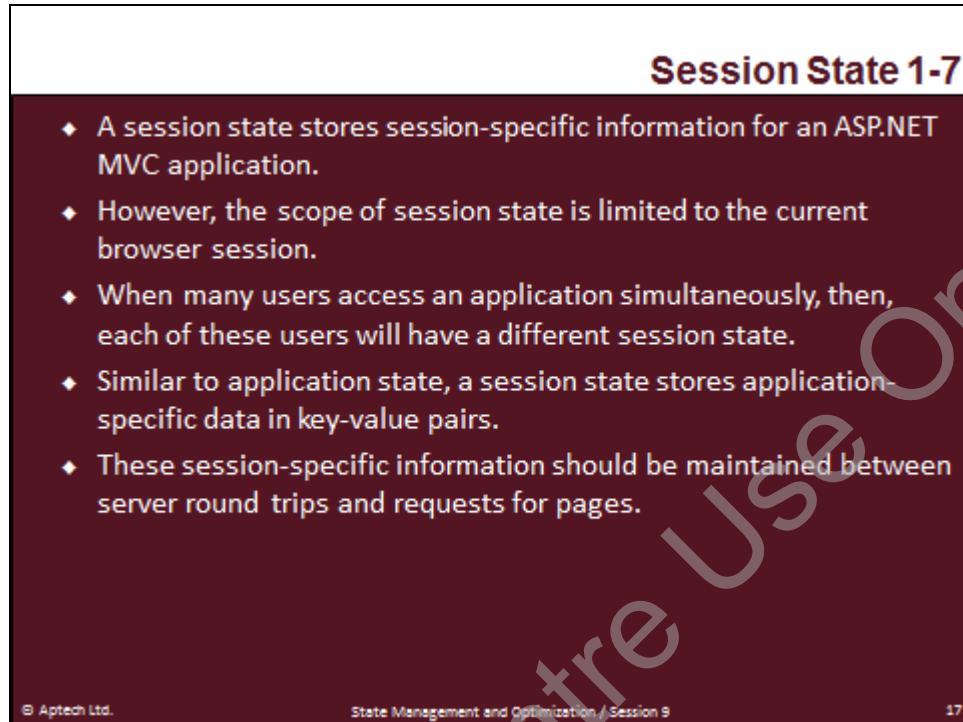
Tell them that they can also use the RemoveAll( ) method to remove all the available variables present in the application state.

Refer to the second code that shows using the RemoveAll( ) method.

Finally, explain them that this code will remove all the existing variables or objects from the application state.

## Slides 17 to 23

Understand session state.



The slide has a dark red header bar with the title "Session State 1-7" in white. The main content area is white with a dark red sidebar on the left. The sidebar contains the following text:

- ◆ A session state stores session-specific information for an ASP.NET MVC application.
- ◆ However, the scope of session state is limited to the current browser session.
- ◆ When many users access an application simultaneously, then, each of these users will have a different session state.
- ◆ Similar to application state, a session state stores application-specific data in key-value pairs.
- ◆ These session-specific information should be maintained between server round trips and requests for pages.

At the bottom of the slide, there is footer text: "© Aptech Ltd.", "State Management and Optimization / Session 9", and "17".

In slide 17, explain the students that a session state stores session-specific information for an ASP.NET MVC application. However, the scope of session state is limited to the current browser session. When many users access an application simultaneously, then, each of these users will have a different session state.

Tell them that similar to application state, a session state stores application-specific data in key-value pairs. These session-specific information should be maintained between server round trips and requests for pages.

### Additional Information:

The session state has a built-in support in ASP.NET. The built-in session state feature automatically performs the following actions:

- Identify and classify requests coming from a browser into a logical session on the server.
- Store session-specific data on the server.
- Raise session lifetime-related events, such as Session.Start and Session.End, which can be handled in the Global.asax file.
- Automatically release session data if a browser does not access an application within a specified timeout period.

## Session State 2-7

- ◆ An active session is identified and tracked by a unique session ID string containing American Standard Code for Information Interchange (ASCII) characters.
- ◆ The ASP.NET MVC Framework provides the SessionStateModule class that enables you to generate and obtain the session ID strings.
- ◆ Like application state, you can also store objects and variables in a session state.
- ◆ However, a session variable remains active for 20 minutes without any user interaction.
- ◆ You can use the same methods as application state, to add and access variable or objects from the session state.

Use slide 18 to explain the students that an active session is identified and tracked by a unique session ID string containing American Standard Code for Information Interchange (ASCII) characters. The ASP.NET MVC Framework provides the SessionStateModule class that enables them to generate and obtain the session ID strings.

Tell them that like application state, they can also store objects and variables in a session state. However, a session variable remains active for 20 minutes without any user interaction.

Mention that they can use the same methods as application state, to add and access variable or objects from the session state.

## Session State 3-7

♦ Following code snippet shows adding a variable named, TestVariable in a session state:

**Snippet**

```
Session["TestVariable"] = "Welcome to MVC Application";
```

♦ This code creates a variable named, TestVariable that contains 'Welcome to MVC Application' as its value.

♦ Following code snippet shows how to access the value of the newly created variable:

**Snippet**

```
stringval = (string) Session["TestVariable"];
```

♦ This code will access the value of TestVariable and stores it in the variable named, val.

© Aptech Ltd.      State Management and Optimization / Session 9      19

In slide 19, you refer to the first code displayed on the slide that shows adding a variable named, TestVariable in a session state.

Explain that this code creates a variable named, TestVariable that contains 'Welcome to MVC Application' as its value.

Then refer to the second code that shows how to access the value of the newly created variable.

Explain the students that this code will access the value of TestVariable and stores it in the variable named, val.

## Session State 4-7

- ◆ While using session state, you need to consider the following issues:
  - ◆ A variable or an object that is added to a session state remains until a user closes the browser window. By default, a variable or object is automatically removed from the session state after 20 minutes if a user does not request a page.
  - ◆ A variable or an object added to the session state is related to a particular user.
- ◆ Similar to application state, a session state can be globally accessed by the current user.
- ◆ A session state can be lost in case of the following situations:
  - ◆ User closes or restarts the browser.
  - ◆ User accesses the same Web page through a different browser window.
  - ◆ The session times out because of inactivity.
  - ◆ The Session.Abandon() method is called within the page code.

Display slide 20 and explain the students that while using session state, they need to consider the following issues:

- A variable or an object that is added to a session state remains until a user closes the browser window. By default, a variable or object is automatically removed from the session state after 20 minutes if a user does not request a page.
- A variable or an object added to the session state is related to a particular user.

Then tell them that similar to application state, a session state can be globally accessed by the current user.

Explain the students that a session state can be lost in case of the following situations:

- User closes or restarts the browser.
- User accesses the same Web page through a different browser window.
- The session times out because of inactivity.
- The Session.Abandon() method is called within the page code.

## Session State 5-7

- ◆ To remove an object or a variable that has been added to a session state, you can use the Remove() or RemoveAll() methods.
- ◆ While using session states in an application, you need to configure it in the Web.config file of the application.
- ◆ The Web.config file enables you to define advanced options, such as the timeout and the session state mode.
- ◆ Following code snippet shows the options that you can configure inside the <sessionState> element:

Snippet

```
<sessionState  
cookieless="UseCookies" cookieName="Test_SessionID"  
timeout="20"  
mode="InProc" />
```

- ◆ This code uses the cookieless, timeout, and mode session state attributes.

Display slide 21 and explain the students that to remove an object or a variable that has been added to a session state, they can use the Remove( ) or RemoveAll( ) methods.

While using session states in an application, they need to configure it in the Web.config file of the application. The Web.config file enables them to define advanced options, such as the timeout and the session state mode.

Then refer to the code displayed on slide 21 that shows the options that you can configure inside the <sessionState> element.

Explain them that this code uses the cookieless, timeout, and mode session state attributes.

## Session State 6-7

- ◆ You can use the cookieless attribute to specify whether a cookie is to be used or not.
- ◆ Following table lists the values that you can specify for the cookieless attribute:

Value	Description
UseCookies	Allows you to specify that cookies are always used.
UseUri	Allows you to specify that cookies are never used.
UseDeviceProfile	Allows you to specify that the application should check the Browser Capabilities object to identify whether to use cookies.
AutoDetect	Allows you to specify that cookies should be used if the browser supports them.

© Aptech Ltd.      State Management and Optimization / Session 9      22

In slide 22, explain that they can use the cookieless attribute to specify whether a cookie is to be used or not.

Then refer to the tables displayed on slide 22 that lists the values that they can specify for the cookieless attribute.

## Session State 7-7

- ◆ You can use the timeout attribute of session state to specify a time period in minutes.
- ◆ You can use the mode attribute to specify where the values of the session state is to be stored.
- ◆ You can use one of the following values of the mode attribute:
  - + Custom: Allows you to specify that a custom data store should be used to store the session-state data.
  - + InProc: Allows you to specify that the data is to be stored in the same process as the application. The Session.End event is raised only in this mode.
  - + Off: Allows you to specify that the session state is disabled.
  - + SQLServer: Allows you to specify that the session state is to be stored by using SQL Server database to store the state data.
  - + StateServer: Allows you to specify that the session state is to be stored in a service running on the server or a dedicated server.

In slide 23, you can use the timeout attribute of session state to specify a time period in minutes. They can use the mode attribute to specify where the values of the session state is to be stored.

Then tell them that they can use one of the following values of the mode attribute:

- Custom: Allows specifying that a custom data store should be used to store the session-state data.
- InProc: Allows specifying that the data is to be stored in the same process as the application. The Session.End event is raised only in this mode.
- Off: Allows specifying that the session state is disabled.
- SQLServer: Allows specifying that the session state is to be stored by using SQL Server database to store the state data.
- StateServer: Allows specifying that the session state is to be stored in a service running on the server or a dedicated server.

## Slide 24

Understand optimizing Web application performance.

### Optimizing Web Application Performance

- ◆ You can use a Web application for different purposes, such as discussion forums, online shopping, and social networking.
- ◆ To improve the performance of an application, you can reduce the number of interaction between the server and the database.
- ◆ The ASP.NET MVC Framework provides different techniques to improve the performance of an application.
- ◆ Following are two techniques that you can use to improve performance of an application:
  - ◆ First one is caching that you can use to optimize the performance of an application. Caching enables reducing the number of interactions between the server and database.
  - ◆ Second one is bundling and minification, which allows you to reduce the number of requests and the size of the requests between a client and a server.

© Aptech Ltd. State Management and Optimization / Session 9 24

In slide 24, explain the students that they can use a Web application for different purposes, such as discussion forums, online shopping, and social networking. To improve the performance of an application, they can reduce the number of interaction between the server and the database. The ASP.NET MVC Framework provides different techniques to improve the performance of an application.

Then tell them about the following two techniques that they can use to improve performance of an application:

- First one is caching that they can use to optimize the performance of an application. Caching enables reducing the number of interactions between the server and database.
- Second one is bundling and minification, which allows them to reduce the number of requests and the size of the requests between a client and a server.

## Slides 25 to 30

Understand output caching.

**Output Caching 1-6**

- ◆ Output caching allows you to cache the content returned by an action method.
- ◆ As a result, the same content does not need to be generated each time the action method is invoked.
- ◆ You need to consider the following factors, before implementing output caching:
  - ◆ Check whether output caching should be used in the application. If the content of the application changes frequently, output caching may not be useful.
  - ◆ Check the time duration for which the output has to be cached. This would depend on how frequently data associated with the output changes.
  - ◆ Check the location where you need to cache the output. You can cache the output on different locations, such as client machines and server.

© Aptech Ltd. State Management and Optimization / Session 9 25

In slide 25, explain them that output caching allows them to cache the content returned by an action method. As a result, the same content does not need to be generated each time the action method is invoked.

Then tell them that they need to consider the following factors, before implementing output caching:

- Check whether output caching should be used in the application. If the content of the application changes frequently, output caching may not be useful.
- Check the time duration for which the output has to be cached. This would depend on how frequently data associated with the output changes.
- Check the location where they need to cache the output. They can cache the output on different locations, such as client machines and server.

### Discussion:

Initiate a discussion by explaining the benefits of output caching:

- Reduced access time: Requests are processed by the cache itself and need not be sent to the database server, the access time of the Web page is reduced significantly.
- Less bandwidth required: When caching is implemented at the client computer and at the server, most of the requests and responses do not require data transfer over the network between the client and the server. This, in turn, reduces the bandwidth required for the communication.

- Lesser load on server: Caching avoids executing the same code repetitively to create the same result. Thus it reduces the CPU usage at the server, and in the process, lowers the load on the server.

## Output Caching 2-6

- ◆ In an ASP.NET MVC application, you can use output caching to cache the pages of the application for a specific time period.
- ◆ When a user request arrives for a same page within that time period, the application renders the cached page to the user instead of re-rendering the page.
- ◆ Following figure shows how ASP.NET MVC implements output caching:

```
graph LR; Browser[Browser] -- "First request" --> App[ASP.NET MVC]; App -- Response --> Browser; Browser -- "Subsequent request" --> App; App -- Cached response --> Browser;
```

© Aptech Ltd. State Management and Optimization / Session 9 26

In slide 26, explain that in an ASP.NET MVC application, they can use output caching to cache the pages of the application for a specific time period. When a user request arrives for a same page within that time period, the application renders the cached page to the user instead of re-rendering the page.

Then you can refer to the figure that shows how ASP.NET MVC implements output caching.

## Output Caching 3-6

- ◆ In an application, you can use caching at an action method.
- ◆ You can use the output cache attribute to an action method or the whole controller that needs to be cached. Thus, you can cache the result returned by the action method.
- ◆ Following code snippet shows how to add an output cache attribute to an action method:

Snippet

```
public class HomeController : Controller
{
    [OutputCache]
    public ActionResult Index()
    {
        //Code to perform some operations
    }
}
```

- ◆ In this code, the [OutputCache] attribute is added to the Index() action method of the Home controller.

In slide 27, explain the students that in an application, they can use caching at an action method. They can use the output cache attribute to an action method or the whole controller that needs to be cached. Thus, they can cache the result returned by the action method.

Then refer to the code displayed on slide 27 that shows how to add an output cache attribute to an action method.

Explain them that in this code, the [OutputCache] attribute is added to the Index( ) action method of the Home controller.

## Output Caching 4-6

- ◆ You can also use caching to cache the output for a specified duration.
- ◆ Caching the output for a small period of time is known as micro caching that you can use when the traffic on an application is high.
- ◆ Following code snippet shows specifying the duration and location of a cache attribute:

Snippet

```
public class HomeController : Controller
{
    [OutputCache(Duration=5)]
    public ActionResult Index()
    {
        ViewBag.Message = "This page is cashed for " + DateTime.Now;
        return View();
    }
}
```

- ◆ In this code, the output is cached for 5 seconds. This will refresh the page in every five seconds.

Display slide 28 and explain the students that they can also use caching to cache the output for a specified duration. Caching the output for a small period of time is known as micro caching that they can use when the traffic on an application is high.

Then refer to the code snippet that shows specifying the duration and location of a cache attribute.

Explain them that in this code, the output is cached for 5 seconds. This will refresh the page in every five seconds.

## Output Caching 5-6

- ◆ You can also define the location where you want the data to be cached.
- ◆ By convention, the data is cached in three locations when you use the [OutputCache] attribute.
- ◆ These locations can be the server, a proxy server, and the Web browser.
- ◆ However, you can specify a location of the output to be cached.
- ◆ For this, you need to modify the Location property of the [OutputCache] attribute.

Display slide 29 and explain that they can also define the location where they want the data to be cached. By convention, the data is cached in three locations when they use the [OutputCache] attribute. These locations can be the server, a proxy server, and the Web browser.

Tell them that however, they can specify a location of the output to be cached. For this, they need to modify the Location property of the [OutputCache] attribute.

## Output Caching 6-6

- ◆ You can use any one of the following values for the Location property:
  - ❖ Any: Specifies that the output cache should be stored on the browser, a proxy server, or the server where the request was processed.
  - ❖ Client: Specifies that the output cache should be stored on the browser where the request originated.
  - ❖ Downstream: Specifies that the output cache should be stored in any HTTP cache-capable device other than the original server.
  - ❖ Server: Specifies that the output cache should be stored on the server where the request was processed.
  - ❖ None: Specifies that the output cache is disabled for the requested page.

In slide 30, explain the students that once they can use any one of the following values for the Location property:

- Any: Specifies that the output cache should be stored on the browser, a proxy server, or the server where the request was processed.
- Client: Specifies that the output cache should be stored on the browser where the request originated.
- Downstream: Specifies that the output cache should be stored in any HTTP cache-capable device other than the original server.
- Server: Specifies that the output cache should be stored on the server where the request was processed.
- None: Specifies that the output cache is disabled for the requested page.

## Slides 31 and 32

Understand data caching.

### Data Caching 1-2

- ◆ In recent times, most of the Web applications contain dynamic data.
- ◆ These data needs to be retrieved from a database.
- ◆ So executing a database query for different users every time can reduce the performance of the application.
- ◆ To overcome such problems, you can use the data caching technique.
- ◆ To perform data caching in an ASP.NET MVC application, you can use the MemoryCache class.
- ◆ Following code snippet shows how to use the MemoryCache class to cache data:

Snippet

```
Customer dbUser = System.Runtime.Caching.MemoryCache.Default.  
AddOrGetExisting("UserData", getUser(),  
System.DateTime.Now.AddHours(1));
```

© Aptech Ltd. State Management and Optimization / Session 9 31

In slide 31, explain to the students that in recent times, most of the Web applications contain dynamic data. These data needs to be retrieved from a database. So executing a database query for different users every time can reduce the performance of the application.

Then tell them that to overcome such problems, they can use the data caching technique. To perform data caching in an ASP.NET MVC application, they can use the MemoryCache class.

Then you refer to the code that shows how to use the MemoryCache class to cache data.

## Data Caching 2-2

- ◆ In the preceding code:
  - ◆ The AddOrGetExisting() method enables the application to refresh and access data from the cache. This method accesses the data from the cache if it contains the relevant data.
  - ◆ If there is no relevant data in the cache, then, the AddOrGetExisting() method allows adding the data to the cache by invoking the getUser() method.
  - ◆ The first parameter of the AddOrGetExisting() method specifies the unique identifier of the object that needs to be stored in the memory cache.
  - ◆ The second parameter specifies the object that needs to be stored in the cache, and the third parameter specifies the time when the cache should expire.
- ◆ You can use a HTTP caching technique in the browser and proxy cache.
- ◆ Storing data in the browser cache allows you to reduce the process of downloading content from the server repeatedly.

Display slide 32 and refer to the code displayed on slide 31.

Explain them that in the preceding code, the AddOrGetExisting() method enables the application to refresh and access data from the cache. This method accesses the data from the cache if it contains the relevant data. If there is no relevant data in the cache, then, the AddOrGetExisting() method allows adding the data to the cache by invoking the getUser() method. The first parameter of the AddOrGetExisting() method specifies the unique identifier of the object that needs to be stored in the memory cache. The second parameter specifies the object that needs to be stored in the cache, and the third parameter specifies the time when the cache should expire.

Then tell them that they can use a HTTP caching technique in the browser and proxy cache. Storing data in the browser cache allows them to reduce the process of downloading content from the server repeatedly.

**Slide 33**

Understand bundling and minification.

**Bundling and Minification**

- ◆ The ASP.NET MVC Framework provides two techniques known as Bundling and Minification.
- ◆ These techniques allow you to improve the load time of a Web page in an application.
- ◆ This is done by reducing the number of user requests to the server and the size of the requested resources.

© Aptech Ltd. State Management and Optimization / Session 9 33

Use slide 33 to explain the students that the ASP.NET MVC Framework provides two techniques known as Bundling and Minification. These techniques allow improving the load time of a Web page in an application. This is done by reducing the number of user requests to the server and the size of the requested resources.

**Slides 34 to 36**

Understand using bundling.

### Using Bundling 1-3

- ◆ Bundling is a technique that allows you to reduce the user requests in your application by combining several individual scripts into a single request.
- ◆ When there are multiple files that need to be downloaded, at times it became very time consuming.
- ◆ As a solution, you can use the bundling technique that allows you to combine multiple files and then, it can be downloaded as a single entity.
- ◆ To use the bundling technique in an application, you need to define the files that you want to combine together.
- ◆ You can achieve this using the BundleConfig class in the App\_Start folder of your application.
- ◆ You can create a bundle using the RegisterBundles() method of the BundleConfig class.

© Aptech Ltd. State Management and Optimization / Session 9 34

Use slide 34 to explain the students that bundling is a technique that allows reducing the user requests in an application by combining several individual scripts into a single request. When there are multiple files that need to be downloaded, at times it became very time consuming.

Then tell them that as a solution, they can use the bundling technique that allows them to combine multiple files and then, it can be downloaded as a single entity. To use the bundling technique in an application, they need to define the files that they want to combine together.

Mention that they can combine and bundle files using the BundleConfig class in the App\_Start folder of an application. They can create a bundle using the RegisterBundles() method of the BundleConfig class.

## Using Bundling 2-3

- ◆ Following code snippet shows using the RegisterBundles() method of the BundleConfig class:

```
public static void RegisterBundles(BundleCollection bundles) {
    bundles.Add(new ScriptBundle("~/bundles/jqueryval").Include(
        "~/Scripts/jquery.unobtrusive",
        "~/Scripts/jquery.validate"));
    bundles.Add(new styleBundle("~/Content/css").Include("~/Content/Site.css",
        "~/Content/styles.css"));
}
```

Snippet

- ◆ In this code:

- ◆ The RegisterBundles() method creates, registers, and configures bundles.
- ◆ A new instance of the ScriptBundle class is created to create a bundle of scripts.
- ◆ The path, ~/bundles/jqueryval, has been assigned to the bundle.
- ◆ Then, the Include() method on the bundle object enables you to specify the files that should be included in the bundle.
- ◆ Finally, the \* symbol is used to indicate that the bundle should include all the script files whose name begins with jquery.unobtrusive or jquery.validate.

Display slide 35 to refer to the code displayed on the slide that shows using the RegisterBundles( ) method of the BundleConfig class.

Explain the students that in this code, the RegisterBundles( ) method creates, registers, and configures bundles. A new instance of the ScriptBundle class is created to create a bundle of scripts. The path, ~/bundles/ jqueryval, has been assigned to the bundle.

Then, the Include( ) method on the bundle object enables them to specify the files that should be included in the bundle. Finally, the \* symbol is used to indicate that the bundle should include all the script files whose name begins with jquery.unobtrusive or jquery.validate.

## Using Bundling 3-3

- ◆ You can include these bundled files in a view.
- ◆ For that, you need to use the @Styles.Render and @Scripts.Render methods in your view.
- ◆ After that, you need to specify a path that you have configured in the bundle configuration as a parameter to these methods.
- ◆ Following code snippet shows using the @Styles.Render and @Scripts.Render methods in a view:

Snippet

```
@Scripts.Render("~/bundles/jqueryval")
@Styles.Render("~/Content/css")
```

© Aptech Ltd. State Management and Optimization / Session 9 36

Use slide 36 to explain the students that they can include these bundled files in a view. For that, they need to use the @Styles.Render and @Scripts.Render methods in a view. After that, they need to specify a path that they have configured in the bundle configuration as a parameter to these methods.

Then refer to the code given on slide 36 and explain to students that using the @Styles.Render and @Scripts.Render methods in a view helps in bundling the Styles and Scripts along with the application.

## Slide 37

Understand using minification.

The slide has a dark red background with a white header bar. The title 'Using Minification' is centered in a white font. Below the title is a bulleted list of four points, also in white font. At the bottom of the slide, there is a footer bar with three items: '© Aptech Ltd.', 'State Management and Optimization / Session 9', and the number '37'.

**Using Minification**

- ◆ Minification allows you to reduce the size of a file by removing unnecessary whitespaces and comments, and shortening the variable names.
- ◆ You can use such code while using interpreted languages, such as JavaScript.
- ◆ By using minification, you can reduce the size of a file and thus reduce the time required to access it from the server and load it into the browser.

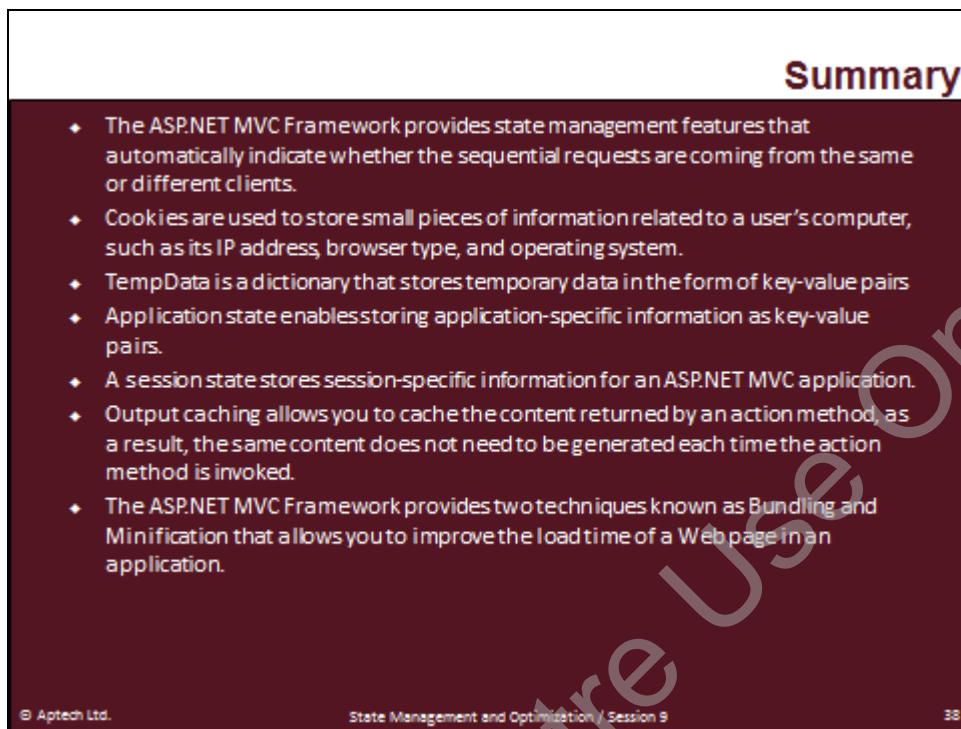
© Aptech Ltd. State Management and Optimization / Session 9 37

Use slide 37 to explain the students that minification allows reducing the size of a file by removing unnecessary whitespaces and comments, and shortening the variable names. They can use such code while using interpreted languages, such as JavaScript.

Tell them that by using minification, they can reduce the size of a file and thus reduce the time required to access it from the server and load it into the browser.

### Slide 38

Let us summarize the session.



The slide has a dark red background with a white header bar. The header bar contains the word "Summary" in a large, bold, white font. Below the header, there is a list of bullet points in white text. At the bottom of the slide, there is a footer bar with three items: "© Aptech Ltd.", "State Management and Optimization / Session 9", and a small number "38". A large, semi-transparent watermark reading "Aptech Centre of Excellence Only" is diagonally across the slide.

### Summary

- ◆ The ASP.NET MVC Framework provides state management features that automatically indicate whether the sequential requests are coming from the same or different clients.
- ◆ Cookies are used to store small pieces of information related to a user's computer, such as its IP address, browser type, and operating system.
- ◆ TempData is a dictionary that stores temporary data in the form of key-value pairs.
- ◆ Application state enables storing application-specific information as key-value pairs.
- ◆ A session state stores session-specific information for an ASP.NET MVC application.
- ◆ Output caching allows you to cache the content returned by an action method, as a result, the same content does not need to be generated each time the action method is invoked.
- ◆ The ASP.NET MVC Framework provides two techniques known as Bundling and Minification that allows you to improve the load time of a Web page in an application.

© Aptech Ltd. State Management and Optimization / Session 9 38

In slide 38, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- The ASP.NET MVC Framework provides state management features that automatically indicate whether the sequential requests are coming from the same or different clients.
- Cookies are used to store small pieces of information related to a user's computer, such as its IP address, browser type, and operating system.
- TempData is a dictionary that stores temporary data in the form of key-value pairs.
- Application state enables storing application-specific information as key-value pairs.
- A session state stores session-specific information for an ASP.NET MVC application.
- Output caching allows you to cache the content returned by an action method, as a result, the same content does not need to be generated each time the action method is invoked.
- The ASP.NET MVC Framework provides two techniques known as Bundling and Minification that allows you to improve the load time of a Web page in an application.

### **9.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. What are the advantages and disadvantages of TempData?
2. What are application state variables?
3. What are the actions that the built-in session state automatically performs?
4. What are the benefits of output caching?

## Session 10

### Authentication and Authorization

#### **10.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **10.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe authentication
- Explain and describe how to implement authentication
- Define and describe ASP.NET Identity
- Define and describe the process of authorization

#### **10.1.2 Teaching Skills**

To teach this session successfully, you must know about the concepts of authorization and authentication used for securing Web applications. You should aware yourself with the ASP.NET framework mechanism to restrict users access to Web applications. You should also aware yourself with the use of ASP.NET identity.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities:**

Follow the order given here during In-Class activities.

**Overview of the Session:**

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the process of authentication and authorization used to secure ASP.NET applications. They will learn about how to implement authentication in an ASP.NET MVC application. Finally, this session will discuss about the ASP.NET Identity.

For Aptech Centre Use Only

## **10.2 Introduction**

### **Slide 3**

Let us understand authentication process.

The slide has a dark background with a subtle geometric pattern. At the top right, the word 'Authentication' is written in a bold, white, sans-serif font. Below it is a bulleted list of three items, also in white font:

- ◆ Authentication is the process of validating the identity of a user before granting access to a restricted resource in an application.
- ◆ When the application receives a request from users, it tries to authenticate the user.
- ◆ Typically, authentication allows identifying an individual based on the user name and password provided by the user.

At the bottom left, there is a small logo and the text '© Aptech Ltd.'. In the center, it says 'Authentication and Authorization / Session 10'. On the far right, there is a small number '3'.

Use slide 3 to explain authentication process.

Discuss a scenario of buying products on an online Web store. Consider a scenario where you are developing an ASP.NET MVC application. This application allows users to view and buy products. On this application, though available products can be viewed by any users, you want that only registered users can buy products. So, a user who wants to buy products first needs to register to the application and then, he/she can buy products on the Web site.

To achieve such a restriction on the Web applications, you can identify a user by using the authentication mechanism.

Then, explain them that authentication is the process of validating the identity of a user, before granting access to the restricted resources in the application. When the application receives a request from users, it tries to authenticate the user.

Authentication provides answers to following questions:

- a. Who is the current user?
- b. Does the user represent what it says it can?

Tell them that typically, authentication allows identifying an individual based on the user name and password provided by the user.

**Additional Information:**

Client-side validation performed using JavaScript in the browser is one of the common validation technique used in the applications. However, it is very easy for attackers to submit a form directly to the server and by-pass the client-side validation. Thus, client validation should not be considered as a convenient security feature and developers should always validate the data they accept from the end-users or clients accessing the Web application on the Internet.

## Slide 4

Let us understand the types of authentication modes.

### Authentication Modes 1-8

The ASP.NET MVC Framework enables you to use the following three types of authentication modes:

- ❖ Forms authentication
- ❖ Windows authentication
- ❖ OpenID/OAuth

While using the Form authentication:

- ❖ The application itself is responsible for collecting user's credentials and then, authenticates the user.
- ❖ The application can authenticate the users by providing a login form on the Web page.
- ❖ When a user successfully logs into the application and gets authenticated to view a Web page, it generates a cookie that is served as an authentication token to the user.
- ❖ The cookie is then, used by the browser for the future requests made by the user, and thus allows the application to validate the requests.

© Aptech Ltd. Authentication and Authorization / Session 10 4

Use slide 4 to explain the students that the ASP.NET MVC Framework enables them to use the following three types of authentication modes:

- Forms authentication
- Windows authentication
- OpenID/OAuth

Tell them that when they are using the Form authentication mode, the application itself is responsible for collecting user's credentials and then, authenticates the user. This is the most common approach for Web applications accessed on the Internet.

Forms Authentication enables a developer to create an HTML login form within their application, and then validate the username/password submitted by the end user. The end user details are validated against a database or other password credential store.

Then, explain that when user successfully logs into the application and gets authenticated to view a Web page, the application generates a cookie that is served as an authentication token to the user. The cookie is then used by the browser for the future requests made by the user, and thus, allows the application to validate the subsequent requests from the same client.

Tell them that cookie is tightly coupled with the user's session. In case, if the session times out occurs or the users closes the Web browser, the session as well as cookie will become invalid and user has to login again to establish a new session.

For Aptech Centre Use Only

## Slide 5

Let us understand the authentication modes.

### Authentication Modes 2-8

- ◆ When you create an ASP.NET MVC application in Visual Studio 2013, the application does not automatically uses any type of authentication.
- ◆ To authenticate users by using Forms authentication you need to manually configure the `<authentication>` element under the `<system.web>` element present in the Web.config file.
- ◆ Following code snippet shows the default `<authentication>` element in the Web.config file:

`<system.web>`  
        `<authentication mode="None" />`  
    `</system.web>`

**Snippet**

- ◆ In this code, the `<authentication>` element contains the `mode` attribute that allows you to specify what type of authentication is to be used by the application.

© Aptech Ltd.      Authentication and Authorization / Session 10      5

In slide 5, explain that when they create an ASP.NET MVC application in Visual Studio 2013, the application does not automatically use any type of authentication.

To authenticate users by using Forms authentication, they need to manually configure the `<authentication>` element under the `<system.web>` element present in the Web.config file.

Then, refer to the code displayed on the slide which shows the default `<authentication>` element in the Web.config file. Explain them that in this code, the `<authentication>` element contains the `mode` attribute that allows specifying what type of authentication is to be used by the application.

The `mode` attribute can accept the values such as Windows, Forms, Passport, or None. Tell them that the selection of ASP.NET authentication mode depends upon which IIS authentication scheme you have chosen. If you have selected scheme as Anonymous, this means you are implementing your own authentication and `mode` attribute can be set to Forms, Passport, or None. Otherwise, you can use the Windows authentication mode.

## Slide 6

Let us understand Form authentication mode.

### Authentication Modes 3-8

- ◆ Now to use Forms authentication mode, you need to change the mode attribute of the <authentication> element to Forms.
- ◆ Following code snippet shows specifying the authentication mode as Forms:

```
<system.web>
  <authentication mode="Forms" />
</system.web>
```

**Snippet**

- ◆ In this code, the mode attribute of the <authentication> element is set to Forms, which indicates that application should use the Forms authentication.

© Aptech Ltd.      Authentication and Authorization / Session 10      6

Use slide 6 to explain that, to use Forms authentication mode, they need to change the mode attribute of the <authentication> element to Forms.

Then, refer to the code displayed on the slide which shows how to specify the authentication mode as Forms.

Explain them that in this code, the mode attribute of the <authentication> element is set to Forms, which indicates that application should use the Forms authentication.

#### Additional Information

The advantage of Form authentication is that, it does not have to be part of a domain-based network to have access to the application. It is advantageous for the e-commercial Web sites dealing in order products management. In such Web applications, the access to user information is required. However, it is subjected to the attacks of the cookies; however, the use of Secure Sockets Layer (SSL) can help in securing form authentication.

When **requireSSL** is set to **true**, the encrypted connection helps protect the credentials of the user, and ASP.NET sets the **HttpCookie.Secure** property for the authentication cookie. The compliant browser does not return the cookie, unless the connection uses SSL.

Following example shows how to do this in the Web.config file for your application:

```
<configuration>
  <system.web>
    <authentication mode="Forms">
      <forms name=".ASPxAUTH"
            loginUrl="login.aspx"
            protection="All"
            timeout="20"
            requireSSL="true">
      </forms>
    </authentication>
  . . .
  </system.web>
</configuration>
```

With forms authentication, ASP.NET authenticates the users and then, redirects unauthenticated users to the logon page that is specified by the **loginUrl** attribute of the **<forms>** element in the Web.config file.

#### **Additional Information:**

ASP.NET MVC framework allows the developer to choose between Windows authentication and Form authentication.

## Slide 7

Let us understand Windows authentication.

The slide has a dark purple background with a faint geometric pattern. At the top center, the title 'Authentication Modes 4-8' is displayed in a white, bold, sans-serif font. Below the title, there is a bulleted list of four points, also in a white sans-serif font. At the bottom left, there is small text '© Aptech Ltd.' and at the bottom right, there is small text 'Authentication and Authorization / Session 10' followed by a small number '7'.

### Authentication Modes 4-8

- ◆ Windows authentication is best suited to an intranet environment where a set of known users are already logged on to a network.
- ◆ When you use the Windows authentication mode in your application, it is the firewall that takes care of the authentication process.
- ◆ In this mode, the application uses the users credential authenticated by the company's firewall for security checks.
- ◆ To configure your application to use the Windows authentication mode, you need to change the mode attribute of the <authentication> element to Windows.

Using slide 7, explain that Windows authentication is best suited to an Intranet environment where a set of known users are already logged on to a network. When they use the Windows authentication mode in an application, it is the firewall that takes care of the authentication process.

Tell them that in this mode, the application uses the users credential authenticated by the company's firewall for security checks. To configure the application to use the Windows authentication mode, they need to change the mode attribute of the <authentication> element to Windows.

### Additional Information

You can choose Windows authentication, if your user accounts are maintained by a Microsoft Windows NT domain controller or within Microsoft Windows Active Directory and there are no firewall issues. One of the benefits of using Windows authentication is that, it can be integrated with IIS which validates the user and passes a Windows token for the authenticated user to ASP.NET along with each Web request.

## Slide 8

Let us understand OAuth and OpenID authorization protocols.

### Authentication Modes 5-8

- ◆ The OAuth and OpenID are authorization protocols that enable a user to logon to an application by using the credentials for third party authentication providers, such as Google, Twitter, Facebook, and Microsoft.
- ◆ When you create an ASP.NET MVC Web application using Visual Studio 2013, the third party authentication providers need to be configured.
- ◆ You can do this in the Startup.Auth.cs file present in the App\_Start folder in the application directory.
- ◆ By default, the Startup.Auth.cs file contains the commented code that enables logging in with third party login providers.

© Aptech Ltd.      Authentication and Authorization / Session 10      8

Using slide 8, explain the students that the OAuth and OpenID are authorization protocols that enable a user to logon to an application by using the credentials for third party authentication providers, such as Google, Twitter, Facebook, and Microsoft. When they create an ASP.NET MVC Web application using Visual Studio 2013, the third party authentication providers need to be configured. The developer can do this in the Startup.Auth.cs file present in the App\_Start folder in the application directory.

Then, tell them that by default, the Startup.Auth.cs file contains the commented code that enables logging in with third party login providers.

## Slides 9 to 11

Let us understand how to implement authentication.

### Authentication Modes 6-8

- Following code snippet shows the content of the Startup.Auth.cs file:

**Snippet**

```

using Microsoft.AspNet.Identity;
using Microsoft.Owin;
using Microsoft.Owin.Security.Cookies;
using Owin;
namespace SecureApp
{
    public partial class Startup
    {
        // For more information on configuring authentication, please visit
        // http://go.microsoft.com/fwlink/?LinkId=301864
        public void ConfigureAuth(IAppBuilder app)
        {
            // Enable the application to use a cookie to store information for
            // the signed in user
            app.UseCookieAuthentication(new CookieAuthenticationOptions
            {
                AuthenticationType = DefaultAuthenticationTypes.ApplicationCookie,
                LoginPath = new PathString("/Account/Login")
            });
        }
    }
}

```

© Aptech Ltd.      Authentication and Authorization / Session 10      9

### Authentication Modes 7-8

- Following code snippet shows the content of the Startup.Auth.cs file:

**Snippet**

```

// Use a cookie to temporarily store information about a user logging in
// with a third party login provider
app.UseExternalSignInCookie(DefaultAuthenticationTypes.ExternalCookie);
// Uncomment the following lines to enable logging in with third
// party login providers
// app.UseMicrosoftAccountAuthentication(
//     clientId: "",
//     clientSecret: "");
// app.UseTwitterAuthentication(
//     consumerKey: "",
//     consumerSecret: "");
// app.UseFacebookAuthentication(
//     appId: "",
//     appSecret: "");
// app.UseGoogleAuthentication();
}
]

```

© Aptech Ltd.      Authentication and Authorization / Session 10      10

## Authentication Modes 8-8

- ◆ The preceding code shows content of the Startup.Auth.cs file where the third party login provider's code is commented.
- ◆ As you have seen that the third party login provider's code is commented, because before you can use the services provided by an external login providers, you have to register the application with them.
- ◆ To use the services of Google, you are not required to register with the ASP.NET MVC application if you are already registered with Google.
- ◆ To use the services of Google, you only need to uncomment the call to the OAuthWebSecurity.RegisterGoogleClient() method.

© Aptech Ltd.

Authentication and Authorization / Session 10

11

Using slides 9 to 11, explain the code that shows the content of the Startup.Auth.cs file where the third party login provider's code is commented.

As you have seen that the third party login provider's code is commented, because before you can use the services provided by an external login providers, you have to register the application with them. To use the services of Google, you are not required to register with the ASP.NET MVC application if you are already registered with Google.

To use the services of Google, you only need to uncomment the call to the OAuthWebSecurity.RegisterGoogleClient() method. When you run the default application created by Visual Studio 2013 and click the Log in link on the home page, the application will provide the option to log in using Google credentials.

## Slide 12

Let us understand when to use the Authorize attribute.

### Using the Authorize Attribute

- ◆ When a proper security mechanism is implemented on an application, a user must be enforced to login to the application to access some specific resources.
- ◆ This type of mechanism is implemented in an application by using the Authorize action filter.
- ◆ You can use these attributes to add behavior that can be executed either before an action method is called or after an action method is executed.
- ◆ In an ASP.NET MVC application, you can use the Authorize attribute to secure an action method, a controller, or the entire application.

© Aptech Ltd. Authentication and Authorization / Session 10 12

Using slide 12, explain when to use the Authorize attribute.

There is a fundamental difference in protected resources between WebForms and MVC. In WebForms, the resources you are trying to protect are the pages themselves, and since the pages exist on disk at a well-known path, you can use Web.config to secure them. However, in MVC, the resources you are trying to protect are actually controllers and actions, not individual paths and pages. If you try protecting the path, rather than the controller, your application likely has a security threat.

In MVC, by default, all controllers and actions are accessible to all users, both authenticated and guest. To secure controllers or actions, the Authorize attribute has been provided. Tell them that when a proper security mechanism is implemented on an application, a user must be enforced to login to the application to access some specific resources. This type of mechanism is implemented in an application by using the Authorize action filter.

Then, tell them that they can use these attributes to add behavior that can be executed either before an action method is called or after an action method is executed. In an ASP.NET MVC application, they can use the Authorize attribute to secure an action method, a controller, or the entire application.

**Slides 13 and 14**

Let us understand how to secure a controller action.

## Securing a Controller Action 1-2

Consider a scenario of an online shopping store application,

- ◆ You want to display the list of available items in the home page from where a user can select an item and purchase it.
- ◆ You want that all users can see the list of item on the home page, but only the authenticated user can purchase an item using this application.
- ◆ In such situation, you need to create a `BuyItem()` action method in a controller class, named `Product`.
- ◆ While creating the `BuyItem()` action method, you should ensure that this method is only accessible for the authenticated users.
- ◆ For this, you need to add the `Authorize` attribute on the `BuyItem()` action method.

© Aptech Ltd.

Authentication and Authorization / Session 10

13

## Securing a Controller Action 2-2

- ◆ Following code snippet shows adding the `Authorize` attribute the `BuyItem()` action method:

Snippet

```
[Authorize]
public ActionResult BuyItem()
{
    return View();
}
```

- ◆ In this code, the `Authorize` attribute is added on the `BuyItem()` action method.

© Aptech Ltd.

Authentication and Authorization / Session 10

14

Using slides 13 and 14, explain the students how to secure controller action.

Consider a scenario of an online store shopping application. It needs to display a list of available items on the home page from where a user can select an item and purchase it. Then, they want that all users can see the list of item on the home page, however, only the authenticated user can purchase an item using this application.

In such situation, they need to create a `BuyItem()` action method in a controller class, named `Product`. While creating the `BuyItem()` action method, they should ensure that this method is only accessible for the authenticated users. For this, they need to add the `Authorize` attribute on the `BuyItem()` action method.

Then, explain them the code snippet that shows how to add the `Authorize` attribute to the `BuyItem()` action method. This restricts the `BuyItem()` action method from anonymous access to the method.

## Slides 15 to 17

Let us understand how to secure a controller.

### Securing a Controller 1-3

- ◆ You can also use the Authorize attribute to a controller to secure the entire controller.
- ◆ For this, you need to add the Authorize attribute on the controller.
- ◆ Following code snippet shows adding the Authorize attribute to a controller:

Snippet

```
[Authorize]
public class ProductController : Controller
{
}
```

- ◆ The code will restricts all the action methods defined in the ProductController controller class.

© Aptech Ltd. Authentication and Authorization / Session 10 15

### Securing a Controller 2-3

- ◆ Sometime, you might not want to secure certain action methods in a controller.
- ◆ In such case, you can use the AllowAnonymous attribute on those action methods.
- ◆ Following code snippet shows using the AllowAnonymous attribute:

Snippet

```
[AllowAnonymous]
public ActionResult Index()
{
    return View();
}
```

- ◆ This code uses the AllowAnonymous attribute on the Index() action. As a result, access to this method does not require authentication.

© Aptech Ltd. Authentication and Authorization / Session 10 16

### Securing a Controller 3-3

- ◆ At times, you may want that all the users to be authenticated for an entire application.
- ◆ For this, you need to add the Authorize attribute as a global filter, by adding it to the global filters collection in the RegisterGlobalFilters() method in the FilterConfig.cs file.
- ◆ Following code snippet uses the RegisterGlobalFilters() method in the FilterConfig.cs file:

Snippet

```
public static void RegisterGlobalFilters(GlobalFilterCollection filters)
{
    filters.Add(new System.Web.Mvc.AuthorizeAttribute());
    filters.Add(new HandleErrorAttribute());
}
```

- ◆ This code applies the Authorize attribute to all action methods in the application. However, you can use the AllowAnonymous attribute to allow access to specific controllers or action methods.

Using slides 15 to 17, explain the students that they can also use the Authorize attribute to a controller to secure the entire controller. For this, they need to add the Authorize attribute on the controller.

Tell them to restrict the access to all the actions preformed on the controller, they can apply the Authorize attribute to the controller class. Explain them this with the help of the code snippet restricting access to the ProductController class.

In slide 16, explain that sometimes, it may not be required to secure certain action methods in a controller. In such case, they can use the AllowAnonymous attribute on those action methods. For example, the controller actions such as Login and Register can have an anonymous access for the convenience of the users. In such cases, you can access anonymous access to these actions.

Following example shows the AccountController with anonymous access to Login and Register.

```
[Authorize]
public class AccountController : Controller {
    [AllowAnonymous]
    public ActionResult Login() {
        // ...
    }
}
```

```
[AllowAnonymous]  
public ActionResult Register() {  
    // ...  
}  
}
```

Then, refer to the code on slide 16 that shows use of the AllowAnonymous attribute. Explain that this code uses the AllowAnonymous attribute on the Index( ) action. As a result, access to this method does not require authentication.

In slide 17, explain the students that at times, they may want that all the users to be authenticated for an entire application. For this, they need to add the Authorize attribute as a global filter, by adding it to the global filters collection in the RegisterGlobalFilters( ) method in the FilterConfig.cs file.

Then, refer to the code that uses the RegisterGlobalFilters( ) method in the FilterConfig.cs file.

The System.Web.Mvc is the namespace that contains classes and interfaces that support the ASP.NET Model View Controller (MVC) framework for creating Web applications. The AuthorizeAttribute is the class in this namespace which specifies that access to a controller or action method is restricted to users who meet the authorization requirement.

Explain them that this code applies the Authorize attribute to all action methods in the application. However, they can use the AllowAnonymous attribute to allow access to specific controllers or action methods.

Tell them that using global filters to secure the entire Web site make more sense than using Authorize attribute on the individual controllers.

## Slide 18

Let us understand ASP.NET Identity in ASP.NET MVC 5.

**ASP.NET Identity 1-7**

- ◆ Prior to ASP.NET MVC 5, the MVC Framework used the Membership API for implementing authentication and authorization.
- ◆ ASP.NET MVC 5 uses ASP.NET Identity, which is a new membership system for ASP.NET MVC applications.
- ◆ You can use ASP.NET Identity to add login features to your application.
- ◆ ASP.NET Identity uses the Code First approach to persist user information in a database.
- ◆ When you create an ASP.NET MVC 5 application in Visual Studio 2013, the Identity and Account management classes are automatically added to the project.
- ◆ The `IdentityModel.cs` file created by Visual Studio 2013 manages the identity of an application.

© Aptech Ltd. Authentication and Authorization / Session 10 18

Use slide 18 to explain the students that prior to ASP.NET MVC 5, the MVC Framework used the Membership API for implementing authentication and authorization. ASP.NET MVC 5 uses ASP.NET Identity, which is a new membership system for ASP.NET MVC applications.

### Membership in ASP.NET

The ASP.NET membership system was developed in ASP.NET 2.0. It was designed to solve membership requirements such as form authentication. To store user details such as name, password, and profile data for the Web application, SQL Server database was used.

However, there are limitations of ASP.NET membership that are as follows:

- The database schema was designed for SQL Server and you can't change it.
- You can add profile information, but the additional data is packed into a different table, which makes it difficult to access by any means except through the Profile Provider API.
- The provider system is designed around assumptions appropriate for a relational database.
- The developer can write a provider to store membership information in a non-relational storage mechanism, such as Windows Azure Storage Tables, however, the developer have to work around the relational design by writing a lot of code. This also leads to exceptions for the methods that do not support NOSQL database.

- Mostly, the log-in and log-out functionalities are based on Form authentication, the membership system cannot be used for performing authentication for external identity providers such as Facebook, Twitter, Google, and so on.
- Also, it does not provide support for organizational accounts such as Windows Azure Active Directory.

Today, the Web identity methods have changed, due to Web becoming more social. It does not depend on the assumptions that the user will log-in only with their registered username and password. Users can interact through other social medias such as Facebook, Twitter, and so on. Thus, a modern membership system must enable redirection-based log-ins to authentication providers such as Facebook, Twitter, and others.

Considering these changes in the requirements of a Web application, ASP.NET Identity was developed.

Tell them that the developer can use ASP.NET Identity to add login features to an application. ASP.NET Identity uses the Code First approach to persist user information in a database.

Then, tell them that when they create an ASP.NET MVC 5 application in Visual Studio 2013, the Identity and Account management classes are automatically added to the project. The `IdentityModel.cs` file created by Visual Studio 2013 manages the identity of an application.

### **Additional Information**

Some of the features of ASP.NET Identity are as follows:

- ASP.NET Identity can be used with any ASP.NET framework such as ASP.NET MVC, Web Forms, Web Pages, and so on.
- ASP.NET Identity provides control over the schema of user and profile information.
- By default, the ASP.NET Identity system stores all the user information in a database. Thus, the tasks such as changing table name or data type in the table becomes easier.
- With ASP.NET Identity, it is easy to plug in different storage mechanisms such as SharePoint, Windows Azure Storage Table Service, NoSQL databases, and so on without throwing exception from the runtime.
- ASP.NET Identity allows restricting parts of applications by roles.
- ASP.NET Identity allows the developer to add social log-ins in the application and work with user-specific data.

## Slide 19

Let us understand how to work with ASP.NET Identity.

### ASP.NET Identity 2-7

- ◆ Following code snippet shows the `IdentityModel.cs` file:

**Snippet**

```
namespace WebApplication1.Models {
    public class ApplicationUser : IdentityUser {
    }
    public class ApplicationDbContext : IdentityDbContext<ApplicationUser> {
        public ApplicationDbContext()
            : base("DefaultConnection")
        {
        }
    }
}
```

◆ In this code, the `ApplicationUser` class that extends from the `IdentityUser` class is responsible for managing user accounts in the application. The `ApplicationDbContext` class inherits from `IdentityDbContext` and allows the application to interact with the database where account data of users are stored.

© Aptech Ltd.      Authentication and Authorization / Session 10      19

In slide 19, refer to the code displayed on the slide that shows the `IdentityModel.cs` file.

Explain that in this code, the `ApplicationUser` class that extends from the `IdentityUser` class is responsible for managing user accounts in the application. The `ApplicationDbContext` class inherits from `IdentityDbContext` and allows the application to interact with the database where account data of users are stored.

#### Additional Information

The `Microsoft.AspNet.Identity.EntityFramework` package has the Entity Framework implementation of ASP.NET Identity which will persist the ASP.NET Identity data and schema to SQL Server.

## Slides 20 to 24

Let us understand how ASP.NET is used in ASP.NET MVC application.

### ASP.NET Identity 3-7

- ◆ You will now learn how ASP.NET Identity is used in a default ASP.NET MVC application created using Visual Studio 2013.
- ◆ When you execute the default application, the Home page appears.
- ◆ Following figure shows the Home page:

© Aptech Ltd. Authentication and Authorization / Session 10 20

### ASP.NET Identity 4-7

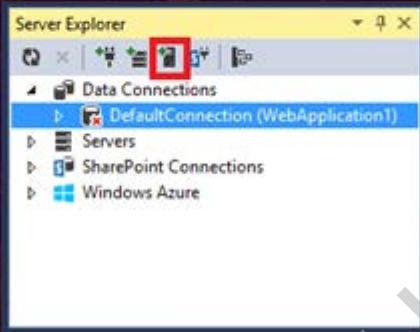
In the Home page, you need to perform the following tasks:

- ◆ Click the Register link. The Register page is displayed.
- ◆ Add the registration details.
- ◆ Click the Register button. The application logs you on and the home page is displayed with a welcome message based on the specified user name and a Log off option.
- ◆ Following figure shows the welcome message:

© Aptech Ltd. Authentication and Authorization / Session 10 21

## ASP.NET Identity 5-7

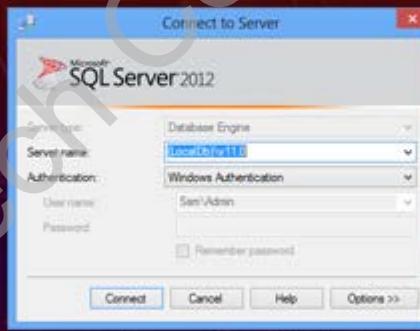
- ◆ Internally, the application uses ASP.NET Identity to create database tables for the application to hold account data.
- ◆ You can view the generated tables by performing the following tasks:
  - + Select View → Server Explorer. The Server Explorer window is displayed.
  - + Select DefaultConnection (WebApplication1) under the Data Connections node and click the Add SQL Server icon.
- ◆ Following figure shows the Server Explorer window:



© Aptech Ltd. Authentication and Authorization / Session 10 22

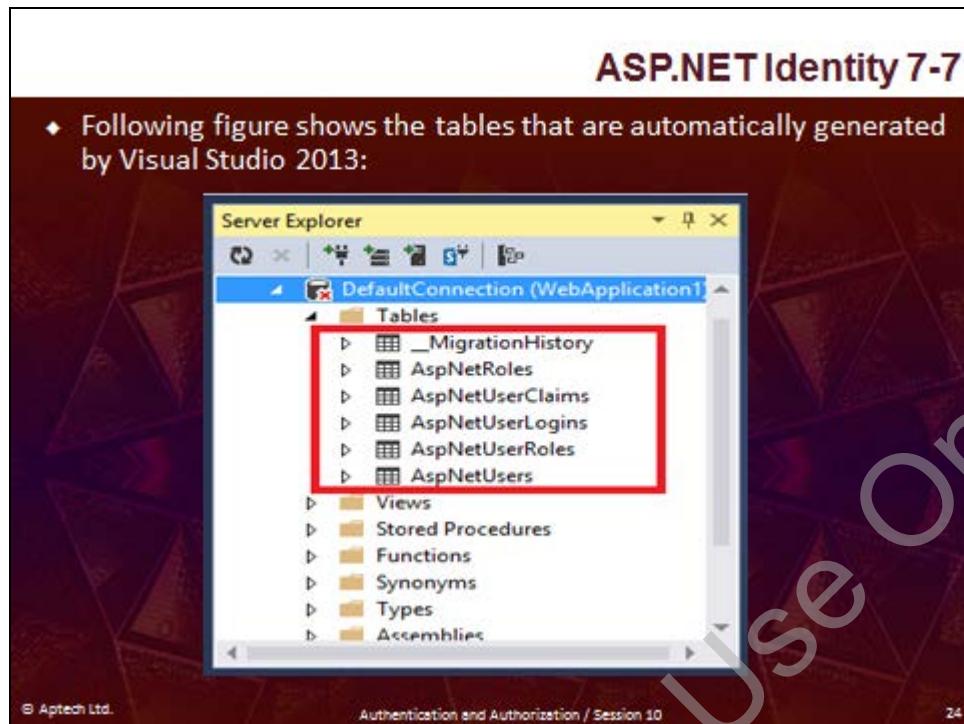
## ASP.NET Identity 6-7

- + In the Connect to SQL Server dialog box that appears, type (LocalDb)\v11.0 in the Server name text field.
- ◆ Following figure shows the Connect to Server dialog box:



- + Click Connect.
- + Expand the DefaultConnection (WebApplication1) → Tables node to view the tables that are automatically generated by Visual Studio 2013 based on ASP.NET Identity.

© Aptech Ltd. Authentication and Authorization / Session 10 23



Using slides 20 to 24, explain the students that they will now learn how ASP.NET Identity is used in a default ASP.NET MVC application created using Visual Studio 2013.

When they execute the default application, the Home page appears. Then, you can refer to the figure displayed on the slide that shows the Home page.

In slide 21, explain the students about the tasks that they need to perform in the Home page, as displayed on the slide. Then, refer to the figure that shows the Welcome message.

In slide 22, explain that internally the application uses ASP.NET Identity to create database tables for the application to hold account data.

Tell them about the tasks that they perform to view the generated tables as displayed on the slide. Refer to the figure that shows the Server Explorer window.

In slide 23, refer to the figure that shows the **Connect to Server** dialog box.

Finally, in slide 24, refer to the figure that shows the tables that are automatically generated by Visual Studio 2013.

## Slide 25

Let us understand authorization.

## Authorization

- ◆ Authorization is the process of verifying whether an authenticated user has the privilege to access a requested resource.
- ◆ After creating users that can access your application by using the membership management service, you need to specify authorization rules for the users.
- ◆ You need to grant different permissions to different users to provide accessibility to the pages on your application.

© Aptech Ltd. Authentication and Authorization / Session 10 25

Using slide 25, explain them that authorization is the process of verifying whether an authenticated user has the privilege to access a requested resource.

Consider a scenario, you need to grant different permissions to different users to provide accessibility to the pages on your application. For example, on an online shopping store application, the users are able to view the list of available products. However, they cannot make any changes to the products that are displayed on a page of the application. However, a user with the administrator role in the application can modify the product list, such as add or remove any product from the list. This requires giving different levels of permission to different users of the application.

After creating users that can access an application by using the membership management service, the students need to specify authorization rules for the users. They need to grant different permissions to different users to provide accessibility to the pages on the application.

### Additional Information

Authorization allows you to make more granular choices when the developer grant access to resources.

- *Authorization filters* run before the controller action. If the request is not authorized, the filter returns an error response, and the action is not invoked.
- Within a controller action, you can get the current principal from the `ApiController.User` property. For example, you might filter a list of resources based on the user name, returning only those resources that belong to that user.

## Slides 26 to 28

Let us understand users and roles.

### Users and Roles 1-3

- ◆ In order to maintain a different set of settings for each user is a complex task.
- ◆ So, you should assemble the users in a group known as roles and then, define the permissions on these roles.
- ◆ In an ASP.NET MVC application, you can authorize some specific users and roles by using the Authorize attribute.
- ◆ Following code snippet uses the Authorize attribute to authorize specific users:

Snippet

```
[Authorize(Users="Mathews, Jones")]
public class ManagerProduct: Controller
{
    // Code to perform some operation
}
```

- ◆ In this code, only the users whose name is Mathews and Jones are the authorized users to access the ManagerProduct controller.

© Aptech Ltd. Authentication and Authorization / Session 10 26

### Users and Roles 2-3

- ◆ In an ASP.NET MVC application, you can also assign roles to different users.
- ◆ Thereafter, you can use the authorization at the role level.
- ◆ Following code snippet shows specifying role to users:

Snippet

```
[Authorize(Roles="Administrator")]
public class ManagerProduct: Controller
{
    // Code to perform some operation
}
```

- ◆ This code allows accessing the ManagerProduct controller to users with the Administrator role.

© Aptech Ltd. Authentication and Authorization / Session 10 27

## Users and Roles 3-3

- ◆ In an ASP.NET MVC application, you can also use a combination of roles and users.
- ◆ Following code snippet shows how to use a combination of roles and users:

**Snippet**

```
[Authorize(Roles="Manager", Users="Steve, Mathews")]
public class ManagerProduct : Controller
{
    // Code to perform some operation
}
```

- ◆ This code allows the users whose name is Steve and Mathews, and users with the Manager role to access the ManageStore controller.

© Aptech Ltd.      Authentication and Authorization / Session 10      28

Using slides 26 to 28, explain to the students that in order to maintain a different set of settings for each user, they should assemble the users in a group user known as roles and then, define the permissions on these roles.

Tell them that in an ASP.NET MVC application, they can authorize some specific users and roles by using the `Authorize` attribute. For example, the developer can restrict access to specific users such as Mathews and Jones as shown in the code snippet mentioned on slide 26.

Explain them that in this code, only the users whose name is Mathews and Jones are the authorized users to access the `ManagerProduct` controller.

In slide 27, you explain the students that in an ASP.NET MVC application, they can also assign roles to different users. Thereafter, they can use the authorization at the role level. Then, refer to the code displayed on slide 28 that shows specifying role to users.

Explain them that this code allows accessing the `ManagerProduct` controller to users with the `Administrator` role.

Using slide 28, explain the students that in an ASP.NET MVC application, they can also use a combination of roles and users.

Explain them that this code allows the users whose name are Steve and Mathews, and users with the Manager role to access the `ManageStore` controller.

## Slide 29

Let us understand role providers.

The slide has a dark purple header bar with the title 'Role Providers 1-2'. The main content area has a dark purple background with a subtle geometric pattern. A large, faint watermark reading 'For Aptech Centre Only' is visible across the slide. The footer contains the copyright notice '© Aptech Ltd.', the slide title 'Authentication and Authorization | Session 10', and the slide number '29'.

### Role Providers 1-2

- ◆ You need to create roles so that you can assign specific rights to specific users.
- ◆ The ASP.NET MVC Framework provides different role providers that you can use to implement roles.
- ◆ Some of them are as follows:
  - ◆ ActiveDirectoryRoleProvider: This provider class enables you to manage role information for an application in the Active Directory.
  - ◆ SqlRoleProvider: This provider class enables you manage role information for an application in SQL database.
  - ◆ SimpleRoleProvider: This role provider works with different versions of SQL Server.
  - ◆ UniversalProviders: This provider works with any database that Entity Framework supports.

Using slide 29, explain that they need to create roles so that they can assign specific rights to specific users. The ASP.NET MVC Framework provides different role providers that they can use to implement roles.

Then, tell them about the following roles:

- **ActiveDirectoryRoleProvider:** This provider class enables managing role information for an application in the Active Directory.
- **SqlRoleProvider:** This provider class enables managing role information for an application in SQL database.
- **SimpleRoleProvider:** This role provider works with different versions of SQL Server.
- **UniversalProviders:** This provider works with any database that Entity Framework supports.

**Tips:**

ASP.NET MVC also enables you to create custom role providers. By using custom role providers, you can implement role management that uses your own database schema and logic.

There are two primary reasons for creating a custom role provider that are as follows:

- You need to store role information in a data source that is not supported by the role providers included with the .NET Framework, such as a FoxPro database, an Oracle database, or other data source.
- You need to manage role information using a database schema that is different from the database schema used by the providers that ship with the .NET Framework. A common example of this would be authorization data that already exists in a SQL Server database for a company or Web site.

## Slide 30

Let us understand how to use Provider property.

### Role Providers 2-2

- ◆ You can use the Provider property provided by the Roles class to access the current role provider.
- ◆ Following code snippet uses the Provider property:

Snippet

```
var roles = (TestRoleProvider)Roles.Provider;
```

- ◆ In this code, the Provider property allows to access the role provider, named, TestRoleProvider. This piece of code will only work when the System.Web.Security namespace is included.

© Aptech Ltd. Authentication and Authorization / Session 10 30

Using slide 30, explain the students that once they can use the Provider property provided by the Roles class to access the current role provider.

Then, refer to the code that uses the Provider property.

Explain them that in this code, the Provider property allows accessing the role provider, named, TestRoleProvider. This piece of code will only work when the System.Web.Security namespace is included.

## Slides 31 to 34

Let us understand user accounts and roles.

### User Accounts and Roles 1-4

- ◆ When you create an MVC application using Visual Studio 2013, you need to create the membership database with a connection string that you need to specify in the Web.config file.
- ◆ You can add sample data in this database with the required roles by using the Seed() method.
- ◆ Following code snippet shows using the Seed() method:

Snippet

```
var roles = (SimpleRoleProvider)Roles.Provider;
if (!roles.RoleExists("Administrator"))
{
    roles.CreateRole("Administrator");
}
```

- ◆ In this code,
  - ◆ You first retrieve a reference of the simple role provider.
  - ◆ Then, you use this reference to check whether the Administrator exists in the database.
  - ◆ If it does not exist, the CreateRole() method creates the role named, Administrator.

© Aptech Ltd. Authentication and Authorization / Session 10 31

### User Accounts and Roles 2-4

- ◆ Once you have created an Administrator role in the database, you can add users to that role.
- ◆ Following code snippet shows assigning a user with the Administrator role:

Snippet

```
if (!roles.GetRolesForUser("Mark_Jones").Contains("administrator"))
{
    roles.AddUsersToRoles(new[] { "Mark_Jones" }, new[] { "administrator" });
}
```

In this code, if the user with the name Mark\_Jones is already assigned with the Administrator role, then, the AddUserstoRoles() method adds the Administrator role, to the user with the name, Mark\_Jones.

© Aptech Ltd. Authentication and Authorization / Session 10 32

## User Accounts and Roles 3-4

- Once you have created a role and users in the database, you can then, use the Authorize attribute on the action method that needs to be restricted.
- Following code snippet shows using the Authorize attribute so that the action method can be accessed by users with the Administrator role:

**Snippet**

```
[Authorize(Roles = "administrator")]
public ActionResult ManageProduct()
{
    return View();
}
```

- This code will display a login page when a user tries to access the ManageProduct page. When the user provides the valid credentials for a user with the administrator role then, the ManageProduct page is displayed.

## User Accounts and Roles 4-4

- You can also display only those links that a user has rights to access.
- For example, the ManageProduct link should be visible only to the users with administrator role.
- In such situation, you can use the ActionLink() method to hide the ManageProduct link from other users.
- Following code snippet shows using the ActionLink() method:

**Snippet**

```
@if(User.IsInRole("administrator"))
{
    @Html.ActionLink("ManageProduct", "ManageProduct")
}
```

- This code displays the ManageProduct link only when a user logs in as an administrator.

Using slides 31 to 34, explain to the students how to work with user accounts and roles.

Explain to the students when they create an MVC application using Visual Studio 2013, they need to create the membership database with a connection string that they need to specify in the Web.config file. They can add sample data in this database with the required roles by using the Seed( ) method.

Then, refer to the code that shows using the `Seed()` method. Explain them that this code first retrieves a reference of the simple role provider. Then, uses this reference to check whether the Administrator exists in the database. If it does not exist, the `CreateRole()` method creates the role named, Administrator.

Using slide 32, explain the students that once they have created an Administrator role in the database, they can add users to that role.

Then, refer to the code displayed on the slide that shows how to assign a user with the Administrator role.

Explain them that in this code, if the user with the name `Mark_Jones` is already assigned with the Administrator role, then, the `AddUserstoRoles()` method adds the Administrator role, to the user with the name, `Mark_Jones`.

Use slide 33 to explain the students that once they have created a role and users in the database, they can use the `Authorize` attribute on the action method that needs to be restricted.

Then, refer to the code that shows how to use the `Authorize` attribute so that the action method can be accessed by users with the Administrator role. Explain the students that this code will display a login page when a user tries to access the `ManageProduct` page. When the user provides the valid credentials for a user with the administrator role, then the `ManageProduct` page is displayed.

Use slide 34 to explain the students that they can also display only those links that a user has rights to access. For example, the `ManageProduct` link should be visible only to the users with administrator role. In such situation, they can use the `ActionLink()` method to hide the `ManageProduct` link from other users.

Then, refer to the code that shows how to use the `ActionLink()` method. Explain that this code displays the `ManageProduct` link only when a user logs in as an administrator.

### Slide 35

Let us summarize the session.

The slide has a dark purple background with a faint watermark reading "Session 10". At the top right, the word "Summary" is written in white. Below it is a bulleted list of nine points. At the bottom left is the copyright notice "© Aptech Ltd.", in the center is "Authentication and Authorization / Session 10", and at the bottom right is the number "35".

**Summary**

- ◆ Authentication is the process of validating the identity of a user before granting access to a restricted resource in an application.
- ◆ In an ASP.NET MVC application, you can use the Authorize attribute to secure an action method, a controller, or the entire application.
- ◆ ASP.NET Identity uses the Code First approach to persist user information in a database.
- ◆ Authorization is the process of verifying whether an authenticated user has the privilege to access a requested resource.
- ◆ In an ASP.NET MVC application, you can authorize some specific users and roles by using the Authorize attribute.
- ◆ The ASP.NET MVC Framework provides different role providers that you can use to implement roles.
- ◆ When you create an MVC application using Visual Studio 2013, you need to create the membership database with a connection string.

© Aptech Ltd.      Authentication and Authorization / Session 10      35

In slide 35, summarize the session. End the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- Authentication is the process of validating the identity of a user before granting access to a restricted resource in an application.
- In an ASP.NET MVC application, you can use the Authorize attribute to secure an action method, a controller, or the entire application.
- ASP.NET Identity uses the Code First approach to persist user information in a database.
- Authorization is the process of verifying whether an authenticated user has the privilege to access a requested resource.
- In an ASP.NET MVC application, you can authorize some specific users and roles by using the Authorize attribute.
- The ASP.NET MVC Framework provides different role providers that you can use to implement roles.
- When you create an MVC application using Visual Studio 2013, you need to create the membership database with a connection string.

### **10.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

#### **Tips:**

You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. How you will create a custom role provider?
2. How can you restrict all the action methods defined in a controller class?

## Session 11

### Security

#### **11.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **11.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe how to secure applications
- Define and describe different types of malicious attacks
- Explain how to protect a Web application against malicious attacks
- Explain the process of securing application data

#### **11.1.2 Teaching Skills**

To teach this session successfully, you must know about security issues in an ASP.NET MVC application. You must have knowledge about different types of malicious attacks and how to protect a Web application against malicious attacks. You should also be aware of the process of securing application data.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities:**

Follow the order given here during In-Class activities.

#### **Overview of the Session:**

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the security mechanism in an ASP.NET MVC application. They will learn about different types of malicious attacks and how to protect a Web application against such malicious attacks. This session will also discuss about the process of securing application data.

## **11.2 Introduction**

### **Slide 3**

Let us understand common malicious attacks on Web applications.

#### Malicious Attacks and their Preventions

- ◆ Web applications are deployed so that it can be accessed by different types of users.
- ◆ Some of these users might plan to carry out malicious attacks against these applications.
- ◆ Irrespective of the motive behind malicious attacks, while developing an application, you should ensure that a proper security mechanism is used to protect it from such attacks.
- ◆ Some of the common malicious attacks are:
  - ◊ Cross-site Scripting (XSS)
  - ◊ Cross-site Request Forgery (CSRF)
  - ◊ SQL Injection

© Aptech Ltd. Security/Session 11 3

Use slide 3 to explain that Web applications are deployed so that it can be accessed by different types of users. Some of these users might plan to carry out malicious attacks against these applications. Irrespective of the motive behind malicious attacks, while developing an application, they should ensure that a proper security mechanism is used to protect it from such attacks.

Tell them that some of the common malicious attacks are Cross-site Scripting (XSS), Cross-site Request Forgery (CSRF), and SQL Injection.

**Cross-site Scripting (XSS) attack** - XSS enables attackers to inject client-side script into Web pages viewed by the other users.

**Cross-site Request Forgery (CSRF) attack** - Cross-site Request Forgery (CSRF) is an attack where a malicious site sends a request to the users' data for the site, where the user was currently logged in. Typically, CSRF attacks are possible against Web sites that use cookies for authentication, because browsers send all relevant cookies to the destination Web site. However, CSRF attacks are not limited to exploiting cookies. For example, Basic and Digest authentication are also exposed to the attacks. After a user logs in with Basic or Digest authentication, the browser automatically sends the credentials until the session ends.

**SQL Injection attack** - SQL injection is a code injection technique, used to attack data-driven applications, in which malicious SQL statements are inserted into an entry field for execution.

## Slides 4 to 6

Let us understand cross-site scripting.

### Cross-site Scripting 1-5

- ◆ An XSS attack can be of the following two type:
  - ◆ **Persistent attack:** In this type of attack, the harmful code is stored in the database.
  - ◆ **Non-persistent attack:** In this type of attack, malicious code is not stored in the database; instead this code is rendered on the browser itself.
- ◆ To prevent XSS attacks, you should ensure that all HTML code that an application accepts from a user is encoded.
- ◆ HTML encoding is a process that converts potentially unsafe characters to their HTML-encoded equivalent.
- ◆ As a result, when you encode HTML inputs submitted to your application, the HTML engine does not interpret these characters as parts of the HTML markup and renders them as strings.

### Cross-site Scripting 2-5

- ◆ Apart from this, you also require ensuring that all HTML outputs of the application are encoded to prevent XSS attacks.
- ◆ Consider a scenario, where a user has gained access to the database of your application and inserted a script in a table.
- ◆ Following code snippet shows the script inserted in a table:

Snippet

```
<script>alert('Visit the gethacked website')</script>
```

- ◆ This code displays an alert box on the browser when the application accesses the script and sends it to the browser as response.

### Cross-site Scripting 3-5

- ◆ In such scenario, you need to encode the HTML output to prevent such attacks.
- ◆ Following code snippet shows encoding HTML output that converts the preceding markup:

Snippet

```
&lt;script&ampgtalert(&#39;Visit the gethacked website!&#39;)&lt;/script&ampgt
```

- ◆ This code will be displayed by the browser instead of executing the JavaScript code.

Using slides 4 to 6, explain the students about cross-site scripting (XSS).

Consider a scenario where a Web page of your application allows users to submit their e-mail IDs to receive weekly news updates and stores the e-mail IDs in a database table. However, the JavaScript code is embedded in the response data which is executed by the browser. As the browser downloaded the script from a trusted site, so it will not be able to identify the code as malicious and hence, provides no defense for such attacks.

Then, discuss some of the common situations where the Web application is susceptible to cross-site scripting attacks include failing to properly validate input, failing to encode output, and trusting the data retrieved from a shared database.

Tell them that an XSS attack can be of the two types that are as follows:

1. Persistent attack where the harmful code is stored in the database.
2. Non-persistent attack where malicious code is not stored in the database; instead this code is rendered on the browser itself.

Tell them that the two most important countermeasures to prevent cross-site scripting attacks are:

- Constrain input
- Encode output

Explain them that to prevent XSS attacks, they should ensure that all HTML code that an application accepts from a user is encoded. HTML encoding is a process that converts potentially unsafe characters to their HTML-encoded equivalent.

As a result, when they encode HTML inputs submitted to an application, the HTML engine does not interpret these characters as parts of the HTML markup and renders them as strings.

In slide 5, tell the students that they require to ensure that all HTML outputs of the application are encoded to prevent XSS attacks.

Ask them to consider a scenario, where a user has gained access to the database of an application and inserted a script in a table. Refer to the code snippet provided on the slide that shows the script inserted in a table.

Then, explain them that this code displays an alert box on the browser when the application accesses the script and sends it to the browser as response.

In slide 6, explain them that in such scenario, they need to encode the HTML output to prevent such attacks. Then, refer to the code that shows encoding HTML output that converts the preceding markup.

Explain them that this code will be displayed by the browser instead of executing the JavaScript code.

## Slide 7

Let us understand request validation feature in ASP.NET MVC framework.

**Cross-site Scripting 4-5**

- ◆ The ASP.NET MVC Framework provides a request validation feature that examines an HTTP request to check and prevent potentially malicious content.
- ◆ Following figure shows an error message that the browser displays once a form is submitted that contains HTML or JavaScript code to carry out an XSS attack:



© Aptech Ltd. Security/Session 11 7

Use slide 7 to explain the students that the ASP.NET MVC framework provides a request validation feature that examines an HTTP request to check and prevent potentially malicious content.

Consider a scenario, where if a user attempts to submit HTML or JavaScript code to carry out an XSS attack through the body, header, query string, or cookies of the request, the request validation feature will throw an `HttpRequestValidationException` exception.

Then, refer to the figure that shows an error message that the browser displays once a form is submitted that contains HTML or JavaScript code to carry out an XSS attack.

## Slide 8

Let us understand another built-in feature to prevent XSS attack.

**Cross-site Scripting 5-5**

- ◆ Another built-in feature that you can use in an ASP.NET MVC application to prevent XSS attack is, the Razor HTML encoding of the Razor view engine.
- ◆ When you use the @ symbol in a view to refer any data, the Razor HTML encoding feature automatically encodes the data and makes it safe to display as HTML on the browser.

© Aptech Ltd. Security/Session 11 8

Using slide 8, explain them that another built-in feature that they can use in an ASP.NET MVC application to prevent XSS attack is, the Razor HTML encoding of the Razor view engine.

Then, tell them that when they use the @ symbol in a view to refer any data, the Razor HTML encoding feature automatically encodes the data and makes it safe to display as HTML on the browser.

### Additional Information

To declare a helper with the @helper directive in a Razor view or in a .cshtml file in the application, you can use the following syntax:

```
@helper PageHeader(string pageTitle)
{
    <h1>@pageTitle</h1>
}
```

## Slides 9 to 13

Let us understand cross-site request forgery.

### Cross-site Request Forgery 1-5

- ◆ There are two approaches that you can use to block CSRF attacks, such as domain referrer and user-generated token.
- ◆ The domain referrer approach:
  - ◆ Checks whether or not an incoming request has a referrer header for your domain.
  - ◆ Enables you to ensure that the request has only been initiated from your application.
  - ◆ Prevents any requests coming from unknown sources.
- ◆ While developing an application, you can use the domain referrer approach by checking whether a user that posts data through a form is only from your application.
- ◆ You can use the Request.Url.Host property in an action method to retrieve the host name of your application.

### Cross-site Request Forgery 2-5

- ◆ You can retrieve the host name of the referrer by using the Request.UrlReferrer.Host property.
- ◆ Following code snippet shows comparison of both the host names and throws an exception if they do not match.

**Snippet**

```
string referrerHostName = Request.UrlReferrer.Host;
string appHostName = Request.Url.Host;
if (appHostName != referrerHostName)
{
    throw new UnauthorizedAccessException();
}
```

- ◆ In this code:

- ◆ The host name of the application and the referrer is retrieved and compared.
- ◆ If both the host names do not match, then the UnauthorizedAccessException exception is thrown.

## Cross-site Request Forgery 3-5

- ◆ You can also use the user-generated token approach to prevent a CSRF attack.
- ◆ This approach enables storing a user-generated token using a HTML hidden field in the user session.
- ◆ Thereafter, for each incoming request from a user, you can verify whether or not the submitted token is valid.
- ◆ ASP.NET MVC Framework provides a set of helpers that you can use to detect and block any CSRF attacks.
- ◆ This can be done by creating a user-generated token, which is passed between the view and the controller and verified on each request.
- ◆ One of such helper is the `@Html.AntiForgeryToken()` method that allows adding a hidden HTML field in a page that the controller will verify at each request.

## Cross-site Request Forgery 4-5

- ◆ Following code snippet shows how to add the `@Html.AntiForgeryToken()` to a form in a view:

**Snippet**

```
<form action="/user/register" method="post">
@Html.AntiForgeryToken()
...
</form>
```

- ◆ This code will generate an encrypted value for a hidden input and sends back a cookie named, `_RequestVerificationToken`, with the same encrypted value.
- ◆ Following code snippet shows the generated encrypted value for a hidden input:

**Snippet**

```
<input type="hidden" value="012837udny31w90hjhf7u">
```

In addition, you need to add the `[ValidateAntiForgeryToken]` filter to the action method that processes the posted form.

## Cross-site Request Forgery 5-5

- Following code snippet shows using the [ValidateAntiForgeryToken] filter:

Snippet

```
[ValidateAntiForgeryToken]
public ActionResult RegisterUser(RegisterUserModel model)
{
    /*Code to register and return an ActionResult object*/
}
```

- In this code:
  - The [ValidateAntiForgeryToken] filter checks for the \_RequestVerificationToken cookie.
  - The \_RequestVerificationToken form value when a form is posted.
  - If both the values are present, the filter matches.
  - A successful match results in the execution of the RegisterUser action. Else, the filter throws an exception.

Using slides 9 to 13, explain cross-site request forgery.

In slide 9, explain to the students that there are two approaches that they can use to block CSRF attacks, such as domain referrer and user-generated token.

The domain referrer approach checks whether or not an incoming request has a referrer header for a domain. It enables them to ensure that the request has only been initiated from an application. In addition, it prevents any requests coming from unknown sources. Then, refer to the code that shows passing a collection of model objects to a view.

Tell them that while developing an application, they can use the domain referrer approach by checking whether a user that posts data through a form is only from an application. They can use the Request.Url.Host property in an action method to retrieve the host name of an application.

In slide 10, explain them that they can retrieve the host name of the referrer by using the Request.UrlReferrer.Host property.

Then, refer to the code that shows comparison of both the host names and throws an exception if they do not match.

Then, tell them that in the code, the host name of the application and the referrer is retrieved and compared. If both the host names do not match, then the UnauthorizedAccessException exception is thrown.

In slide 11, explain to the students that they can also use the user-generated token approach to prevent a CSRF attack. This approach enables storing a user-generated token using a HTML hidden field in the user session. Thereafter, for each incoming request from a user, they can verify whether or not the submitted token is valid.

Then, explain them that ASP.NET MVC framework provides a set of helpers that they can use to detect and block any CSRF attacks. This can be done by creating a user-generated token, which is passed between the view and the controller and verified on each request. One of such helper is the `@Html.AntiForgeryToken()` method that allows adding a hidden HTML field in a page that the controller will verify at each request.

In slide 12, refer to the first code that shows how to add the `Html.AntiForgeryToken()` to a form in a view. Explain them that this code will generate an encrypted value for a hidden input and sends back a cookie named, `_RequestVerificationToken`, with the same encrypted value.

Then, refer to the second code that shows the generated encrypted value for a hidden input. Tell them that in addition, they need to add the `[ValidateAntiforgeryToken]` filter to the action method that processes the posted form.

Use slide 13 to refer to the code that shows use of the `[ValidateAntiforgeryToken]` filter. Then, explain to the students that in this code, the `[ValidateAntiforgeryToken]` filter checks for the `_RequestVerificationToken` cookie.

The `_RequestVerificationToken` form value when a form is posted. If both the values are present, the filter matches. A successful match results in the execution of the `RegisterUser` action. Otherwise, the filter throws an exception.

## Slides 14 to 16

Let us understand SQL Injection.

### SQL Injection 1-3

- ◆ SQL injection is a form of attack in which a user posts SQL code to an application.
- ◆ It creates an SQL statement that will execute with malicious intent.
- ◆ Using a SQL injection, a user can store and update malicious data, access sensitive data, or delete any existing data of the application.
- ◆ Following code snippet shows a dynamic SQL query that accesses data from the User table based on a userName form value:

Snippet

```
String UserName= context.Request.Form["userName"];
String Query = "select * from User where User_name ='"+UserName +"'";
```

© Aptech Ltd. Security/Session 11 14

### SQL Injection 2-3

- ◆ In the code:
  - ◆ The variable named, UserName stores the value of the userName field of a submitted form.
  - ◆ The Query variable constructs a dynamic SQL query by adding the UserName variable to retrieve those records from the User table whose User\_Name field matches the value of the UserName variable.
  - ◆ When a user submits a user name, this code will execute and return all data from the User table that matches the user name.
  - ◆ A user who identifies the vulnerabilities for SQL injection in the code, can carry out an attack by submitting a query in the User name text field of the form.

© Aptech Ltd. Security/Session 11 15

### SQL Injection 3-3

- Following code snippet shows submitting a query in the User name text field of a form:

**Snippet**

```
drop table User;
```

- On submitting the form, following query will be generated dynamically:

**Snippet**

```
select * from User where User_name = ''; drop table User --'
```

- This code will execute in the following two phases:
  - In the first phase, the query will attempt to retrieve records from the User table.
  - Then, in the next phase, the query will attempt to drop the User table. The -- symbol after the drop statement in the query specifies that the remaining part of the query does not required to be executed.

Using slides 14 to 16, explain SQL injection.

SQL injection is a form of attack in which a user posts SQL code to an application. It creates an SQL statement that will execute with malicious intent. Using a SQL injection, a user can store and update malicious data, access sensitive data, or delete any existing data of the application.

Common situations that can make your data access code susceptible to SQL injection attacks include:

- Weak input validation.
- Dynamic construction of SQL statements without the use of type-safe parameters.
- Use of over-privileged database logins.

Then, refer to the code that shows a dynamic SQL query that accesses data from the User table based on a userName form value.

In slide 15, explain to the students that in the code, the variable named, UserName stores the value of the userName field of a submitted form. The Query variable constructs a dynamic SQL query by adding the UserName variable to retrieve those records from the User table whose User\_Name field matches the value of the UserName variable. When a user submits a user name, this code will execute and return all data from the User table that matches the user name.

Then, tell them that a user who identifies the vulnerabilities for SQL injection in the code, can carry out an attack by submitting a query in the `User` name text field of the form.

In slide 16, refer to the first code that shows how to submit a query in the user name text field of a form. Then, refer to the second code and tell them that on submitting the form, this query will be generated dynamically.

Then, explain to the students that this code will execute in the following two phases:

- In the first phase, the query will attempt to retrieve records from the `User` table.
- Then, in the next phase, the query will attempt to drop the `User` table. The `--` symbol after the drop statement in the query specifies that the remaining part of the query does not required to be executed.

### Additional Information

To prevent SQL injection attacks, you need to:

- **Constrain and sanitize input data** - Check for known good data by validating for type, length, format, and range.
- **Use type-safe SQL parameters for data access** - You can use these parameters with stored procedures or dynamically constructed SQL command strings. Parameter collections such as `SqlParameterCollection` provide type checking and length validation.

If you use a parameters collection, input is treated as a literal value, and SQL Server does not treat it as executable code. An additional benefit of using a parameters collection is that one can enforce type and length checks. Values outside of the range trigger an exception. This is a good example of defense in depth.

- **Use an account that has restricted permissions in the database** - Ideally, you should only grant execute permissions to selected stored procedures in the database and provide no direct table access.
- **Avoid disclosing database error information** - In the event of database errors, make sure you do not disclose detailed error messages to the user.

## Slides 17 to 20

Let us understand deferred validation and unvalidated requests.

### Deferred Validation and Unvalidated Requests 1-4

- ◆ In an ASP.NET MVC application, you can enable deferred validation by configuring it in the Web.config file of the application.
- ◆ For this, you need to set the **requestValidationMode** attribute of the `<httpRuntime>` element to 4.5.
- ◆ Following code snippet allows setting the requestValidationMode attribute of the `<httpRuntime>` element:

**Snippet**

```
<httpRuntime requestValidationMode="4.5" />
```

- ◆ Once you configure deferred validation, you can validate data of a particular field in a form.
- ◆ You can do this when you expect a field that can contain HTML code or scripts that the request validation process checks for potential XSS attack.

© Aptech Ltd.

Security/Session 11

17

### Deferred Validation and Unvalidated Requests 2-4

- ◆ One of the approaches to instruct the request validation process not to validate the data of a particular field is by using the **Unvalidated** property of the **HttpRequest** class.
- ◆ Following code snippet shows using the **Unvalidated** property to access the content of the `code` field:

**Snippet**

```
var c = context.Request.Unvalidated.Form["code"];
```

- ◆ Apart from this, you can also instruct the request validation process not to perform validation at the controller level or at action method level.
- ◆ You can do this by setting the value of the **ValidateInput** attribute to false.

© Aptech Ltd.

Security/Session 11

18

## Deferred Validation and Unvalidated Requests 3-4

- Following code snippet shows disabling the validation of form data processed by the SubmitQuery() action method:

**Snippet**

```
[HttpPost]
[ValidateInput(false)]
public ActionResult SubmitQuery(QueryModel model)
```

- In this code, the value of the ValidateInput attribute is set to false to disable the validation of form data processed by the SubmitQuery() action method.
- You can also disable validation for a field for a strongly-typed view by adding the AllowHtml attribute to the corresponding property of the model.

## Deferred Validation and Unvalidated Requests 4-4

- Following code snippet shows using the AllowHtml attribute:

**Snippet**

```
[AllowHtml]
[Required]
[DisplayName = "Code Snippet"]
public string CodeSnippet{ get; set; }
```

- In this code:
  - The AllowHtml attribute is applied to the CodeSnippet property.
  - As a result, when the user adds HTML markups and script in the CodeSnippet field and submits the form, the request validation process will not report an error, as it will do for other fields.

Using slides 17 to 20, explain deferred validation and unvalidated requests.

Use slide 17 to explain to the students that in an ASP.NET MVC application, they can enable deferred validation by configuring it in the Web.config file of the application. For this, they need to set the requestValidationMode attribute of the <httpRuntime> element to 4.5.

Then, refer to the code that shows how to set the `requestValidationMode` attribute of the `<httpRuntime>` element.

Tell them once we set request validation mode to 4.5, request validation will be triggered for only request data which is accessed by code and only when it is actually executed by code. For example, if `Request.Form["comment"]` is accessed within button click event, then only **comment** field will be validated when button click event fire, if user does not click the button, then it will not validate.

Then, explain the new feature introduced in request validation is that access to unvalidated data. In the earlier versions of ASP.NET, it was not possible to bypass the request validation, however, you were able to disable it.

Thus, to access unvalidated data without disabling the request validation, ASP.NET introduced **Unvalidated** property. Explain them that once they configure deferred validation, they can validate data of a particular field in a form. They can do this when they expect a field that can contain HTML code or scripts that the request validation process checks for potential XSS attack.

In slide 18, explain the approaches to instruct the request validation process not to validate the data of a particular field is by using the `Unvalidated` property of the `HttpRequest` class.

Then, refer to the code that shows use of the `Unvalidated` property to access the content of the code field.

Explain them that apart from this, they can also instruct the request validation process not to perform validation at the controller level or at action method level. They can do this by setting the value of the `ValidateInput` attribute to false.

In slide 19, refer to the code that shows how to disable the validation of form data processed by the `SubmitQuery()` action method.

Then, explain them that in this code, the value of the `ValidateInput` attribute is set to `false` to disable the validation of form data processed by the `SubmitQuery()` action method.

Mention that they can also disable validation for a field for a strongly-typed view by adding the `AllowHtml` attribute to the corresponding property of the model.

In slide 20, refer to the code that shows use of the `AllowHtml` attribute.

Explain them that in this code, the `AllowHtml` attribute is applied to the `CodeSnippet` property. As a result, when the user adds HTML markups and script in the `CodeSnippet` field and submits the form, the request validation process will not report an error, as it will do for other fields.

**Slide 21**

Let us understand data security.

### Data Security

- ◆ All organizations need to handle sensitive data.
- ◆ This data can be either present in storage or may be exchanged between different entities within and outside the organization over a network.
- ◆ To avoid misuse of such data, there should be some security mechanism that can ensure confidentiality and integrity of the data.
- ◆ One of the commonly used techniques to secure such sensitive data is known as encryption.

© Aptech Ltd. Security/Session 11 21

Using slide 21, explain the students that all organizations need to handle sensitive data.

This data can be either present in storage or may be exchanged between different entities within and outside the organization over a network. Consider a scenario where you are working as a CEO of a company. So, while travelling, you need to access performance appraisal reports of the top management of the company. These reports contain data that are confidential and are stored in the company's server. Such data is often prone to misuse either intentionally with malicious intent or unintentionally.

To avoid misuse of such data, there should be some security mechanism that can ensure confidentiality and integrity of the data. One of the commonly used techniques to secure such sensitive data is known as encryption.

## Slide 22

Let us understand encryption and decryption.

### Encryption and Decryption

- ◆ Encryption:
  - ❖ Is a technique that ensures data confidentiality.
  - ❖ Converts data in plain text to cipher (secretly coded) text.
- ◆ Decryption is a process that converts the encrypted cipher text back to the original plain text.
- ◆ Following figure shows the process of encryption and decryption of a password as an example:

```
graph LR; A[pass@123] --> B[Encryption]; B --> C["Q@#123%%^Sdd*"]; C --> D[Decryption]; D --> E[pass@123]
```

- ◆ In this figure, the plain text, pass@123 is encrypted to a cipher text. The cipher text is decrypted back to the original plain text.

© Aptech Ltd. Security/Session 11 22

In slide 22, explain the students that encryption is a technique that ensures data confidentiality. It converts data in plain text to cipher (secretly coded) text.

Decryption is a process that converts the encrypted cipher text back to the original plain text.

Then, refer to the figure that shows the process of encryption and decryption of a password as an example.

Explain them that in the figure, the plain text, pass@123 is encrypted to a cipher text. The cipher text is decrypted back to the original plain text.

## Slides 23 to 26

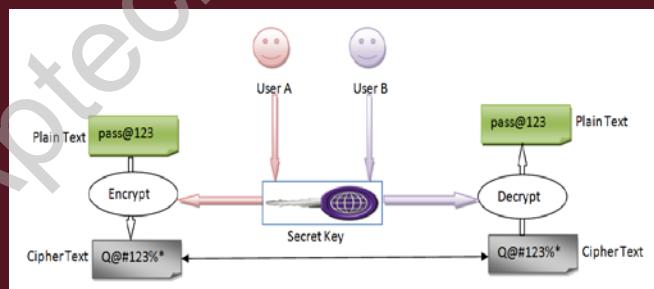
Let us understand the types of encryption and decryption.

### Types of Encryption and Decryption 1-4

- ◆ There are two types of encryption and decryption, such as symmetric and asymmetric.
- ◆ Symmetric encryption, or secret key encryption, uses a single key, known as the secret key both to encrypt and decrypt data.
- ◆ Following steps outline an example usage of symmetric encryption:
  - ❖ User A uses a secret key to encrypt a plain text to cipher text.
  - ❖ User A shares the cipher text and the secret key with User B.
  - ❖ User B uses the secret key to decrypt the cipher text back to the original plain text.

### Types of Encryption and Decryption 2-4

- ◆ Following figure shows the symmetric encryption and decryption process:

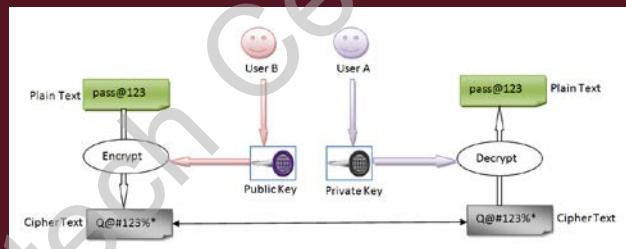


### Types of Encryption and Decryption 3-4

- ◆ The asymmetric encryption uses a pair of public and private key to encrypt and decrypt data.
- ◆ Following steps outline an example usage of asymmetric encryption:
  - ◆ User A generates a public and private key pair.
  - ◆ User A shares the public key with User B.
  - ◆ User B uses the public key to encrypt a plain text to cipher text.
  - ◆ User A uses the private key to decrypt the cipher text back to the original plain text.

### Types of Encryption and Decryption 4-4

- ◆ Following figure shows the asymmetric encryption and decryption process:



Using slides 23 to 26, explain the students that there are two types of encryption and decryption, such as symmetric and asymmetric.

Symmetric encryption or secret key encryption, uses a single key, known as the secret key both to encrypt and decrypt data.

Then, explain about the following steps that outlines an example usage of symmetric encryption:

- User A uses a secret key to encrypt a plain text to cipher text.
- User A shares the cipher text and the secret key with User B.
- User B uses the secret key to decrypt the cipher text back to the original plain text

In slide 24, refer to the figure that shows the symmetric encryption and decryption process.

In slide 25, explain to the students that the asymmetric encryption uses a pair of public and private key to encrypt and decrypt data.

Then, explain about the following steps that outlines an example usage of asymmetric encryption:

- User A generates a public and private key pair.
- User A shares the public key with User B.
- User B uses the public key to encrypt a plain text to cipher text.
- User A uses the private key to decrypt the cipher text back to the original plain text.

**Tips:**

Private keys used to decrypt data in asymmetric encryption should be stored using a secured mechanism. To achieve this, you can use a key container that is a logical structure to securely store asymmetric keys.

In slide 26, refer to the figure that shows the asymmetric encryption and decryption process.

## Slide 27

Let us understand how to use symmetric encryption.

### Using Symmetric Encryption 1-13

- ◆ You need to use one of the symmetric encryption implementation classes of the .NET Framework to perform symmetric encryption.
- ◆ The first step to perform symmetric encryption is to create the symmetric key.
- ◆ When you use the default constructor of the symmetric encryption classes, such as RijndaelManaged and AesManaged, a key and IV is automatically generated.
- ◆ The generated key and the IV can be accessed as byte arrays using the Key and IV properties of the encryption class.

© Aptech Ltd. Security/Session 11 27

Using slide 27, explain symmetric encryption.

Tell them that cryptography is the process of creating and managing keys. Thus, encryption involves generation of keys.

Symmetric algorithms require the creation of a key and an Initialization Vector (IV). The key must be kept secret from anyone who should not decrypt your data. The IV does not have to be secret, but should be changed for each session.

Asymmetric algorithms require the creation of a public key and a private key. The public key can be made public to anyone, while the private key must know only by the party who will decrypt the data encrypted with the public key

Explain the student that they need to use one of the symmetric encryption implementation classes of the .NET Framework to perform symmetric encryption. The first step to perform symmetric encryption is to create the symmetric key. The symmetric encryption classes supplied by the .NET Framework require a key and a new Initialization Vector (IV) to encrypt and decrypt data.

Then, explain them that when they use the default constructor of the symmetric encryption classes, such as `RijndaelManaged` and `AesManaged`, a key and IV is automatically generated. The generated key and the IV can be accessed as byte arrays using the `Key` and `IV` properties of the encryption class.

**Tips:**

To make symmetric encryption more secure, an Initialization Vector (IV) can be used with the secret key. An IV is a random number that generates different sequence of encrypted text for identical sequence of text present in the plain text. When you use an IV with a key to symmetrically encrypt data, you will need the same IV and key to decrypt data.

## Slides 28 and 29

Let us understand how to create the symmetric key.

### Using Symmetric Encryption 2-13

- Following code snippet shows creating a symmetric key and IV using the RijndaelManaged class:

```
using System;
using System.Security.Cryptography;
using System.Text;
...
RijndaelManaged symAlgo = new RijndaelManaged();
Console.WriteLine("Generated key: {0}, \nGenerated IV: {1}",
Encoding.Default.GetString(symAlgo.Key),
Encoding.Default.GetString(symAlgo.IV));
```

Snippet

- This code uses the default constructor of the RijndaelManaged class to generate a symmetric key and IV.
- The Key and IV properties are accessed and printed as strings using the default encoding to the console.

© Aptech Ltd. Security/Session 11 28

### Using Symmetric Encryption 3-13

- Following figure shows the output of the preceding code:

The symmetric encryption classes, such as RijndaelManaged also provide the GenerateKey() and GenerateIV() methods that you can use to generate keys and IVs.

© Aptech Ltd. Security/Session 11 29

Using slides 28 and 29, explain the code that shows how to create a symmetric key and IV using the RijndaelManaged class.

Explain them that this code uses the default constructor of the `RijndaelManaged` class to generate a symmetric key and IV. The `Key` and `IV` properties are accessed and printed as strings using the default encoding to the console.

In slide 29, refer to the figure that shows the output of the code.

Explain them that the symmetric encryption classes, such as `RijndaelManaged` also provide the `GenerateKey()` and `GenerateIV()` methods that they can use to generate keys and IVs.

## Slides 30 to 34

Let us understand how to generate keys and IVs.

### Using Symmetric Encryption 4-13

- Following code snippet shows using the GenerateKey() and GenerateIV() methods to generate keys and IVs:

Snippet

```
using System;
using System.Security.Cryptography;
using System.Text;
. . .
RijndaelManagedsymAlgo = new RijndaelManaged();
RijndaelManagedsymAlgo.GenerateKey();
RijndaelManagedsymAlgo.GenerateIV();
byte[] generatedKey = RijndaelManagedsymAlgo.Key;
byte[] generatedIV = RijndaelManagedsymAlgo.IV;
Console.WriteLine("Generated key through GenerateKey(): {0}",
\nGenerated IV through GenerateIV(): {1}",
Encoding.Default.GetString(generatedKey),
Encoding.Default.GetString(generatedIV));
```

- This code creates a RijndaelManaged object and then, calls the GenerateKey() and GenerateIV() methods to generate a key and an IV. The Key and the IV properties are then, accessed and printed as strings using the default encoding to the console.

### Using Symmetric Encryption 5-13

- The symmetric encryption classes of the ASP.NET MVC Framework provides the CreateEncryptor() method that returns an object of the ICryptoTransform interface.
- The ICryptoTransform object is responsible for transforming the data based on the algorithm of the encryption class.
- Once you have obtained an ICryptoTransform object, you can use the CryptoStream class to perform the encryption.
- The CryptoStream class acts as a wrapper of a stream-derived class, such as FileStream, MemoryStream, and NetworkStream.
- A CryptoStream object operates in one of the following two modes defined by the CryptoStreamMode enumeration:
  - First is Write mode that allows writing operation on the underlying stream, and you can use this mode to perform encryption.
  - Second is the Read mode that allows reading operation on the underlying stream.

## Using Symmetric Encryption 6-13

- ◆ You can create a CryptoStream object by calling the constructor that accepts the following three parameters:
  - ❖ The underlying stream object
  - ❖ The ICryptoTransform object
  - ❖ The mode defined by the CryptoStreamMode enumeration
- ◆ After creating the CryptoStream object, you can call the Write() method to write the encrypted data to the underlying stream.
- ◆ Following code snippet encrypts data using the RijndaelManaged class and writes the encrypted data to a file:

Snippet

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;
class SymmetricEncryptionDemo
{
    static void EncryptData(String plainText, RijndaelManaged algo)
    {
        // Your code here
    }
}
```

## Using Symmetric Encryption 7-13

- ◆ Following code snippet encrypts data using the RijndaelManaged class and writes the encrypted data to a file:

Snippet

```
{
    byte[] plaindataArray =
    ASCIIEncoding.ASCII.GetBytes(plainText);

    ICryptoTransform transform=algo.CreateEncryptor();
    using (var fileStream = new
    FileStream("D:\\CipherText.txt", FileMode.OpenOrCreate,
    FileAccess.Write))
    {
        using (var cryptoStream = new CryptoStream(fileStream,
        transform, CryptoStreamMode.Write))
        {
            cryptoStream.Write(plaindataArray, 0,
            plaindataArray.GetLength(0));
            Console.WriteLine("Encrypted data written to:
D:\\CipherText.txt");
        }
    }
}
```

## Using Symmetric Encryption 8-13

Snippet

```

static void Main()
{
    RijndaelManaged symAlgo = new RijndaelManaged();
    Console.WriteLine("Enter data to encrypt.");
    string dataToEncrypt = Console.ReadLine();
    EncryptData(dataToEncrypt, symAlgo);
}
}
```

♦ In this code:

- ◆ The Main() method creates a RijndaelManaged object and passes it along with the data to encrypt to the EncryptData() method.
- ◆ In the EncryptData() method, the call to the CreateEncryptor() method creates the ICryptoTransform object.
- ◆ Then, a FileStream object is created to write the encrypted text to the CipherText.txt file.
- ◆ Next, the CryptoStream object is created and its Write() method is called.

Using slides 30 to 34, explain them the methods provided by RijndaelManaged symmetric encryption class.

In slide 30, refer to the code displayed on the slide that shows use of the GenerateKey() and GenerateIV() methods to generate keys and IVs.

Then, explain them that this code creates a RijndaelManaged object and then, calls the GenerateKey() and GenerateIV() methods to generate a key and an IV. The Key and the IV properties are then, accessed and printed as strings using the default encoding to the console.

In slide 31, tell the students that the symmetric encryption classes of the ASP.NET MVC framework provides the CreateEncryptor() method that returns an object of the ICryptoTransform interface. The ICryptoTransform object is responsible for transforming the data based on the algorithm of the encryption class.

Explain them that once they have obtained an ICryptoTransform object, they can use the CryptoStream class to perform the encryption. The CryptoStream class acts as a wrapper of a stream-derived class, such as FileStream, MemoryStream, and NetworkStream. A CryptoStream object operates in one of the following two modes defined by the CryptoStreamMode enumeration:

- First is Write mode that allows writing operation on the underlying stream, and they can use this mode to perform encryption.
- Second is the Read mode that allows reading operation on the underlying stream.

In slide 32, explain that they can create a `CryptoStream` object by calling the constructor that accepts the following three parameters:

- The underlying stream object
- The `ICryptoTransform` object
- The mode defined by the `CryptoStreamMode` enumeration

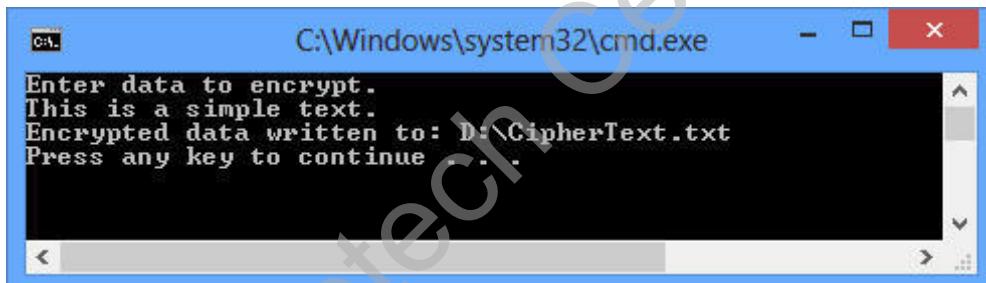
Tell them that after creating the `CryptoStream` object, they can call the `Write( )` method to write the encrypted data to the underlying stream.

Then, refer to the code on slides 32 and 33 that encrypts data using the `RijndaelManaged` class and write the encrypted data to a file.

Using slide 34, explain the students that in this code, the `Main( )` method creates a `RijndaelManaged` object and passes it along with the data to encrypt to the `EncryptData( )` method. In the `EncryptData( )` method, the call to the `CreateEncryptor( )` method creates the `ICryptoTransform` object.

Then, a `FileStream` object is created to write the encrypted text to the `CipherText.txt` file. Next, the `CryptoStream` object is created and its `Write( )` method is called.

The output of the code is as follows:



A screenshot of a Windows Command Prompt window titled "C:\Windows\system32\cmd.exe". The window contains the following text:  
Enter data to encrypt.  
This is a simple text.  
Encrypted data written to: D:\CipherText.txt  
Press any key to continue . . .

**Slide 35**

Let us understand how to decrypt data.

### Using Symmetric Encryption 9-13

- ◆ On the other hand, to decrypt data:
  - ❖ You can use the same symmetric encryption class, key, and IV used for encrypting the data.
  - ❖ You call the CreateDecryptor() method to obtain a ICryptoTransform object that will perform the transformation.
  - ❖ You then, need to create the CryptoStream object in Read mode and initialize a StreamReader object with the CryptoStream object.
  - ❖ Finally, you need to call the ReadToEnd() method of the StreamReader that returns the decrypted text as a string.
- ◆ Following code snippet shows creating a program that performs both encryption and decryption:

Snippet

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;
class SymmetricEncryptionDemo
{
    static void EncryptData(String plainText, RijndaelManaged algo)
    {
        String cipherText = null;
        // Create a new instance of the RijndaelManaged class
        // and set its properties.
        RijndaelManaged rijndael = new RijndaelManaged();
        rijndael.Key = Encoding.UTF8.GetBytes("This is a secret key");
        rijndael.IV = Encoding.UTF8.GetBytes("This is a secret IV");
        rijndael.Mode = CipherMode.CBC;
        rijndael.Padding = PaddingMode.PKCS7;
        rijndael.BlockSize = 128;
        rijndael.KeySize = 256;
        rijndael.GenerateIV();
        rijndael.GenerateKey();
        rijndael.Padding = PaddingMode.PKCS7;
        rijndael.Mode = CipherMode.CBC;
```

© Aptech Ltd.

Security/Session 11

35

In slide 35, explain the students to decrypt data, they can use the same symmetric encryption class, key, and IV used for encrypting the data.

Tell them that they can call the CreateDecryptor( ) method to obtain an ICryptoTransform object that will perform the transformation. Then, they need to create the CryptoStream object in Read mode and initialize a StreamReader object with the CryptoStream object.

Finally, they need to call the ReadToEnd( ) method of the StreamReader that returns the decrypted text as a string

Then, refer to the code that shows how to create a program that performs both encryption and decryption.

**Slides 36 to 39**

Let us understand how to perform both encryption and decryption.

**Using Symmetric Encryption 10-13****Snippet**

```
byte[] plaindataArray = ASCIIEncoding.ASCII.GetBytes(plainText);
    ICryptoTransform transform = algo.CreateEncryptor();
    using (var fileStream = new FileStream("D:\\CipherText.txt",
    FileMode.OpenOrCreate, FileAccess.Write))
    {
        using (var cryptoStream = new CryptoStream(fileStream, transform,
        CryptoStreamMode.Write))
        {
            cryptoStream.Write(plaindataArray, 0, plaindataArray.GetLength(0));
            Console.WriteLine("Encrypted data written to: D:\\CipherText.txt");
        }
    }
static void DecryptData(RijndaelManaged algo)
{
    ICryptoTransform transform = algo.CreateDecryptor();
    using (var fileStream = new FileStream("D:\\CipherText.txt",
    FileMode.Open, FileAccess.Read))
    using (CryptoStream = new CryptoStream(fileStream, transform,
    CryptoStreamMode.Read))
    {
}
```

**Using Symmetric Encryption 11-13****Snippet**

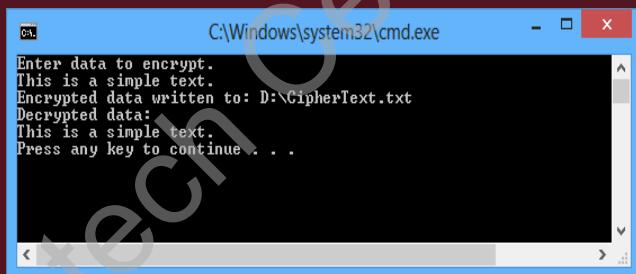
```
using (var streamReader = new StreamReader(cryptoStream))
{
    string decryptedData = streamReader.ReadToEnd();
    Console.WriteLine("Decrypted data: \n{0}", decryptedData);
}
}
static void Main()
{
    RijndaelManaged symAlgo = new RijndaelManaged();
    Console.WriteLine("Enter data to encrypt.");
    string dataToEncrypt = Console.ReadLine();
    EncryptData(dataToEncrypt, symAlgo);
    DecryptData(symAlgo);
}
```

## Using Symmetric Encryption 12-13

- ◆ In the code:
  - ❖ The Main() method creates a RijndaelManaged object and passes it along with the data to encrypt the EncryptData() method. The encrypted data is saved to the CipherText.txt file.
  - ❖ The Main() method calls the DecryptData() method passing the same RijndaelManaged object created for encryption.
  - ❖ The DecryptData() method creates the ICryptoTransform object and uses a FileStream object to read the encrypted data from the file.
  - ❖ The CryptoStream object is created in the Read mode initialized with the FileStream and ICryptoTransform objects.
  - ❖ A StreamReader object is created by passing the CryptoStream object to the constructor.
  - ❖ The ReadToEnd() method of the StreamReader object is called.

## Using Symmetric Encryption 13-13

- ◆ Following figure shows the decrypted text returned by the ReadToEnd() method:



Using slides 36 to 39, refer to the code that shows how to create a program that performs both encryption and decryption.

In slide 38, refer to the code displayed on slides 36 and 37 and explain to the students that in this code:

- The Main() method creates a RijndaelManaged object and passes it along with the data to encrypt the EncryptData() method. The encrypted data is saved to the CipherText.txt file.
- The Main() method calls the DecryptData() method passing the same RijndaelManaged object created for encryption.
- The DecryptData() method creates the ICryptoTransform object and uses a FileStream object to read the encrypted data from the file.
- The CryptoStream object is created in the Read mode initialized with the FileStream and ICryptoTransform objects.
- A StreamReader object is created by passing the CryptoStream object to the constructor.

Using slide 39, refer to the output figure that shows the decrypted text returned by the ReadToEnd() method.

## Slide 40

Let us understand how to use asymmetric encryption.

### Using Asymmetric Encryption 1-12

- ◆ The System.Security.Cryptography namespace provides the RSACryptoServiceProvider class that you can use to perform asymmetric encryption.
- ◆ When you call the default constructor of the RSACryptoServiceProvider class, a new public-private key pair is automatically generated.
- ◆ After you create a new instance of this class, you can export the key information by using one of the following methods:
  - ❖ **ToXMLString()**: Returns an XML representation of the key information.
  - ❖ **ExportParameters()**: Returns an RSAParameters structure that holds the key information.
- ◆ Both of these methods accept a Boolean value.
- ◆ A false value indicates that the method should return only the public key information, while a true value indicates that the method should return information of both the public and private keys.

© Aptech Ltd. Security/Session 11 40

Use slide 40 to explain the students that the System.Security.Cryptography namespace provides the RSACryptoServiceProvider class that they can use to perform asymmetric encryption. When they call the default constructor of the RSACryptoServiceProvider class, a new public-private key pair is automatically generated.

Tell them that after they create a new instance of this class, they can export the key information by using one of the two methods. First one is `ToXMLString()` method that returns an XML representation of the key information. The second one is the `ExportParameters()` method that returns an RSAParameters structure that holds the key information.

Both of these methods accept a Boolean value. A `false` value indicates that the method should return only the public key information, while a `true` value indicates that the method should return information of both the public and private keys.

#### Additional Information:

In the .NET Framework, the RSACryptoServiceProvider class derives from the RSA class to provide an implementation of the RSA algorithm.

The RSA algorithm was designed in 1977 by Ron Rivest, Adi Shamir, and Leonard Adleman and till now is the most widely adopted algorithm to perform asymmetric encryption and decryption. This algorithm functions in three steps: key generation, encryption, and decryption. The RSA algorithm generates a public key as a product of two large prime numbers, along with a public (or encryption) value. The algorithm generates a private key as a product of the same two large prime numbers, along with a private (or decryption) value. The public key is used to perform encryption while the private key is used to perform decryption.

For Aptech Centre Use Only

## Slides 41 and 42

Let us understand the asymmetric encryption.

### Using Asymmetric Encryption 2-12

- Following code snippet shows creating and initializing an RSACryptoServiceProvider object and then, exports the public key in XML format:

Snippet

```
using System;
using System.Security.Cryptography;
using System.Text;
...
RSACryptoServiceProvider rSAKeyGenerator = new
RSACryptoServiceProvider();
string publicKey = rSAKeyGenerator.ToXmlString(false);
```

- This code creates and initializes an RSACryptoServiceProvider object and then, export the public key in XML format.

### Using Asymmetric Encryption 3-12

- Following code snippet shows creating and initializing an RSACryptoServiceProvider object:

Snippet

```
using System;
using System.Security.Cryptography;
using System.Text;
...
RSACryptoServiceProvider rSAKeyGenerator = new
RSACryptoServiceProvider();
RSAParameters rSAKeyInfo = rSAKeyGenerator.ExportParameters(true);
```

- This code creates and initializes an RSACryptoServiceProvider object and then, exports both the public and private keys as an RSAParameters structure.
- Now, to encrypt data, you need to create a new instance of the RSACryptoServiceProvider class and call the ImportParameters() method to initialize the instance with the public key information exported to an RSAParameters structure.

Using slides 41 and 42, refer to the code that shows how to create and initialize an RSACryptoServiceProvider object and then, exports the public key in XML format.

Then, explain them that this code creates and initializes an `RSACryptoServiceProvider` object and then, exports both the public and private keys as an `RSAParameters` structure.

Explain them that now, to encrypt data, they need to create a new instance of the `RSACryptoServiceProvider` class and call the `ImportParameters()` method to initialize the instance with the public key information exported to an `RSAParameters` structure.

For Aptech Centre Use Only

## Slide 43

Let us understand how to initialize the object with public key exported to RSAParameters structure.

### Using Asymmetric Encryption 4-12

- Following code snippet shows initializing an RSACryptoServiceProvider object with the public key exported to an RSAParameters structure:

Snippet

```
using System;
using System.Security.Cryptography;
using System.Text;
...
...
RSACryptoServiceProvider rSAKeyGenerator = new
RSACryptoServiceProvider();
RSAParameters rSAKeyInfo = rSAKeyGenerator.ExportParameters(false);
RSACryptoServiceProvider rsaEncryptor= new RSACryptoServiceProvider();
rsaEncryptor.ImportParameters(rSAKeyInfo);
```

- If the public key information is exported to XML format, you need to call the FromXmlString() method to initialize the RSACryptoServiceProvider object with the public key.

In slide 43, refer to the code that shows how to initialize an RSACryptoServiceProvider object with the public key exported to an RSAParameters structure.

Then, explain them that if the public key information is exported to XML format, they need to call the FromXmlString() method to initialize the RSACryptoServiceProvider object with the public key.

## Slide 44

Let us understand how to initialize the public key.

### Using Asymmetric Encryption 5-12

- ◆ Following code snippet shows initializing the RSACryptoServiceProvider object with the public key:

Snippet

```
using System;
using System.Security.Cryptography;
using System.Text;
...
RSACryptoServiceProvider rSAKeyGenerator = new
RSACryptoServiceProvider();
string publicKey = rSAKeyGenerator.ToXmlString(false);
RSACryptoServiceProvider rsaEncryptor= new RSACryptoServiceProvider();
rsaEncryptor.FromXmlString(publicKey);
```

- ◆ Once you have initialized the RSACryptoServiceProvider object with the public key, you can encrypt data by using the Encrypt() method of the RSACryptoServiceProvider class.
- ◆ The Encrypt() method accepts the two parameters byte array of the data to encrypt and a Boolean value that indicates whether or not to perform encryption.

© Aptech Ltd. Security/Session 11 44

In slide 44, refer to the code that shows how to initialize the RSACryptoServiceProvider object with the public key.

Then, explain them that once they have initialized the RSACryptoServiceProvider object with the public key, they can encrypt data by using the Encrypt() method of the RSACryptoServiceProvider class. The Encrypt() method accepts the two parameters: byte array of the data to encrypt and a Boolean value that indicates whether or not to perform encryption.

## Slide 45

Let us understand the return type of the encrypted text.

### Using Asymmetric Encryption 6-12

- ◆ The Encrypt() method after performing encryption returns a byte array of the encrypted text.
- ◆ Following code snippet shows returning a byte array of the encrypted text:

Snippet

```
byte[] plainbytes = new UnicodeEncoding().GetBytes("Plain text to encrypt.");
byte[] cipherbytes = rsaEncryptor.Encrypt(plainbytes, true);
```
- ◆ To decrypt data, you need to initialize an RSACryptoServiceProvider object using the private key of the key pair whose public key was used for encryption.

© Aptech Ltd. Security/Session 11 45

In slide 45, explain to the students that the Encrypt( ) method after performing encryption returns a byte array of the encrypted text.

Then, refer to the code that shows returning a byte array of the encrypted text.

Explain them that to decrypt data, they need to initialize an RSACryptoServiceProvider object using the private key of the key pair whose public key was used for encryption.

**Slide 46**

Let us understand how export the private key to an RSAParameters structure.

### Using Asymmetric Encryption 7-12

- ◆ Following code snippet shows how to initialize an RSACryptoServiceProvider object with the private key exported to an RSAParameters structure:

RSACryptoServiceProvider rSAKeyGenerator = new RSACryptoServiceProvider();  
RSAParameters rSAKeyInfo = rSAKeyGenerator.ExportParameters(true);  
RSACryptoServiceProvider rsaDecryptor = new RSACryptoServiceProvider();  
rsaDecryptor.ImportParameters(rSAKeyInfo);

- ◆ This code initializes an RSACryptoServiceProvider object with the private key exported to an RSAParameters structure.

Snippet

© Aptech Ltd.

Security/Session 11

46

In slide 46, refer to the code that shows how to initialize an RSACryptoServiceProvider object with the private key exported to an RSAParameters structure.

## Slide 47

Let us understand how to export key to XML format.

### Using Asymmetric Encryption 8-12

Following code snippet shows how to initialize an RSACryptoServiceProvider object with the private key exported to XML format:

**Snippet**

```
RSACryptoServiceProvider rSAKeyGenerator = new RSACryptoServiceProvider();
string keyPair = rSAKeyGenerator.ToXmlString(true);
RSACryptoServiceProvider rsaDecryptor = new RSACryptoServiceProvider();
rsaDecryptor.FromXmlString(keyPair);
```

- When the RSACryptoServiceProvider object is initialized with the private key, you can decrypt data by using the Decrypt() method of the RSACryptoServiceProvider class.
- The Decrypt() method accepts the two parameters, a byte array of the encrypted data and a Boolean value that indicates whether or not to perform encryption.

© Aptech Ltd. Security/Session 11 47

In slide 47, refer to the code that shows how to initialize an RSACryptoServiceProvider object with the private key exported to XML format.

Explain them that when the RSACryptoServiceProvider object is initialized with the private key, they can decrypt data by using the Decrypt() method of the RSACryptoServiceProvider class. The Decrypt() method accepts the two parameters, a byte array of the encrypted data and a Boolean value that indicates whether or not to perform encryption.

## Slides 48 to 51

Let us understand how to perform asymmetric encryption and decryption.

### Using Asymmetric Encryption 9-12

- Following code snippet shows a program that performs asymmetric encryption and decryption:

Snippet

```
using System;
using System.IO;
using System.Security.Cryptography;
using System.Text;
class AsymmetricEncryptionDemo {
    static byte[] EncryptData(string plainText, RSAParameters rsaParameters)
    {
        byte[] plainTextArray = new UnicodeEncoding().GetBytes(plainText);
        RSACryptoServiceProvider RSA = new RSACryptoServiceProvider();
        RSA.ImportParameters(rsaParameters);
        byte[] encryptedData = RSA.Encrypt(plainTextArray, true);
        return encryptedData;
    }
    static byte[] DecryptData(byte[] encryptedData, RSAParameters rsaParameters)
    {
        RSACryptoServiceProvider RSA = new RSACryptoServiceProvider();
```

© Aptech Ltd.

Security/Session 11

48

### Using Asymmetric Encryption 10-12

Snippet

```
RSA.ImportParameters(rsaParameters);
byte[] decryptedData = RSA.Decrypt(encryptedData, true);
return decryptedData;
}
static void Main(string[] args)
{
Console.WriteLine("Enter text to encrypt:");
String inputText = Console.ReadLine();
RSACryptoServiceProvider RSA = new RSACryptoServiceProvider();
RSAParameters RSAParam=RSA.ExportParameters(false);
byte[] encryptedData = EncryptData(inputText, RSAParam);
string encryptedString = Encoding.Default.GetString(encryptedData);
Console.WriteLine("\nEncrypted data \n{0}", encryptedString);
byte[] decryptedData = DecryptData(encryptedData,
RSA.ExportParameters(true));
String decryptedString = new
UnicodeEncoding().GetString(decryptedData);
Console.WriteLine("\nDecrypted data \n{0}", decryptedString);
}
```

© Aptech Ltd.

Security/Session 11

49

## Using Asymmetric Encryption 11-12

- ◆ In the code:
  - ❖ The Main() method creates a RSACryptoServiceProvider object and exports the public key as a RSAParameters structure.
  - ❖ The EncryptData() method is called passing the user entered plain text and the RSAParameters object. The EncryptData() method uses the exported public key to encrypt the data and returns the encrypted data as a byte array.
  - ❖ The Main() method then, exports both the public and private key of the RSACryptoServiceProvider object into a second RSAParameters object.
  - ❖ The DecryptData() method is called passing the encrypted byte array and the RSAParameters object.
  - ❖ Finally, the DecryptData() method performs the decryption and returns the original plain text as a string.

## Using Asymmetric Encryption 12-12

- ◆ Following figure shows the output of the code:

```

C:\Windows\system32\cmd.exe
Enter text to encrypt:
This is a simple text.

Encrypted data
[long string of encrypted bytes]

Decrypted data
This is a simple text.
Press any key to continue . . .

```

Using slides 48 to 51, explain how to perform asymmetric encryption and decryption.

In slides 48 and 49, refer to the code that shows a program that performs asymmetric encryption and decryption.

In slide 50, explain that in the code:

- The Main( ) method creates an RSACryptoServiceProvider object and exports the public key as an RSAParameters structure.
- The EncryptData( ) method is called passing the user entered plain text and the RSAParameters object. The EncryptData( ) method uses the exported public key to encrypt the data and returns the encrypted data as a byte array.
- The Main( ) method then exports both the public and private key of the RSACryptoServiceProvider object into a second RSAParameters object.
- The DecryptData( ) method is called passing the encrypted byte array and the RSAParameters object.
- Finally, the DecryptData( ) method performs the decryption and returns the original plain text as a string.

In slide 51, refer to the figure that shows the output of the code.

## Slides 52 and 53

Let us understand salting and hashing techniques.

### Salting and Hashing 1-2

- ◆ Hashing is a technique that allows generating a unique hash value of data by using a hashing algorithm.
- ◆ Once you have generated a unique hash value for any data, this data cannot be converted back to the original form.
- ◆ The process of hashing in order to protect passwords in a database can be better understood by the following steps:
  - ❖ First, the application generates a hash value for a user password and stores it to the database.
  - ❖ Next, the application receives a login request comprising of a user name and password.
  - ❖ Then, the application generates a new hash value for the received password.
  - ❖ Finally, the application compares the newly generated hash value with the existing value stored in the database.

### Salting and Hashing 2-2

- ◆ Salting is another security technique that you can apply on hashing.
- ◆ Salting allows creating and adding a random string to the input data before generating a hash value.
- ◆ The process of hashing in combination with salting to protect passwords stored in the database can be better understood by the following steps:
  - ❖ First, the application adds a random salt to the user password and generates a hash value and the application stores both the hash value and the salt to the database.
  - ❖ Then, the application receives a login request with a user name and password.
  - ❖ Next, the application retrieves the salt, applies it to the received password, and generates a new hash value.
  - ❖ Finally, the application compares both the new hash value and the existing one stored in the database.

Using slides 52 and 53, explain salting and hashing techniques.

Consider a scenario, where you are creating an ASP.NET MVC application that stores user credentials in the form of user name and password in a database. Though, you have implemented

security to prevent the data of the application from malicious attacks, the user credentials stored in the database might be compromised if an attacker gains direct access to the database. So, you need to ensure that the user credentials are secure in the database. For this, you need to use an additional security measures in the application. Explain that hashing is a technique that allows generating a unique hash value of data by using a hashing algorithm. Once they have generated a unique hash value for any data, this data cannot be converted back to the original form.

Tell them to use such additional security on the data stored in a database, you can use two techniques, known as hashing and salting.

Explain that the process of hashing in order to protect passwords in a database can be better understood by the following steps:

- First, the application generates a hash value for a user password and stores it to the database.
- Next, the application receives a login request comprising of a user name and password.
- Then, the application generates a new hash value for the received password.
- Finally, the application compares the newly generated hash value with the existing value stored in the database.

In slide 53, explain to the students that salting is another security technique that they can apply on hashing. Salting allows creating and adding a random string to the input data before generating a hash value.

Then, explain that the process of hashing in combination with salting to protect passwords stored in the database can be better understood by the following steps:

- First, the application adds a random salt to the user password and generates a hash value and stores both the hash value and the salt to the database.
- Then, the application receives a login request with a user name and password.
- Next, the application retrieves the salt, applies it to the received password, and generates a new hash value.
- Finally, the application compares both the new hash value and the existing one stored in the database.

## Slides 54 and 55

Let us understand digital signature.

## Digital Signature 1-7

- ◆ You can use digital signature to authenticate the identity of a sender of some kind of data.
- ◆ This type of signature also ensures that the data has not been tampered while in transit.
- ◆ By using digital signature, a user sending a signed data cannot later deny his or her ownership of the data.
- ◆ The process of using digital signature requires the following operations to be performed:
  - ◊ First, a sender computes a hash value from the data being sent.
  - ◊ Then, the sender encrypts the hash value with the private key of an asymmetric key pair.
  - ◊ Next, the sender sends the data and the digital signature to the receiver.
  - ◊ Next, the receiver computes a hash from the received data.
  - ◊ Finally, the receiver decrypts the signature using the public key of the sender and then, compares the hash values for authenticity.

## Digital Signature 2-7

- ◆ To understand how digital signatures work, you first need to generate a digital signature for some data.
- ◆ To generate the asymmetric keys whose private key can be used to sign data, you need to:
  - ◊ Use the RSACryptoServiceProvider class.
  - ◊ Then, you need to store the asymmetric keys using a secured mechanism.
  - ◊ Then, the CspParameters class can be used to create a key container and to add and remove keys to and from the container.
- ◆ To create a key container for an RSACryptoServiceProvider object:
  - ◊ You first need to use the default constructor of the CspParameters class to create a key container instance.
  - ◊ Then, you need to set the container name using the KeyContainerName property of the CspParameters class.

Using slides 54 and 55, explain digital signatures.

In slide 54, explain that they can use digital signature to authenticate the identity of a sender of some kind of data. This type of signature also ensures that the data has not been tampered while in transit. By using digital signature, a user sending a signed data cannot later deny his/her ownership of the data.

Then, explain them that the process of using digital signature requires the following operations to be performed:

- First, a sender computes a hash value from the data being sent.
- Then, the sender encrypts the hash value with the private key of an asymmetric key pair.
- Next, the sender sends the data and the digital signature to the receiver.
- Next, the receiver computes a hash from the received data.
- Finally, the receiver decrypts the signature using the public key of the sender and then, compares the hash values for authenticity.

In slide 55, explain that to understand how digital signatures work, they first need to generate a digital signature for some data.

Then, explain them the steps to generate the asymmetric keys whose private key can be used to sign data, as displayed on the slide.

Then, explain them the steps to create a key container for an RSACryptoServiceProvider object, as displayed on the slide.

## Slide 56

Let us understand how to use KeyContainerName property.

### Digital Signature 3-7

- ◆ Following code snippet shows using the KeyContainerName property of the CspParameters class:

CspParameters param = new CspParameters();  
param.KeyContainerName = "SignatureContainer121";

- ◆ Now, you need to store the key pair of the RSACryptoServiceProvider object in the key container. For this:
  - ❖ You need to pass the CspParameters object to the constructor while creating the RSACryptoServiceProvider object.
  - ❖ Then, you need to set the PersistKeyInCsp property of the RSACryptoServiceProvider object to true.

© Aptech Ltd. Security/Session 11 56

Using slide 56, refer to the code that shows use of the KeyContainerName property of the CspParameters class.

Then, explain them that now, they need to store the key pair of the RSACryptoServiceProvider object in the key container. For this, they need to pass the CspParameters object to the constructor while creating the RSACryptoServiceProvider object. Then, they need to set the PersistKeyInCsp property of the RSACryptoServiceProvider object to true.

## Slide 57

Let us understand the `PersistKeyInCsp` property.

### Digital Signature 4-7

- Following code snippet shows how to set the `PersistKeyInCsp` property:

```
using (RSACryptoServiceProvider rsa = new
    RSACryptoServiceProvider(param))
{
    ...
    rsa.PersistKeyInCsp = true;
    ...
}
```
- In this code, an `RSACryptoServiceProvider` object is initialized with a `CspParameters` object and then, the key pair is stored in the `CspParameters` object.
- Then, you can generate the signature using the `SignData()` method.
- This method accepts the data to sign in a byte array and the hash algorithm is used to create the hash value.
- You can pass the `SHA256` string value to use the `SHA256` hash algorithm to the `SignData()` method that returns the signature in a byte array.

© Aptech Ltd. Snippet Security/Session 11 57

Using slide 57, refer to the code that shows how to set the `PersistKeyInCsp` property.

Explain them that in this code, an `RSACryptoServiceProvider` object is initialized with a `CspParameters` object and then, the key pair is stored in the `CspParameters` object.

Then, explain them that they can generate the signature using the `SignData()` method. This method accepts the data to sign in a byte array and the hash algorithm is used to create the hash value.

Mention them that they can pass the `SHA256` string value to use the `SHA256` hash algorithm to the `SignData()` method that returns the signature in a byte array.

## Slide 58

Let us understand how to use the `SignData()` method.

### Digital Signature 5-7

- ◆ Following code snippet shows how to use the `SignData()` method:

**Snippet**

```
byte[] byteData = Encoding.Unicode.GetBytes("Data to Sign.");
byte[] byteSignature = rsa.SignData(byteData, "SHA256");
```

- ◆ In this code, the `SignData()` generates a digital signature and returns the signature in a byte array.
- ◆ Once you have created the digital signature, you can verify it.
- ◆ For that, you need to first create a `CspParameters` object and use the same key container name that you used while creating the signature.
- ◆ This will ensure that when you later create the `RSACryptoServiceProvider` object, it will be initialized with the same key pair that was used to generate the signature.

© Aptech Ltd. Security/Session 11 58

Use slide 58, refer to the code that shows how to use the `SignData()` method.

Explain them that in this code, the `SignData()` generates a digital signature and returns the signature in a byte array.

Then, explain them that once they have created the digital signature, they can verify it. For that, they need to first create a `CspParameters` object and use the same key container name that they used while creating the signature. This will ensure that when they later create the `RSACryptoServiceProvider` object, it will be initialized with the same key pair that was used to generate the signature.

## Slides 59 and 60

Let us understand `CspParameters` object.

### Digital Signature 6-7

- Following code snippet shows creating a `CspParameters` object:

**Snippet**

```
CspParameters param = new CspParameters();
param.KeyContainerName = "SignatureContainer101";
```

- After creating a `CspParameters` object, you can create the `RSACryptoServiceProvider` object initialized with the `CspParameters` object.
- You can then, call the `VerifyData()` method that accepts three different types of parameters, such as the **byte array of the data** that needs to be verified against the signature, the **hash algorithm** that is used to generate the signature, and the **byte array of the generated signature**.

### Digital Signature 7-7

- The `VerifyData()` method returns a Boolean true value if the signature is successfully verified, or false.
- Following code snippet shows using the `VerifyData()` method that returns a Boolean true value:

**Snippet**

```
bool isSuccess = false;
using (RSACryptoServiceProvider rsa = new
RSACryptoServiceProvider(param))
{
    isSuccess = rsa.VerifyData(byteData, "SHA256", byteSignature);
}
```

This code creates an `RSACryptoServiceProvider` object and then, calls the `VerifyData()` method to verify the signature.

Using slides 59 and 60, refer to the code that shows how to create a `CspParameters` object.

Explain them that after creating a `CspParameters` object, they can create the `RSACryptoServiceProvider` object initialized with the `CspParameters` object. They can then, call the `VerifyData()` method that accepts three different types of parameters, such as the byte array of the data that needs to be verified against the signature, the hash algorithm that is used to generate the signature, and the byte array of the generated signature.

In slide 60, explain that the `VerifyData()` method returns a Boolean true value if the signature is successfully verified, or false.

Then, refer to the code that shows use of the `VerifyData()` method that returns a Boolean true value.

Explain them that this code creates an `RSACryptoServiceProvider` object and then, calls the `VerifyData()` method to verify the signature.

## Slide 61

Let us summarize the session.

Summary
<ul style="list-style-type: none"><li>◆ HTML encoding is a process that converts potentially unsafe characters to their HTML-encoded equivalent.</li><li>◆ The ASP.NET MVC Framework provides a request validation feature that examines an HTTP request to check and prevent potentially malicious content.</li><li>◆ The user-generated token approach enables storing a user-generated token using a HTML hidden field in the user session.</li><li>◆ SQL injection is a form of attack in which a user posts SQL code to an application, thus it creates an SQL statement that will execute with malicious intent.</li><li>◆ Encryption is a technique that ensures data confidentiality by converting data in plain text to cipher (secretly coded) text.</li><li>◆ Hashing is a technique that allows generating a unique hash value of data by using a hashing algorithm.</li><li>◆ Salting is a security technique that you can apply on hashing to create and add a random string to the input data before generating a hash value.</li></ul>

In slide 61, summarize the session. End the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- HTML encoding is a process that converts potentially unsafe characters to their HTML-encoded equivalent.
- The ASP.NET MVC Framework provides a request validation feature that examines an HTTP request to check and prevent potentially malicious content.
- The user-generated token approach enables storing a user-generated token using a HTML hidden field in the user session.
- SQL injection is a form of attack in which a user posts SQL code to an application, thus, it creates an SQL statement that will execute with malicious intent.
- Encryption is a technique that ensures data confidentiality by converting data in plain text to cipher (secretly coded) text.
- Hashing is a technique that allows generating a unique hash value of data by using a hashing algorithm.
- Salting is a security technique that you can apply on hashing to create and add a random string to the input data before generating a hash value.

### **11.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

#### **Tips:**

You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. What are the private keys?
2. How can you make symmetric encryption more secure?
3. What is RSA algorithm?

## Session 12

### Globalization

#### **12.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **12.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe how to implement internationalization
- Define and describe how to implement localization

#### **12.1.2 Teaching Skills**

To teach this session successfully, you must know about the layout. You should know about implementing internationalization in an ASP.NET MVC application. You should also be aware of the process of implementing globalization in applications.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities:**

Follow the order given during In-Class activities:

#### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students Slide 2 of the presentation. Tell them that they will be introduced to internationalization in ASP.NET MVC applications. They will learn about how to implement internationalization in ASP.NET MVC applications. This session will also discuss about how to implement globalization.

## **12.2 Introduction**

### **Slide 3**

Let us understand internationalization.

### **Implementing Internationalization**

- ◆ At recent times, most of the organizations do business in the global marketplace.
- ◆ So, these types of organizations must design applications to accommodate users from a wide variety of cultures.
- ◆ Therefore, you need to enable your Web applications to represent information in the users' native language and formats.
- ◆ This functionality can be provided by implementing internationalization.
- ◆ The ASP.NET MVC Framework allows you to develop multicultural applications.
- ◆ You can develop an ASP.NET MVC application to retrieve information about a particular culture and region to which a user belongs.
- ◆ Based on this information, you can format and present your application data in the desired language.

Use slide 3 to explain the students that at recent times, most of the organizations do business in the global marketplace. So, these types of organizations design applications to accommodate users from a wide variety of cultures. Therefore, they need to enable Web applications to represent information in the users' native language and formats.

Tell them that this functionality can be provided by implementing internationalization. The ASP.NET MVC Framework allows developing multicultural applications. They can develop an ASP.NET MVC application to retrieve information about a particular culture and region to which a user belongs. Based on this information, they can format and present the application data in the desired language.

### **Additional Information**

Internationalization is the process of making the application support a range of languages and locales. Internationalization is often abbreviated to "I18N". The abbreviation takes the first and last letters and the number of letters between them, so 18 stands for the number of letters between the first "I" and the last "N".

## Slides 4 and 5

Let us understand about culture code in achieving globalization.

### Culture Code 1-2

- ◆ To implement internationalization in an application, you need to understand about the culture code that the .NET Framework uses.
- ◆ The .NET Framework uses the following syntax to represent a culture code:
 

```
<languagecode>-<country/regioncode>
```

where,

  - ◆ **<languagecode>**: Is a lowercase two-letter code for a language.
  - ◆ **<country/regioncode>**: Is an uppercase two-letter code for a country or a region.
- ◆ In an ASP.NET MVC application, you can access all the supported culture of the .NET Framework using the CultureInfo class of the System.Globalization namespace.

### Culture Code 2-2

- ◆ Following code snippet shows using the CultureInfo class:

Snippet

```
ArrayList cultureNameList = new ArrayList();
CultureInfo[] cInfo =
CultureInfo.GetCultures(CultureTypes.AllCultures);
foreach (CultureInfo cultures in cInfo)
{
    cultureNameList.Add(String.Format("Culture Name {0} :
Display Name {1}", cultures.Name, cultures.DisplayName));
}
```

- ◆ This code:
  - ◆ First, calls the CultureInfo.GetCultures() method passing a CultureTypes.AllCultures enumeration member.
  - ◆ Next, it accesses the CultureInfo.Name and CultureInfo.DisplayName properties of all the supported cultures and adds the information as a formatted string to an ArrayList.

Using slides 4 and 5, explain the students that to implement internationalization in an application, they need to understand about the culture code that the .NET Framework uses.

Tell them culture represents the language and the region. Region is optional. Then, explain them the syntax to represent a culture code as displayed on the slide. Explain them that <languagecode> is a lowercase two-letter code for a language.

<country/regioncode> is an uppercase two-letter code for a country or a region. For example, the culture name en-US represents the English language of the US. However, the culture name en-GB represents the English language of the UK.

Mention that in an ASP.NET MVC application, they can access all the supported culture of the .NET framework using the CultureInfo class. This class is present in the System.Globalization namespace. It provides information about a specific culture also called as a locale.

Using slide 5, refer to the code that shows using the CultureInfo class.

Then explain them that this code first, calls the CultureInfo.GetCultures() method passing a CultureTypes.AllCultures enumeration member. Next, it accesses the CultureInfo.Name and CultureInfo.DisplayName properties of all the supported cultures and adds the information as a formatted string to an ArrayList.

Following is the additional code for displaying the culture:

```
public static void Main()
{
    // Displays several properties of the neutral cultures.
    Console.WriteLine("CULTURE ISO ISO WIN DISPLAYNAME
ENGLISHNAME");
    foreach (CultureInfo ci in
CultureInfo.GetCultures(CultureTypes.NeutralCultures))
    {
        Console.Write("{0,-7}", ci.Name);
        Console.Write(" {0,-3}", ci.TwoLetterISOLanguageName);
        Console.Write(" {0,-3}", ci.ThreeLetterISOLanguageName);
        Console.Write(" {0,-3}", ci.ThreeLetterWindowsLanguageName);
        Console.Write(" {0,-40}", ci.DisplayName);
        Console.WriteLine(" {0,-40}", ci.EnglishName);
    }
}
```

A neutral culture is specified by only the two-letter lowercase language code. For example, fr specifies the neutral culture for French, and de specifies the neutral culture for German.

### **Additional Information**

To know the language code and region code for different countries, you can refer to the link:  
[http://msdn.microsoft.com/en-us/library/ee825488\(v=cs.20\).aspx](http://msdn.microsoft.com/en-us/library/ee825488(v=cs.20).aspx)

## Slides 6 to 9

Let us understand setting cultures.

### Setting Cultures 1-4

- ◆ Before you configure internationalization in an ASP.NET MVC application, you need to understand about the following properties of the Page class:
  - ❖ The Culture property is a string that determines the results of culture-dependent functions, such as formatting of date and time and currency.
  - ❖ The UICulture property is a string that determines the resource file that contains locale specific content for a page.
- ◆ While configuring internationalization in an application, you might often need to consider assigning different values for these properties.
- ◆ In an ASP.NET MVC, you can set and access the current culture and UI culture of a page from the request processing thread by using the following properties:
  - ❖ Thread.CurrentCulture
  - ❖ Thread.CurrentThread.UICulture

© Aptech Ltd. Globalization/Session 12 6

### Setting Cultures 2-4

- ◆ The .NET Framework examines both these properties before rendering culture-dependent functions.
- ◆ Following code snippet shows explicitly setting the CurrentCulture property of the thread executing the Index() action method of the HomeController:

```
public class HomeController : Controller {
    public ActionResult Index() {
        Thread.CurrentThread.CurrentCulture = new
System.Globalization.CultureInfo("fr-FR");
        ViewBag.CultureName =
System.Globalization.CultureInfo.CurrentCulture.Name;
        ViewBag.Message = DateTime.Now.ToString("yyyy-MM-dd");
        return View();
    }
}
```

Snippet

- ◆ This code:
  - ❖ Sets the CurrentCulture property to a CultureInfo object initialized with the fr-FR culture name.
  - ❖ Then, adds the name of the current culture and the current date to a ViewBag object before it returns the view.

© Aptech Ltd. Globalization/Session 12 7

## Setting Cultures 3-4

- ◆ In the corresponding Index.cshtml view, you can access and display the ViewBag messages.
- ◆ Following code snippet shows how to access and display the ViewBag messages:

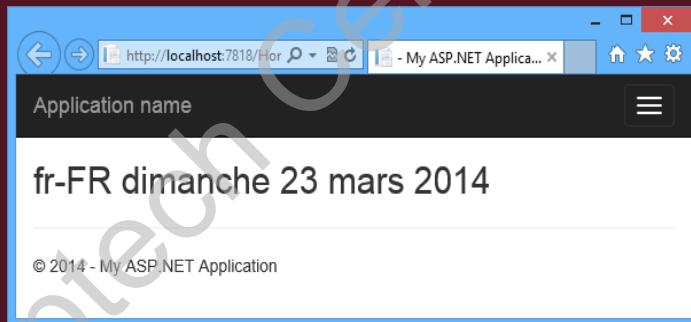
**Snippet**

```
<h2>@ViewBag.CultureName @ViewBag.Message</h2>
```

- ◆ When the Index.cshtml view is accessed in a browser, it displays the fr-FR as the culture name and the date in French.

## Setting Cultures 4-4

- ◆ Following figure shows the Index.cshtml view in French:



Using slides 6 to 9, explain the students that before they configure internationalization in an ASP.NET MVC application, they need to understand about the following properties of the Page class:

- The Culture property is a string that determines the results of culture-dependent functions, such as formatting of date and time and currency.

- The `UICulture` property is a string that determines the resource file containing locale specific content for a page.

Tell them that while configuring internationalization in an application, they might often need to consider assigning different values for these properties.

Explain them that in an ASP.NET MVC, they can set and access the current culture and UI culture of a page from the request processing thread by using properties, such as `Thread.CurrentThread.CurrentCulture` and `Thread.CurrentThread.CurrentUICulture`.

**Tips:**

Although both the `Culture` and `UICulture` properties accept String values, you can only assign predefined culture names and not just any string to them.

Use slide 7 to explain them that the .NET Framework examines both these properties before rendering culture-dependent functions.

Then, refer to the code that shows explicitly setting the `CurrentCulture` property of the thread executing the `Index()` action method of the `HomeController`.

Explain them that code sets the `CurrentCulture` property to a `CultureInfo` object initialized with the `fr-FR` culture name. Then, adds the name of the current culture and the current date to a `ViewBag` object before it returns the view.

Use slide 8 to explain them that in the corresponding `Index.cshtml` view, they can access and display the `ViewBag` messages. Then, refer to the code that shows how to access and display the `ViewBag` messages.

Explain them that when the `Index.cshtml` view is accessed in a browser, it displays the `fr-FR` as the culture name and the date in French. Using slide 9, refer to the figure that shows the `Index.cshtml` view in French.

## Slides 10 to 15

Let us understand using the Web.config file.

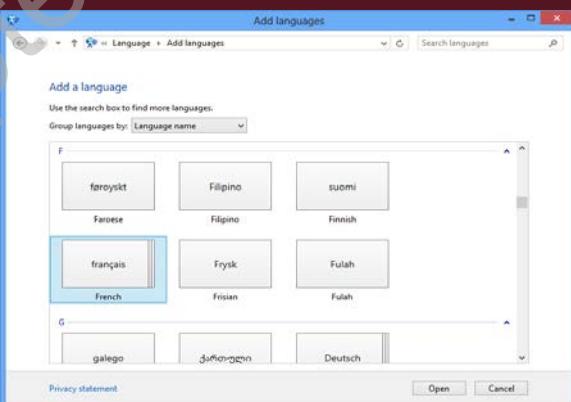
### Using the Web.config File 1-6

- ◆ In an ASP.NET MVC application, you can use the Web.config file to use the language that is set as the default language in the browser that sends requests to the application.
- ◆ For that, you need to know how to set the preferred language of the browser.
- ◆ Then, you need to know how to configure the Web.config file to use the preferred language.
- ◆ To set the preferred language in a browser, you need to perform the following steps:
  - ❖ Open **Internet Explorer (IE)**.
  - ❖ Press the **Alt+X** keys to open the Tools menu.
  - ❖ Click **Internet Options** in the Tools menu. The **Internet Options** dialog box is displayed.
  - ❖ Click **Languages** in the Internet Options dialog box. The **Language Preference** dialog box is displayed.

© Aptech Ltd. Globalization/Session 12 10

### Using the Web.config File 2-6

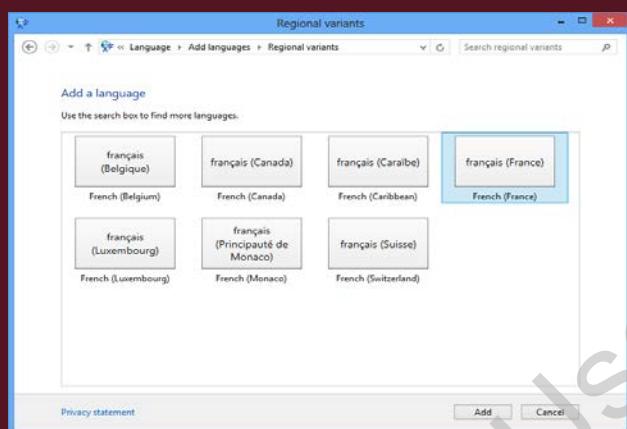
- ❖ Click **Set Language Preferences** in the Language Preference dialog box. The **Add languages** window is displayed.
- ❖ Locate and select French in the Add Languages window.
- ◆ Following figure shows selecting French in the Add languages window:



© Aptech Ltd. Globalization/Session 12 11

## Using the Web.config File 3-6

- ❖ Click the **Open** button. The **Regional variants** dialog box is displayed.
- ❖ Select French (France) in the Regional variants dialog box.
- ◆ Following figure shows selecting French (France) in the Regional variants window:



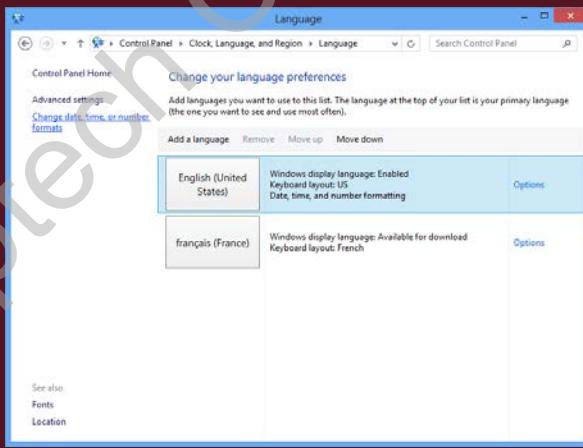
© Aptech Ltd.

Globalization/Session 12

12

## Using the Web.config File 4-6

- ❖ Click the **Add** button. The **Language** window displays the newly added language.
- ◆ Following figure shows the newly added français (France) language:



© Aptech Ltd.

Globalization/Session 12

13

## Using the Web.config File 5-6

- ◆ Select **francais (France)** and click the Move up button to make French (France) as the first preferred language of the browser.
- ◆ The Language window displays francais (France) at the top of the language list.
- ◆ Close the Language window.
- ◆ Close the Internet Options dialog box.
- ◆ Click **OK** in the Language Preferences dialog box.
- ◆ Click **OK** in the Internet Options dialog box
- ◆ Once you set the preferred language of the browser, you can configure the Web.config file to use the browser preferred language by adding the `<globalization>` element under the `<system.web>` element.

## Using the Web.config File 6-6

- ◆ Following code snippet shows using the `<globalization>` element:

**Snippet**

```
<system.web>
    <globalization culture="auto" uiCulture="auto"
        enableClientBasedCulture="true"/>
    ...
    ...
<system.web>
```

- ◆ In this code, the true value of the `enableClientBasedCulture` attribute instructs ASP.NET to set the `UICulture` and `Culture` properties for a Web page, based on the preferred languages set in the browser sending the request.

Using slides 10 to 15, explain the students that in an ASP.NET MVC application, they can use the `Web.config` file to use the language that is set as the default language in the browser that sends requests to the application. For that, they need to know how to set the preferred language of the browser. Then, they need to know how to configure the `Web.config` file to use the preferred language.

Next, explain them the steps to set the preferred language in a browser, as displayed on the slide.

Using slide 11, refer to the figure that shows selecting French in the Add languages window. Using slide 12 and refer to the figure that shows selecting French (France) in the Regional variants window.

**Additional Information:**

Apart from determining the preferred language of the user in `Web.config` file, you can determine the same by using the request header fields. When a browser sends a request for a Web page, it sends several HTTP header fields along with the request. These header fields provide information about the request such as media types, encoding, and languages that the requested in response.

Use slide 13, refer to the figure that shows the newly added French (France) language.

Use slide 14 to continue explaining the steps to set preferred language of the browser.

Then, explain that once they set the preferred language of the browser, they can configure the `Web.config` file to use the browser preferred language by adding the `<globalization>` element under the `<system.web>` element.

In slide 15, refer to the code that shows using the `<globalization>` element. Then, explain them that in this code the true value of the `enableClientBasedCulture` attribute instructs ASP.NET to set the `UICulture` and `Culture` properties for a Web page, based on the preferred languages set in the browser sending the request.

**Slide 16**

Let us understand implementing localization.

## Implementing Localization

- ◆ You can implement localization in your application to display content in languages that a user prefers.
- ◆ For that, you need to separate the language-related and the culture-related content from the application code.
- ◆ The two approaches to implement localization are as follows:
  - ❖ Using Resource files
  - ❖ Using Separate views

© Aptech Ltd. Globalization/Session 12 16

In slide 16, explain the students that they can implement localization in an application to display content in languages that a user prefers. For that, they need to separate the language-related and the culture-related content from the application code.

Then tell them about the following two approaches to implement localization:

- Using Resource files
- Using Separate views

## Slide 17

Let us understand resource files.

### Resource Files 1-7

- ◆ When you use resource files, you must first create a base resource file for the application, for example, MyResources.resx.
- ◆ Next, you must create separate resource files for each culture that your application supports.
- ◆ These resource files will have the name, as shown in the following syntax:  
`<base_resource-filename>.<language_code>-<country_code>.resx`  
where,
  - ◆ **base\_resource-filename**: Is the name of the base resource file, such as MyResources.
  - ◆ **language\_code**: Is the language code of the resource file, such as 'fr' for the French language.
  - ◆ **country\_code**: Is the country code of the resource file, such as 'FR' for France.

© Aptech Ltd. Globalization/Session 12 17

Using slide 17, to explain the students that when they use resource files, they must first create a base resource file for the application, for example, MyResources.resx. Next, they must create separate resource files for each culture that the application supports.

Then, explain them the syntax of naming a resource file, as displayed on the slide.

## Slide 18

Let us understand the steps to create resource file.

### Resource Files 2-7

- ◆ In Visual Studio 2013, you can create resource files by performing the following tasks:
  - ❖ Create a **Resources** folder in your application.
  - ❖ Right-click the **Resources** folder and select **Add→New Item**.
  - ❖ Select **General** from the Installed templates and select **Resources File**.
  - ❖ Replace the name given in the Name text box with **MyResources.resx**.
  - ❖ Click the **Add** button. The Resource Editor displays the MyResources.resx file.
  - ❖ Select **Public** from the Access Modifier drop-down list.
  - ❖ Select **MyResources.resx** file in the Solution Explorer window. The Properties window displays the properties of the MyResources.resx file.
  - ❖ Type **Resources** for the Custom Tool Namespace property.

© Aptech Ltd. Globalization/Session 12 18

Use slide 18 to explain the steps to create resource files in Visual Studio 2013, as displayed on the slide.

#### Additional Information:

When you set **Public** as the access modifier, Visual Studio uses a localization tool named **PublicResXFileCodeGenerator**, which is responsible for creating a public class corresponding to a .resx file. This class contains public properties for each key-value pair of the resource file.

When you specify a namespace for the **Custom Tool Namespace** property of the resource file, further, you can import this namespace in your views and controllers to access that resource file.

## Slide 19

Let us understand how to set Namespace property.

### Resource Files 3-7

- Following figure shows specifying values for the Custom Tool Namespace property:

The screenshot shows the 'Properties' window for a file named 'MyResources.resx'. The 'File Properties' tab is selected. In the 'Custom Tool Namespace' section, the value 'Resources' is listed under 'Custom Tool Namespace'. A tooltip below the entry provides the definition: 'The namespace into which the output of the custom tool is placed.' Other properties shown include Build Action (Embedded Resource), Copy to Output Direct (Do not copy), and File Name (MyResources.resx).

© Aptech Ltd. Globalization/Session 12 19

Use slide 19, refer to the figure that shows specifying values for the Custom Tool **Namespace** property.

## Slides 20 and 21

Let us understand how to assign name and value pair.

### Resource Files 4-7

- ◆ Add the Name-Value pairs in the Resource Editor.
- ◆ Following figure shows specifying Name-Value pairs in the Resource Editor:



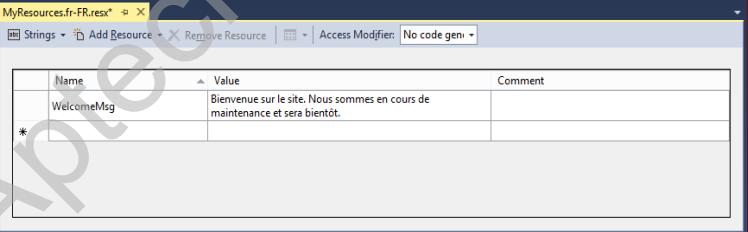
Name	Value	Comment
WelcomeMsg	Welcome to the website. We are under maintenance and will be up soon.	
String1		

- ◆ Repeat steps to create a MyResources.fr-FR.resx file in the Resources folder. Then, add the Name-Value pairs of the MyResources.fr-FR.resx file in the Name and Value columns of the Resource Editor.

© Aptech Ltd. Globalization/Session 12 20

### Resource Files 5-7

- ◆ Following figure shows specifying the Name-Value pairs of the MyResources.fr-FR.resx file:



Name	Value	Comment
WelcomeMsg	Bienvenue sur le site. Nous sommes en cours de maintenance et sera bientôt.	

- ◆ In the Index.cshtml page of the HomeController, you can access the value having the WelcomeMsg key that you have specified in the resource files.

© Aptech Ltd. Globalization/Session 12 21

Use slides 20 and 21, explain the steps to create resource files. Then, refer to the figure that shows specifying Name-Value pairs in the **Resource Editor**.

Using slide 21, refer to the figure that shows specifying the Name-Value pairs of the **MyResources.fr-FR.resx** file.

Tell them that in the `Index.cshtml` page of the `HomeController`, they can access the value having the `WelcomeMsg` key that you have specified in the resource files.

For Aptech Centre Use Only

## Slides 22 and 23

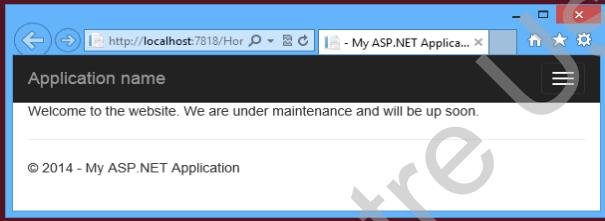
Let us understand how to access key and value from a resource.

### Resource Files 6-7

- Following code snippet shows how to access the value having the WelcomeMsg key:

```
<p>@Resources.MyResources.WelcomeMsg</p>
```

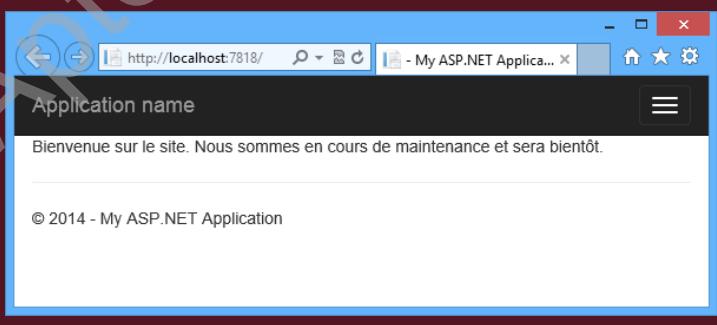
- A browser whose preferred language is not set to fr-FR will render the view using content from the default resource file, named MyResources.resx file.
- Following figure shows the view using the content from the default resource file:



© Aptech Ltd. Globalization/Session 12 22

### Resource Files 7-7

- A browser whose preferred language is set to fr-FR will render the view using content from the resource file, named MyResources.fr-FR.resx file.
- Following figure shows the view using the content from MyResources.fr-FR.resx file:



© Aptech Ltd. Globalization/Session 12 23

Using slides 22 and 23, explain the code that shows how to access the value having the WelcomeMsg key.

Explain them that a browser whose preferred language is not set to fr-FR will render the view using content from the default resource file, named **MyResources.resx** file.

Then, refer to the figure that shows the view using the content from the default resource file.

In slide 23, explain to the students that a browser whose preferred language is set to fr-FR will render the view using content from the resource file, named **MyResources.fr-FR.resx** file.

Then, refer to the figure that shows the view using the content from **MyResources.fr-FR.resx** file.

For Aptech Centre Use Only

## Slide 24

Let us understand using separate views.

### Using Separate Views 1-2

- ◆ In an application, you can use separate views each for the supported culture and hard code the localized text in them.
- ◆ This will not only result in clean and readable views, but also enable you to control how to position localized text elements in the view.
- ◆ You can use the approach of using separate views by creating different culture-based views for a particular view that needs to be localized.
- ◆ After creating the views, you can make conditional checks in the controller action to obtain the browser preferred language and accordingly return the corresponding view.

© Aptech Ltd. Globalization/Session 12 24

Using slide 24, explain that in an application, they can use separate views each for the supported culture and hard code the localized text in them. This will not only result in clean and readable views, but also enable them to control how to position localized text elements in the view.

Tell them that they can use the approach of using separate views by creating different culture-based views for a particular view that needs to be localized.

Mention that after creating the views, they can make conditional checks in the controller action to obtain the browser preferred language and accordingly return the corresponding view.

## Slide 25

Let us understand how to check the language set in the browser.

### Using Separate Views 2-2

- Following code snippet shows the Index() action method of a HomeController, that returns different views based on the preferred language of the browser:

Snippet

```
public ActionResult Index()
{
    if(HttpContext.Request.UserLanguages[0] == "fr-FR")
        return View("Index.fr-FR");
    else if (HttpContext.Request.UserLanguages[0] == "en-US")
        return View("Index.en-US");
    else
        return View();
}
```

- This code uses an if-else statement to check the browser preferred language and accordingly returns a corresponding view. If the application does not have a view for a specific language, the action returns the default view.

© Aptech Ltd. Globalization/Session 12 25

Using slide 25, refer to the code that shows the Index( ) action method of a HomeController, that returns different views based on the preferred language of the browser.

Then, explain them that this code uses an if-else statement to check the browser preferred language and accordingly returns a corresponding view. If the application does not have a view for a specific language, the action returns the default view.

## Slide 26

Let us summarize the session.

### Summary

- ◆ Globalization is a process of designing and developing applications that can be used for multiple cultures.
- ◆ In an ASP.NET MVC application, you can access all the supported culture of the .NET Framework using the CultureInfo class.
- ◆ In an ASP.NET MVC application, you can set and access the current culture and UI culture of a page from the request processing thread.
- ◆ In an ASP.NET MVC application, you can use the Web.config file to use the language that is set as the default language in the browser that sends requests to the application.
- ◆ You can implement localization in your application to display content in languages that a user prefers.
- ◆ You can use two approaches to implement localization in your application, such as using resource files and using separate views.
- ◆ You can use separate views by creating different culture-based views for a particular view that needs to be localized.

© Aptech Ltd. Globalization/Session 12 26

In slide 26, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- Globalization is a process of designing and developing applications that can be used for multiple cultures.
- In an ASP.NET MVC application, you can access all the supported culture of the .NET Framework using the CultureInfo class.
- In an ASP.NET MVC application, you can set and access the current culture and UI culture of a page from the request processing thread.
- In an ASP.NET MVC application, you can use the Web.config file to use the language that is set as the default language in the browser that sends requests to the application.
- You can implement localization in your application to display content in languages that a user prefers.
- You can use two approaches to implement localization in your application, such as using resource files and using separate views.
- You can use separate views by creating different culture-based views for a particular view that needs to be localized.

### **12.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

#### **Tips:**

You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. How will you set German as the browser preferred language?
2. What is request header field?
3. What is the purpose of the PublicResXFileCodeGenerator localization tool?
4. What is the purpose of the Custom Tool Namespace property?

## Session 13 -

# Debugging and Monitoring

### **13.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **13.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe how to perform debugging
- Explain how to perform health monitoring of an application
- Define and describe how to use the Performance Monitoring Tool

#### **13.1.2 Teaching Skills**

To teach this session successfully, you must know about the layout. You should know about debugging ASP.NET MVC applications. You should also know about performing health monitoring of an application. You should also be aware of the process of using the Performance Monitoring tool.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

## Overview of the Session

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to debugging in ASP.NET MVC applications. They will learn about how to perform health monitoring of an application. This session will also discuss about how to use the Performance Monitoring tool.

## 13.2 In-Class Explanation

### Slide 3

Understand Debugging.

**Debugging**

- ◆ While developing an application it may contain syntax errors, logical errors, and run-time errors.
- ◆ The syntax errors are resolved when you compile the application in Visual Studio 2013.
- ◆ However other errors, such as logical and run-time errors cannot be identified while compiling the application.
- ◆ To identify such errors you need to debug the code while it is running.
- ◆ Debugging is a technique that enables you to resolve such errors present in the application.
- ◆ To perform debugging in your application, you can use various techniques that Visual Studio 2013 provides.

© Aptech Ltd.      Debugging and Monitoring/Session 13      3

Use slide 3 to explain the students that while developing an application it may contain syntax errors, logical errors, and run-time errors. The syntax errors are resolved when they compile the application in Visual Studio 2013.

Tell them that however other errors, such as logical and run-time errors cannot be identified while compiling the application. To identify such errors they need to debug the code while it is running. Debugging is a technique that enables the students to resolve such errors present in the application.

Finally, mention them that to perform debugging in an application, they can use various techniques that Visual Studio 2013 provides.

**Slide 4**

Understand Visual Studio Debugger.

**Visual Studio Debugger**

- ◆ There are various tools available that you can use to debug your application.
- ◆ One such tool is Visual Studio debugger.
- ◆ The Visual Studio debugger:
  - ◆ Enables you to run your code line by line, so that you can monitor the program execution properly.
  - ◆ Allows you to check the state of the application objects such as variables and database table values to ensure that the application is running properly.
  - ◆ Includes various features that enable you to debug your application.

© Aptech Ltd.      Debugging and Monitoring/Session 13      4

Display slide 4 and explain the students that there are various tools available that they can use to debug an application. One such tool is Visual Studio debugger.

Tell them that the Visual Studio debugger allows them to:

- Run code line by line, so that they can monitor the program execution properly.
- Check the state of the application objects such as variables and database table values to ensure that the application is running properly.
- To debug an application.

## Slides 5 to 8

Understand breakpoint.

### Breakpoints 1-4

- ◆ Breakpoints are places that you can specify in the code where the debugger stops the execution of the application.
- ◆ This allows you to view the current state of data in your application.
- ◆ To set up a breakpoint in Visual Studio 2013, you need to perform the following steps:
  - ◆ Open a project for example the MVCDemo in Visual Studio 2013.
  - ◆ Click the AccountController controller class in the Solution Explorer window. The Code Editor of Visual Studio 2013 displays the code of the controller class.
  - ◆ Locate the Register() action method of the controller class.
  - ◆ Right-click the beginning of the if-else statement of the Register() action method and select Breakpoint→Insert Breakpoint from the context menu that appears. A red circle on the left side of the if-else statement of the Register() action method is displayed.

© Aptech Ltd.      Debugging and Monitoring/Session 13      5

In slide 5, explain that Breakpoints are places that they can specify in the code where the debugger stops the execution of the application. This allows them to view the current state of data in an application.

Then explain them the process of setting up a breakpoint in Visual Studio 2013, as displayed on slide 5.

## Breakpoints 2-4

- ◆ Following figure shows adding a breakpoint:

The screenshot shows the Microsoft Visual Studio IDE interface. The main window displays the code for `AccountController.cs` under the `MVCDemo` project. A red dot breakpoint marker is placed on the line of code where the `if (ModelState.IsValid)` condition begins. The code implements a POST handler for the `/Account/Register` action, which includes validation logic and user creation. The Solution Explorer on the right shows the project structure, including the `Controllers` folder containing the `AccountController.cs` file.

```
// POST: /Account/Register
[HttpPost]
[AllowAnonymous]
[ValidateAntiForgeryToken]
public async Task<ActionResult> Register(RegisterViewModel model)
{
    if (ModelState.IsValid)
    {
        var user = new ApplicationUser() { UserName = model.UserName };
        var result = await UserManager.CreateAsync(user, model.Password);
        if (result.Succeeded)
        {
            await SignInAsync(user, isPersistent: false);
            return RedirectToAction("Index", "Home");
        }
        else
        {
            AddErrors(result);
        }
    }
}
```

- ◆ In this figure, the red block over the code line is where the debugger will open.

© Aptech Ltd.

Debugging and Monitoring/Session 13

6

Display slide 6 and you refer to the figure that shows adding a breakpoint.

Explain that the red block over the code line is where the debugger will open.

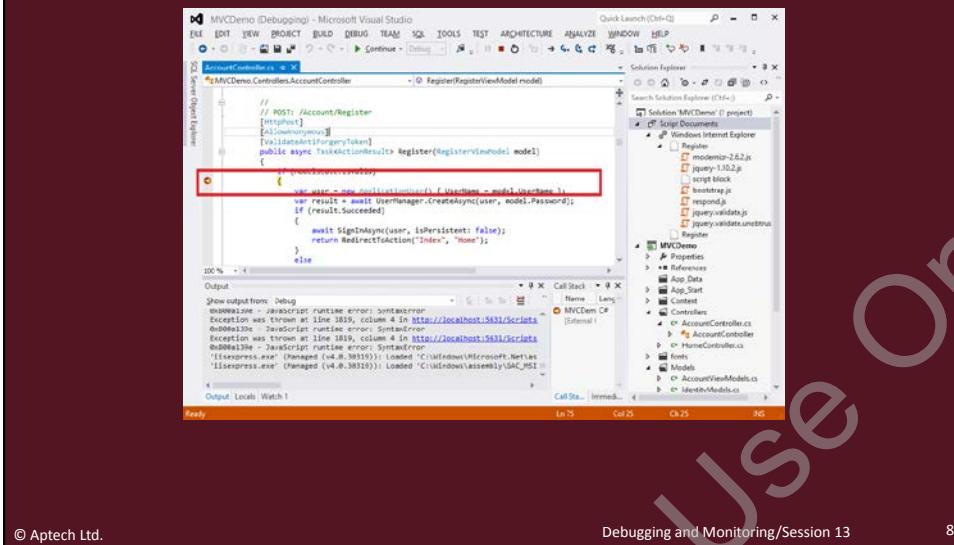
### Breakpoints 3-4

- ❖ Select Debug→Start Debugging from the menu bar of Visual Studio 2013. This will start the debugging process.
- ❖ Click on the Register link on the Home page that appears. The Registration page is displayed.
- ❖ In the Registration page, type a name in the User name text field, type a password in the Password field, and type the same password again in the Confirm password field.
- ❖ Click Register. When the program tries to execute the code where the breakpoint is added, the Code Editor displays the added breakpoint in yellow colour.

Use slide 7 to explain them the process of setting up a breakpoint in Visual Studio 2013, as displayed on the slide.

## Breakpoints 4-4

- Following figure shows the breakpoint when the program tries to execute it:



© Aptech Ltd.

Debugging and Monitoring/Session 13

8

In slide 8, you refer to the figure that shows the breakpoint when the program tries to execute it.

### Additional Information:

Most debugger features, such as viewing variable values in the Locals window or evaluating expressions in the Watch window, are available only in break mode. All the elements of your application remain, such as functions, variables, and objects remain in memory, but their movements and activities are suspended. During break mode, you can examine the elements' positions and states to look for violations or bugs. You can also make adjustments to the app while in break mode.

## Slide 9

Understand viewing runtime data.

### Viewing Runtime Data

- ◆ You view runtime data when the application pauses at a breakpoint.
- ◆ By viewing the runtime data, you can analyse whether or not the application is processing data as expected.
- ◆ To view runtime data, click an object when the application is currently paused at a breakpoint.
- ◆ A drop-down list displays the data of the object.
- ◆ Following figure shows the data of the Model object of the default application:

The screenshot shows a debugger interface with a code editor and a 'Registers' window. The code editor has a red box around a variable named 'model'. The 'Registers' window also has a red box around the same variable 'model'. Inside the 'Registers' window, there is a dropdown menu for the 'model' variable, which is expanded to show its properties: 'UserName' (Value: 'mark'), 'Password' (Value: 'mark123'), and 'ConfirmPassword' (Value: 'mark123').

© Aptech Ltd.      Debugging and Monitoring/Session 13      9

Display slide 9 and explain the students that they can view runtime data when the application pauses at a breakpoint. By viewing the runtime data, they can analyse whether or not the application is processing data as expected.

Explain them that to view runtime data, click an object when the application is currently paused at a breakpoint. A drop-down list displays the data of the object.

Then refer to the figure displayed on slide 9 that shows the data of the Model object of the default application.

## Slide 10

Understand code stepping.

### Code Stepping

- ◆ Once you set up breakpoints, you can use certain commands to step through your code line by line.
- ◆ This process of executing the code line by line is known as code stepping.
- ◆ In Visual Studio 2013, the Debug menu contains the following steps for debugging procedures:
  - ❖ Step into: You can use this command when you need to look inside a function call.
  - ❖ Step over: You can use this command when you want to avoid stepping into functions.
  - ❖ Step out: You can use this command when you are inside a function call and want to return to the calling function.

© Aptech Ltd.      Debugging and Monitoring/Session 13      10

Display slide 10 and explain the students that once they set up breakpoints, they can use certain commands to step through the code line by line. This process of executing the code line by line is known as code stepping.

Explain them about the following steps of debugging procedures that the Debug menu contains in Visual Studio 2013:

- **Step into:** Use this command when they need to look inside a function call.
- **Step over:** Use this command when they want to avoid stepping into functions.
- **Step out:** Use this command when they are inside a function call and want to return to the calling function.

**Slide 11**

Understand health monitoring.

**Health Monitoring**

- ◆ Once you deploy an ASP.NET MVC application, you need to constantly monitor it for its proper functioning.
- ◆ By monitoring an application, you can detect the problems occurring in the application and troubleshoot them.
- ◆ ASP.NET provides health monitoring features that you can use to check for any problems in your application and then use appropriate troubleshooting technique.
- ◆ Using these features, you can monitor your application while it is running for any issues that could affect its performance.

© Aptech Ltd.      Debugging and Monitoring/Session 13      11

Display slide 11 and explain that once they deploy an ASP.NET MVC application, they need to constantly monitor it for its proper functioning. By monitoring an application, they can detect the problems occurring in the application and troubleshoot them.

Explain them that ASP.NET provides health monitoring features that they can use to check for any problems in an application and then use appropriate troubleshooting technique. Using these features, they can monitor an application while it is running for any issues that could affect its performance.

## Slides 12 and 13

Understand health monitoring features.

### Health Monitoring Features 1-2

- ◆ The health monitoring process:
  - ❖ Allows you to monitor the status of a deployed application.
  - ❖ Track system events and errors that affect the application performance.
  - ❖ Allows you to access the detailed run-time information about the resources that an ASP.NET MVC application uses.
- ◆ In Visual Studio 2013, the Debug menu contains a Start Performance Analysis command that you can use to check the performance of an application.
- ◆ To use the Start Performance Analysis command in Visual Studio 2013, Click Debug→Start Performance Analysis. The Output window displays the progress of the performance analysis of the application.
- ◆ Once the process of performance analysis is started you can access the application online and start performing the actions available in the application, such as register to the application and then logout.

© Aptech Ltd.      Debugging and Monitoring/Session 13      12

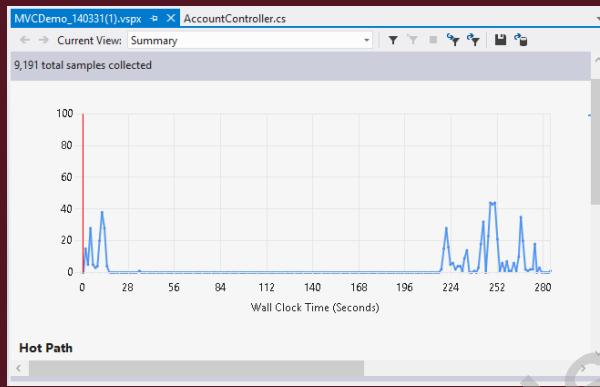
In slide 12, explain to the students that the health monitoring process allows monitoring the status of a deployed application. It allows them to track system events and errors that affect the application performance and to access the detailed run-time information about the resources that an ASP.NET MVC application uses.

Then tell them in Visual Studio 2013, the Debug menu contains a Start Performance Analysis command that they can use to check the performance of an application.

Finally explain the steps to use the Start Performance Analysis command in Visual Studio 2013 as displayed on slide 12.

## Health Monitoring Features 2-2

- ◆ Close the browser to exit the application.
- ◆ Following figure shows the Summary section of Visual Studio 2013 displays the performance of the application in a graphical way:



© Aptech Ltd.

Debugging and Monitoring/Session 13

13

Display slide 13 and refer to the figure that shows the Summary section of Visual Studio 2013 displays the performance of the application in a graphical way.

## Slides 14 and 15

Understand performance monitoring tools.

### Performance Monitoring Tools 1-2

- ◆ The Windows Operating System (OS) provides the Performance Monitor tool that you can use to identify any system-level and application-level performance issues.
- ◆ As a developer you can use the data that the Performance Monitor tool provides to identify issues related to your application or related to the existing hardware environment, such as memory, disk, processor, and network.
- ◆ In the Performance Monitor tool, each of the resources that you monitor is known as performance object.
- ◆ Further, each of these objects provides counters representing data on specific aspects of a system or service.

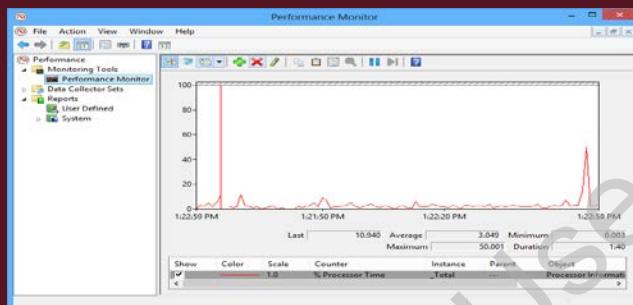
© Aptech Ltd.      Debugging and Monitoring/Session 13      14

Use slide 14 and explain that the Windows Operating System (OS) provides the Performance Monitor tool that they can use to identify any system-level and application-level performance issues. As a developer they can use the data that the Performance Monitor tool provides to identify issues related to your application or related to the existing hardware environment, such as memory, disk, processor, and network.

Explain that in the Performance Monitor tool, each of the resources that they monitor is known as performance object. Further, each of these object provides counters representing data on specific aspects of a system or service.

## Performance Monitoring Tools 2-2

- ◆ To use the Performance Monitor tool, you need to perform the following tasks:
  - ◆ Open Control Panel.
  - ◆ Click Administrative Tools icon in the Control Panel window. The Administrative Tool window is displayed.
  - ◆ Double-click Performance Monitor. The Performance Monitor window displays the performance of memory, processors, and physical disks.
- ◆ Following figure shows the shows the Performance Monitor window:



© Aptech Ltd.

Debugging and Monitoring/Session 13

15

In slide 15, explain the steps to use the Performance Monitor tool as displayed on the slide.

Then refer to the figure that shows the Performance Monitor window.

**Slides 16 and 17**

Understand analyzing monitoring results.

### Analyzing Monitoring Results 1-2

- ◆ Once you have viewed the default performance results, you can also analyze the monitoring data of different resources.
- ◆ To analyze the monitoring result of a selected counter, you need to perform the following steps:
  - ❖ Right-click the graph and select Add Counters from the context menu to add a counter to the monitor. The Add Counters dialog box is displayed.
  - ❖ Select a counter from the Select counters from the computer drop-down list.
  - ❖ Select the counters you want to monitor and click the Add button. The Added Counters section displays the currently added counter.
  - ❖ Click OK. The Performance Monitor window displays the real-time state of the newly added counter graphically.

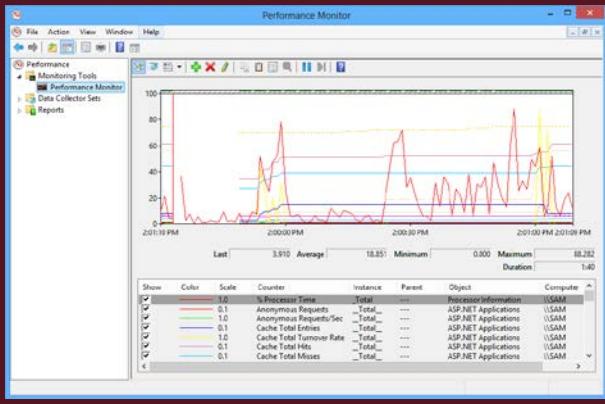
© Aptech Ltd.      Debugging and Monitoring/Session 13      16

In slide 16, explain the students that once they have viewed the default performance results, they can also analyze the monitoring data of different resources.

Then explain the steps to analyze the monitoring result of a selected counter, as displayed on slide 16.

### Analyzing Monitoring Results 2-2

- Following figure shows the real-time state of the newly added counter:



The screenshot shows the Windows Performance Monitor interface. The main area displays a line graph with multiple data series. The x-axis represents time from 20:07:19 PM to 20:10:19 PM. The y-axis ranges from 0 to 100. There are several colored lines: red, yellow, green, blue, and orange, each representing a different performance counter. Below the graph, a status bar shows metrics: Last 3.910, Average 18.851, Minimum 0.800, Maximum 88.282, and Duration 140. At the bottom, a legend lists the counters with their names and object paths: Total Requests/Sec, Anonymous Requests/Sec, Cache Total Entries, Cache Total Eviction Rate, Cache Total Hits, and Cache Total Misses, all under the object path ASP.NET Applications \SAM.

© Aptech Ltd.      Debugging and Monitoring/Session 13      17

Display slide 17 and refer to the figure that shows the real-time state of the newly added counter.

**Slides 18 and 19**

Understand Event Viewer.

**Using Event Viewer 1-2**

- ◆ The Microsoft Management Console (MMC) provides an interface known as Event Viewer.
- ◆ You can use this interface to view the entries of various event logs.
- ◆ To view event logs, you need to perform the following steps:
  - ❖ Open Control Panel.
  - ❖ Click Administrative Tools icon in the Control Panel window. The Administrative Tools window is displayed.
  - ❖ Double-click the Event Viewer. The Event Viewer window is displayed. By default, it displays the Application, Security, and System logs.

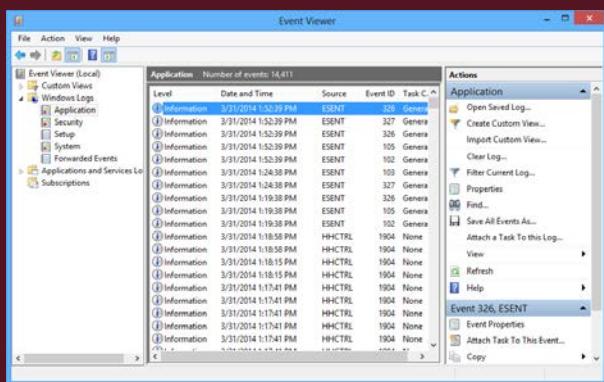
© Aptech Ltd.      Debugging and Monitoring/Session 13      18

Use slide 18 to explain the students that the Microsoft Management Console (MMC) provides an interface known as Event Viewer. They can use this interface to view the entries of various event logs.

Then explain the steps to view event logs as displayed on slide 18.

## Using Event Viewer 2-2

- ◆ Following figure shows the Event Viewer window:



- ◆ Click any one of the log to view its entries.

In slide 19, you refer to the figure that shows the Event Viewer window.

## Slide 20

Let us summarize the session.

**Summary**

- ◆ Debugging is a technique that enables you to resolve such errors present in the application.
- ◆ Visual Studio debugger enables you to run your code line by line, so that you can monitor the program execution properly.
- ◆ Breakpoints are places that you can specify in the code where the debugger stops the execution of the application and thus you can view the current state of data in your application.
- ◆ You view runtime data when the application pauses at a breakpoint so that, you can analyze whether or not the application is processing data as expected.
- ◆ Once you analyze the runtime data, you can use code stepping that allows executing the code line by line.
- ◆ The health monitoring process allows you to monitor the status of a deployed application.
- ◆ The Windows Operating System (OS) provides the Performance Monitor tool that you can use to identify any system-level and application-level performance issues.

© Aptech Ltd.      Debugging and Monitoring/Session 13      20

In slide 20, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- Debugging is a technique that enables you to resolve such errors present in the application.
- Visual Studio debugger enables you to run your code line by line, so that you can monitor the program execution properly.
- Breakpoints are places that you can specify in the code where the debugger stops the execution of the application and thus you can view the current state of data in your application.
- You view runtime data when the application pauses at a breakpoint so that, you can analyze whether or not the application is processing data as expected.
- Once you analyze the runtime data, you can use code stepping that allows executing the code line by line.
- The health monitoring process allows you to monitor the status of a deployed application.
- The Windows Operating System (OS) provides the Performance Monitor tool that you can use to identify any system-level and application-level performance issues.

### **13.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. For what purpose will you use the Visual Studio Debugger and Performance Monitor tools?
2. What tools will you use to view variable values or evaluating expressions in an application?

## Session 14 -

# Advanced Concepts of ASP.NET MVC

---

### **14.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **14.1.1 Objectives**

After the session, the learners will be able to:

- Explain request handling
- Define and describe filters
- Explain Web API

#### **14.1.2 Teaching Skills**

To teach this session successfully, you must know about request handling in an ASP.NET MVC application. You should have the knowledge of MVC filters. You should also be aware Web API.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

## Overview of the Session

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to the request handling process in an ASP.NET MVC application. They will learn about how to use filters in an application. This session will also discuss about Web API.

### 14.2 In-Class Explanation

#### Slides 3 and 4

Understand request handling.

### Request Handling 1-2

- ◆ MVC Framework provides several components that together function to handle requests coming from a client.
- ◆ Following figure shows how MVC Framework performs request handling:

The diagram illustrates the four-step request handling process within the MVC Framework:

- 1. Routing**: A blue box where the request enters the framework.
- 2. Controller creation**: A green box where a controller is generated based on the route.
- 3. Action execution**: A red box where the controller's action method is executed.
- 4. View generation**: An orange box where a view is generated to display the results.

The process starts with a **Browser** sending a **Request** to the **MVC Framework**. The framework then follows these steps in sequence: Routing, Controller creation, Action execution, and finally View generation. The final **Response** is sent back to the **Browser**.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 3

Use slide 3 to explain that MVC Framework provides several components that together function to handle requests coming from a client.

Then refer to the figure displayed on slide 3 that shows how MVC Framework performs request handling.

## Request Handling 2-2

- ◆ In the preceding figure, the different steps to handle a request are as follows:
  - ❖ Routing: In this step, the Framework maps the URL of the incoming request with URL patterns present in the route table. If a matching pattern is found the request is forwarded to an IRouteHandler object.
  - ❖ Controller creation: In this step the MvcHandler object uses the IControllerFactory objects and creates a controller object that implements the IController interface.
  - ❖ Action execution: In this step the CreateActionInvoker() method of the controller object is called to obtain a ControllerActionInvoker object that determines and executes the action.
  - ❖ View generation: In this step a view engine is created that generates the view based on the ActionResult object. The view is then, sent as a response to the client.

In slide 4, refer to the figure displayed on the slide 3 previous slide and explain the different steps required to handle a request as displayed on slide 4. Explain Routing, Controller creation, Action execution, and View generation.

## Slide 5

Understand route handler.

<b>Route Handler</b>
<ul style="list-style-type: none"><li>◆ In an ASP.NET MVC application, when a request is received, the routing module matches the URL of the incoming request with route constraints defined for the application.</li><li>◆ Once a URL matches with a route constraint, the routing module returns a handler for this route.</li><li>◆ This handler is referred to as a route handler for the request and is an implementation of the IRouteHandler interface.</li><li>◆ The default route handler of MVC Framework is MvcRouteHandler.</li><li>◆ However, at times you might need to create custom route handler to handle requests.</li></ul>

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 5

In slide 5, tell the students that in an ASP.NET MVC application, when a request is received, the routing module matches the URL of the incoming request with route constraints defined for the application. Once a URL matches with a route constraint, the routing module returns a handler for this route. This handler is referred to as a route handler for the request and is an implementation of the IRouteHandler interface.

Then tell them that the default route handler of MVC Framework is MvcRouteHandler. However, at times they might need to create custom route handler to handle requests.

## Slides 6 to 8

Understand HTTP handler.

**HTTP Handler 1-3**

- ◆ The HTTP handler handles HTTP requests that the route handler receives as a URL.
- ◆ For a request to an XML file, the HTTP handler will be responsible for loading the file and sending the content of the file as a response.
- ◆ To create an HTTP handler to process requests for XML files, you need to create a class that implements the `IHttpHandler` interface and override the `ProcessRequest()` method.
- ◆ This method accepts an `HttpContext` object that represents information about the HTTP request object.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 6

In slide 6, explain that the HTTP handler handles HTTP requests that the route handler receives as a URL. For a request to an XML file, the HTTP handler will be responsible for loading the file and sending the content of the file as a response.

Then explain them that to create an HTTP handler to process requests for XML files, they need to create a class that implements the `IHttpHandler` interface and override the `ProcessRequest()` method. This method accepts an `HttpContext` object that represents information about the HTTP request object.

### HTTP Handler 2-3

- Following code snippet shows using the CustomHttpHandler class, that implements the IHttpHandler interface, which contains the ProcessRequest() method:

Snippet

```
public class CustomHttpHandler : IHttpHandler {
    public void ProcessRequest(HttpContext context) {
        RouteValueDictionary values =
            context.Request.RequestContext.RouteData.Values;
        string path = context.Server.MapPath("~/XML/" + values["name"] +
            ".xml");
        XDocument xdoc = XDocument.Load(path);
        context.Response.Write(xdoc);
        context.Response.Flush();
    }
    public bool IsReusable
    {
        get { return true; }
    }
}
```

In slide 7, you refer to the codes that shows using the CustomHttpHandler class, that implements the IHttpHandler interface, which contains the ProcessRequest() method. Explain to them when and how the ProcessRequest() method is called and what actions can be performed in this method.

### HTTP Handler 3-3

- ◆ In the preceding code:
  - ❖ The CustomHttpHandler class implements the IHttpHandler interface that contains the ProcessRequest() method. In this method the HttpContext.Request.RequestContext.RouteData.Values property is accessed to retrieve a RouteValueDictionary object that represents a collection of route values.
  - ❖ Then, the path to the requested URL is created and the Load() method of the XDocument class is called to load the XML file as an XDocument object that represents an XML document.
  - ❖ The HttpContext.Response.Write() method is called to send the content of the XML document as response.

In slide 8, you can refer to the code displayed on slide 7 and explain that in code the CustomHttpHandler class implements the IHttpHandler interface that contains the ProcessRequest() method.

In this method, the HttpContext.Request.RequestContext.RouteData.Values property is accessed to retrieve a RouteValueDictionary object that represents a collection of route values.

Then, the path to the requested URL is created and the Load() method of the XDocument class is called to load the XML file as an XDocument object that represents an XML document.

Finally, the HttpContext.Response.Write() method is called to send the content of the XML document as response.

## Slide 9

Understand filters.

**Filters**

- ◆ In an ASP.NET MVC application there might be situations where you need to implement some functionality before or after the execution of an action method.
- ◆ In such situations, you need to use filters.
- ◆ While developing an ASP.NET MVC application, you can use filters at the following levels:
  - ◆ An action method: When you use filters in an action method, the filter will execute only when the associated action method is accessed.
  - ◆ A controller: When you use filters in controller, the filter will execute for all the actions methods defined in the controller.
  - ◆ Application: When you use filters in an application, the filter will execute for all the actions methods present in the application.
- ◆ The ASP.NET MVC Framework supports different types of filters, such as authorization, action, result, and exception filters.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 9

In slide 9, explain the students that in an ASP.NET MVC application there might be situations where they need to implement some functionality before or after the execution of an action method. In such situations, they need to use filters.

Then explain them that while developing an ASP.NET MVC application, they can use filters at the following levels:

- **An action method:** When they use filters in an action method, the filter will execute only when the associated action method is accessed.
- **A controller:** When they use filters in controller, the filter will execute for all the actions methods defined in the controller.
- **Application:** When they use filters in an application, the filter will execute for all the actions methods present in the application.

Finally, tell them that the ASP.NET MVC Framework supports different types of filters, such as authorization, action, result, and exception filters.

**Slides 10 to 12**

Understand Authorization filters.

### Authorization Filters 1-3

- ◆ Authorization filters execute before an action method is invoked to make security decisions on whether to allow the execution of the action method or not.
- ◆ In ASP.NET MVC Framework, the AuthorizeAttribute class of the System.Web.Mvc namespace is an example of authorization filters.
- ◆ This class extends the FilterAttribute class and implements the IAuthorizationFilter interface.
- ◆ The authorization filter allows you to implement standard authentication and authorization functionality in your application.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 10

Use slide 10 to explain the students that authorization filters execute before an action method is invoked to make security decisions on whether to allow the execution of the action method or not. In ASP.NET MVC Framework, the AuthorizeAttribute class of the System.Web.Mvc namespace is an example of authorization filters. This class extends the FilterAttribute class and implements the IAuthorizationFilter interface.

Then mention them that the authorization filter allows you to implement standard authentication and authorization functionality in your application.

### Authorization Filters 2-3

- Following code snippet shows adding the Authorize attribute on the ViewPremiumProduct() method:

Snippet

```
[Authorize]
public ActionResult ViewPremiumProduct()
{
    return View();
}
```

- This code adds the [Authorize] attribute on the ViewPremiumProduct() action method.
- Once you have added the Authorize attribute on the ViewPremiumProduct() action method, the access to the corresponding view is restricted.
- Therefore, whenever any user tries to access this view, a page to authenticate the user is displayed.

In slide 11, refer to the code displayed on the slide that shows adding the Authorize attribute on the ViewPremiumProduct() method.

Explain them that this code adds the [Authorize] attribute on the ViewPremiumProduct() action method.

Finally, explain them that once they have added the Authorize attribute on the ViewPremiumProduct() action method, the access to the corresponding view is restricted. Therefore, whenever any user tries to access this view, a page to authenticate the user is displayed.

### Authorization Filters 3-3

- ◆ You can use the web.config file to specify the page to be displayed for user authentication.
- ◆ Following code snippet shows how to specify the page to be displayed for user authentication:

Snippet

```
<authentication mode="Forms">
  <forms loginUrl="~/Account/Login" timeout="1440" />
</authentication>
```

- ◆ In this code:
  - ◆ The <forms> element specifies the login URL for the application.
  - ◆ Whenever a user tries to access a restricted resource, the user is redirected to the login URL.
  - ◆ The timeout attribute of the <forms> element specifies the amount of time in minutes, after which the authentication cookie expires. It is set to 1440 minutes. Its default value is 30 minutes.

In slide 12, explain the students that they can use the web.config file to specify the page to be displayed for user authentication.

Then you refer to the code displayed on slide 12 that shows how to specify the page to be displayed for user authentication.

Then explain them that in this code the <forms> element specifies the login URL for the application. Whenever a user tries to access a restricted resource, the user is redirected to the login URL. The timeout attribute of the <forms> element specifies the amount of time in minutes, after which the authentication cookie expires. It is set to 1440 minutes. Its default value is 30 minutes.

## Slides 13 to 19

Understand Action filters.

### Action Filters 1-7

- ◆ MVC action filters enable you to execute some business logic before and after an action method executes.
- ◆ When you use an action filter, the filter receives the following notifications from the MVC Framework for each action method that the filter applies to:
  - ◆ The Framework is about to execute the action.
  - ◆ The Framework has completed executing the action.
- ◆ While creating an ASP.NET application, you can create your own action filter as per requirements.
- ◆ To use a custom action filter in an application, you first need to create it and then, apply it to one or more target actions.
- ◆ You can create a custom action filter by implementing the `IActionFilter` interface.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 13

Use slide 13 to explain that MVC action filters enable executing some business logic before and after an action method executes. When they use an action filter, the filter receives the following notifications from the MVC Framework for each action method that the filter applies to:

- The Framework is about to execute the action.
- The Framework has completed executing the action.

Tell them that while creating an ASP.NET application, they can create your own action filter as per requirements. To use a custom action filter in an application, they first need to create it and then, apply it to one or more target actions. You can create a custom action filter by implementing the `IActionFilter` interface.

### Action Filters 2-7

- ◆ Following table describes the methods of the `IActionFilter` interface:

Method	Description
<code>void OnActionExecuting(ActionExecutingContext filterContext)</code>	This method is called before an action method is invoked. The <code>ActionExecutingContext</code> object represents the filter context that you can use to access information about the current controller, HTTP context, request context, action result, and route data.
<code>void OnActionExecuted (ActionExecutedContext filterContext)</code>	This method is called after invocation of an action method. The <code>ActionExecutedContext</code> object represents the filter context that you can use to access information about the current controller, HTTP context, request context, action result, and route data.

Use slide 14 and refer to the table displayed on the slide that describes the methods of the `IActionFilter` interface.

### Action Filters 3-7

- ◆ You can also extend the `ActionFilterAttribute` class to create a custom filter.
- ◆ This class implements the `IActionFilter` interface to provide a base class for filter attributes.
- ◆ To create a custom action filter that creates log messages, you need to create a class that extends from the `ActionFilterAttribute` class.
- ◆ Then, use the `OnActionExecuting()` method to specify the code to log messages.

In slide 15, explain them that they can also extend the `ActionFilterAttribute` class to create a custom filter. This class implements the `IActionFilter` interface to provide a base class for filter attributes.

Then explain the students that to create a custom action filter that creates log messages, they need to create a class that extends from the `ActionFilterAttribute` class. Then, use the `OnActionExecuting()` method to specify the code to log messages.

## Action Filters 4-7

- ◆ Following code shows the CustomActionFilterAttribute class:

Snippet

```
public class CustomActionFilterAttribute : ActionFilterAttribute {  
    public override void OnActionExecuting(ActionExecutingContext  
filterContext)  
    {  
        Debug.WriteLine("Action execution is about to start", "Action  
Filter Log");  
    }  
    . . .  
}
```

- ◆ This code uses the OnActionExecuting() method and logs a message when the execution of the action method is about to start.
- ◆ Next, in the OnActionExecuted() method you can log a message to indicate that the execution of the action method has been completed.

In slide 16, refer to the code that shows the CustomActionFilterAttribute class.

Tell them that this code uses the OnActionExecuting() method and logs a message when the execution of the action method is about to start. Next, in the OnActionExecuted() method you can log a message to indicate that the execution of the action method has been completed.

## Action Filters 5-7

- ◆ Following code shows the code of the OnActionExecuted() method:

Snippet

```
public override void OnActionExecuted(ActionExecutedContext filterContext)
{
    Debug.WriteLine("Action execution has completed", "Action Filter Log");
}
```

- ◆ This code uses the OnActionExecuted() method and creates a log message once the execution of the action method has been completed.
- ◆ Once you have created an action filter, you can apply it to the action method of a controller class.

In slide 17, refer to the code that shows the OnActionExecuted( ) method.

Tell them that this code uses the OnActionExecuted( ) method and creates a log message once the execution of the action method has been completed. Once they have created an action filter, they can apply it to the action method of a controller class.

## Action Filters 6-7

- ◆ Following code shows how to apply the CustomActionFilter filter at the Index() action method:

Snippet

```
public class HomeController : Controller
{
    [CustomActionFilter]
    public ActionResult Index()
    {
        return View();
    }
}
```

- ◆ This code applies the CustomActionFilter filter to the Index() action method of the Home controller.
- ◆ When you debug the application, the CustomActionFilter filter executes and logs the messages in the output window of Visual Studio 2013.

In slide 18, refer to the code that shows how to apply the CustomActionFilter filter at the Index( ) action method.

Tell them that this code applies the CustomActionFilter filter to the Index( ) action method of the Home controller. When they debug the application, the CustomActionFilter filter executes and logs the messages in the output window of Visual Studio 2013.

## Action Filters 7-7

- Following figure shows the log messages in the Output window:

The screenshot shows the Visual Studio Output window with the title 'Output' and the dropdown menu set to 'Debug'. The window displays log messages from the application 'iisexpress.exe'. A red box highlights two specific messages: 'Action Filter Log: Action execution is about to start' and 'Action Filter Log: Action execution has completed'. These messages are part of a larger log output that includes assembly loading information for 'iisexpress.exe'.

```
iisexpress.exe (CLR v4.0.30319: /LM/W3SVC/20/ROOT-1-130411582669331062); Loaded 'C:\Windows\Microsoft.NET\assembly\GAC_Web\iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/20/ROOT-1-130411582669331062); Loaded 'C:\Windows\Microsoft.NET\assembly\GAC_Web\iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/20/ROOT-1-130411582669331062); Loaded 'Anonymously Hosted DynamicMethods Asse  
Action Filter Log: Action execution is about to start  
Action Filter Log: Action execution has completed  
iisexpress.exe (CLR v4.0.30319: /LM/W3SVC/20/ROOT-1-130411582669331062); Loaded 'C:\Windows\Microsoft.NET\assembly\GAC_Web\iisexpress.exe' (CLR v4.0.30319: /LM/W3SVC/20/ROOT-1-130411582669331062); Loaded 'C:\Users\Admin\AppData\Local\Temp\Temp
```

© Aptech Ltd.

Advanced Concepts of ASP.NET MVC/Session 14

19

In slide 19, refer to the figure that shows the log messages in the Output window.

## Slides 20 to 22

Understand Result filters.

### Result Filters 1-3

- ◆ A result filter operates on the result that an action returns.
- ◆ When you implement a result filter, the filter will receive the following notifications from the MVC Framework for each action method that the filter applies to:
  - ◆ The action method is about to execute the action result.
  - ◆ The action method has completed executing the action result.
- ◆ The OutputCacheAttribute class is one example of a result filter, which is used to mark an action method whose output will be cached.
- ◆ The OutputCache filter indicates the MVC Framework to cache the output from an action method.
- ◆ The same content can be reused to service subsequent requests for the same URL.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 20

In slide 20, explain that a result filter operates on the result that an action returns. When they implement a result filter, the filter will receive the following notifications from the MVC Framework for each action method that the filter applies to:

- The action method is about to execute the action result.
- The action method has completed executing the action result.

The OutputCacheAttribute class is one example of a Result filter, which is used to mark an action method whose output will be cached. The OutputCache filter indicates the MVC Framework to cache the output from an action method. The same content can be reused to service subsequent requests for the same URL.

### Result Filters 2-3

- ◆ Caching action output can offer a significant increase in performance, because most of the time-consuming activities required to process a request are avoided.
- ◆ Following code snippet shows how to use the OutputCache attribute:

Snippet

```
public class HomeController : Controller {  
    [OutputCache]  
    public ActionResult Index() {  
        //some code  
    }  
}
```

- ◆ In this code, the [OutputCache] attribute is added to the Index() action method of the Home controller.
- ◆ You can specify the duration for which the output of the action should be cached by specifying a Duration property with the duration time in seconds.

In slide 21, explain the students that caching action output can offer a significant increase in performance, because most of the time-consuming activities required to process a request are avoided.

Then you refer to the code that shows how to use the OutputCache attribute.

Explain them that in this code, the [OutputCache] attribute is added to the Index() action method of the Home controller. They can specify the duration for which the output of the action should be cached by specifying a Duration property with the duration time in seconds.

### Result Filters 3-3

- ◆ Following code shows specifying the Duration property:

Snippet

```
public class HomeController : Controller
{
    [OutputCache(Duration=3)]
    public ActionResult Index()
    {
        //some code
    }
}
```

- ◆ In this code, the Duration property of the OutputCache attribute is set to cache the output for 3 seconds.

In slide 22, refer to the code that shows specifying the Duration property.

Explain them that in this code, the Duration property of the OutputCache attribute is set to cache the output for 3 seconds.

## Slides 23 to 27

Understand Exception filters.

### Exception Filters 1-5

- ◆ In an ASP.NET MVC application, you can use exception filters to handle exceptions that the application throws at runtime.
- ◆ Exception filters are additional exception handling component of MVC Framework besides the built-in .NET Framework exception handling mechanism comprising try-catch block.
- ◆ The MVC Framework provides a built-in exception filter through the HandleError filter that the HandleErrorAttribute class implements.
- ◆ Like other filters, you can use the HandleError filter on an action method or a controller.
- ◆ The HandleError filter handles the exceptions that are raised by the controller actions and other filters applied to the action.
- ◆ This filter returns a view named Error.cshtml which by default is in the shared folder of the application.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 23

In slide 23, explain the students that in an ASP.NET MVC application, they can use exception filters to handle exceptions that the application throws at runtime. Exception filters are additional exception handling component of MVC Framework besides the built-in .NET Framework exception handling mechanism comprising try-catch block. The MVC Framework provides a built-in exception filter through the HandleError filter that the HandleErrorAttribute class implements.

Then tell them that like other filters, they can use the HandleError filter on an action method or a controller. The HandleError filter handles the exceptions that are raised by the controller actions and other filters applied to the action. This filter returns a view named Error.cshtml which by default is in the shared folder of the application.

## Exception Filters 2-5

- Following code snippet shows the Error.cshtml view that displays an error message whenever an action with a HandleError filter throws an exception:

Snippet

```
@model System.Web.Mvc.HandleErrorInfo  
{@  
    ViewBag.Title = "Error";  
}  
<h1 class="text-danger">Error.</h1>  
<h2 class="text-danger">An error occurred while processing your  
request.</h2>
```

- This code displays an error message whenever an action with a HandleError filter throws an exception.
- To use the HandleError filter, you need to configure the web.config file by adding a customErrors attribute inside the <system.web> element.

In slide 24, refer to the code that shows the Error.cshtml view that displays an error message whenever an action with a HandleError filter throws an exception.

Explain the students that this code displays an error message whenever an action with a HandleError filter throws an exception. To use the HandleError filter, they need to configure the web.config file by adding a customErrors attribute inside the <system.web> element.

### Exception Filters 3-5

- Following code snippet shows how to enable custom errors in the web.config file:

Snippet

```
...  
  <system.web>  
    ...  
      <customErrors mode="On" />  
    </system.web>  
...  
...
```

- In this code, the On value of the mode attribute in the `<customErrors>` element enables exception handling using the `HandleError` filter.
- To test how the `HandleError` filter works, you can update the `Index()` action method of the Home controller to throw an exception and handle it using the `HandleError` filter.

In slide 25, refer to the code that shows how to enable custom errors in the web.config file.

Then explain them that in this code, the On value of the mode attribute in the `<customErrors>` element enables exception handling using the `HandleError` filter. To test how the `HandleError` filter works, they can update the `Index()` action method of the Home controller to throw an exception and handle it using the `HandleError` filter.

### Exception Filters 4-5

- Following code snippet shows how to use the HandleError filter on the Index() action method of the Home controller that throws an exception:

Snippet

```
public class HomeController : Controller
{
    [HandleError]
    public ActionResult Index()
    {
        throw new DivideByZeroException();
        return View();
    }
    .
}
```

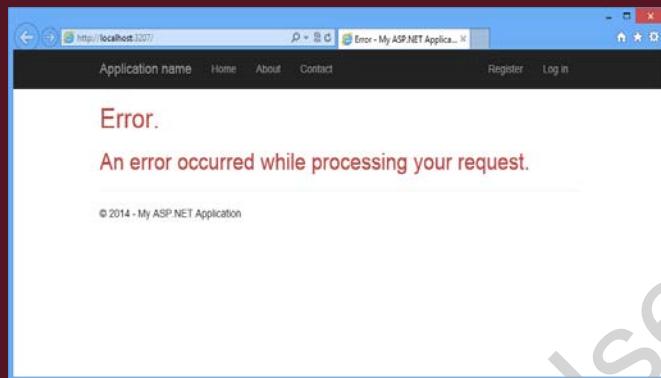
- This code applies the HandleError filter to the Index() action method of the Home controller. The Index() action method throws an exception of type DivideByZeroException.

Use slide 26 to refer to the code that shows how to use the HandleError filter on the Index() action method of the Home controller that throws an exception.

Explain them that this code applies the HandleError filter to the Index() action method of the Home controller. The Index() action method throws an exception of type DivideByZeroException.

### Exception Filters 5-5

- ◆ When you access the Index() action method of the Home controller, the HandleError filter will catch the exception of type DivideByZeroException and display the Error.cshtml view.
- ◆ Following figure shows the Error.cshtml view:



© Aptech Ltd.

Advanced Concepts of ASP.NET MVC/Session 14

27

In slide 27, explain the students that when they access the Index( ) action method of the Home controller, the HandleError filter will catch the exception of type DivideByZeroException and display the Error.cshtml view.

You can refer to the figure that shows the Error.cshtml view.

## Slides 28 and 29

Understand Web API.

### Introduction to Web API 1-2

- ◆ Web API is a Framework that allows you to easily create Web services that provide an API for a wide range of clients, such as browsers and mobile devices using the HTTP protocol.
- ◆ Using Web API you can display data based on the client requests.
- ◆ You can use Web API to develop HTTP based services where client can make a GET, PUT, POST, and DELETE requests and access the response provided by Web API.
- ◆ Web API allows you to add special controllers, known as API Controller.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 28

In slide 28, tell the students that Web API is a Framework that allows easily creating Web services that provide an API for a wide range of clients, such as browsers and mobile devices using the HTTP protocol. Using Web API they can display data based on the client requests.

Tell them that they can use Web API to develop HTTP based services where client can make a GET, PUT, POST, and DELETE requests and access the response provided by Web API. Web API allows them to add special controllers, known as API Controller.

#### Additional Information:

Web API is a framework that allows easily building HTTP services that reach a wide range of users, including browsers and mobile devices. With Web API content negotiation, we can return data based on the users requests. When a user requests the data to be returned as JSON or XML, the Web API framework deals with the request type and returns the data appropriately based on the media type. By default Web API provides JSON and XML based responses.

## Introduction to Web API 2-2

- ◆ The main characteristics of an API controller are as follows:
  - ❖ The action methods return model instead of the ActionResult, objects.
  - ❖ The action methods selected based on the HTTP method used in the request.
- ◆ The model objects that are returned from an API controller action method are encoded as JSON and sent to the client.
- ◆ In addition, as API controllers deliver Web data services, so they do not support views, layouts to generate HTML for browsers to display.

In slide 29, tell the students about the main characteristics of an API controller as displayed on the slide.

Then tell them that the model objects that are returned from an API controller action method are encoded as JSON and sent to the client. In addition, as API controllers deliver Web data services, so they do not support views, layouts to generate HTML for browsers to display.

**Slides 30 to 36**

Understand creating a Web API application.

### Creating a Web API Application 1-7

- ◆ A Web API application is just a regular MVC Framework application with the addition of a special kind of controller.
- ◆ Visual Studio 2013 allows you to create and build Web API application.
- ◆ To create a new Web API project in Visual Studio 2013, you need to perform the following steps:
  - ◆ Open Visual Studio 2013.
  - ◆ Click File→New→Projects menu options in the menu bar of Visual Studio 2013.
  - ◆ In the New Project dialog box that appears, select Web under the Installed section and then, select the ASP.NET Web Application template.
  - ◆ Type WebAPIDemo in the Name text field.
  - ◆ Click the Browse button and specify the location where the application has to be created.

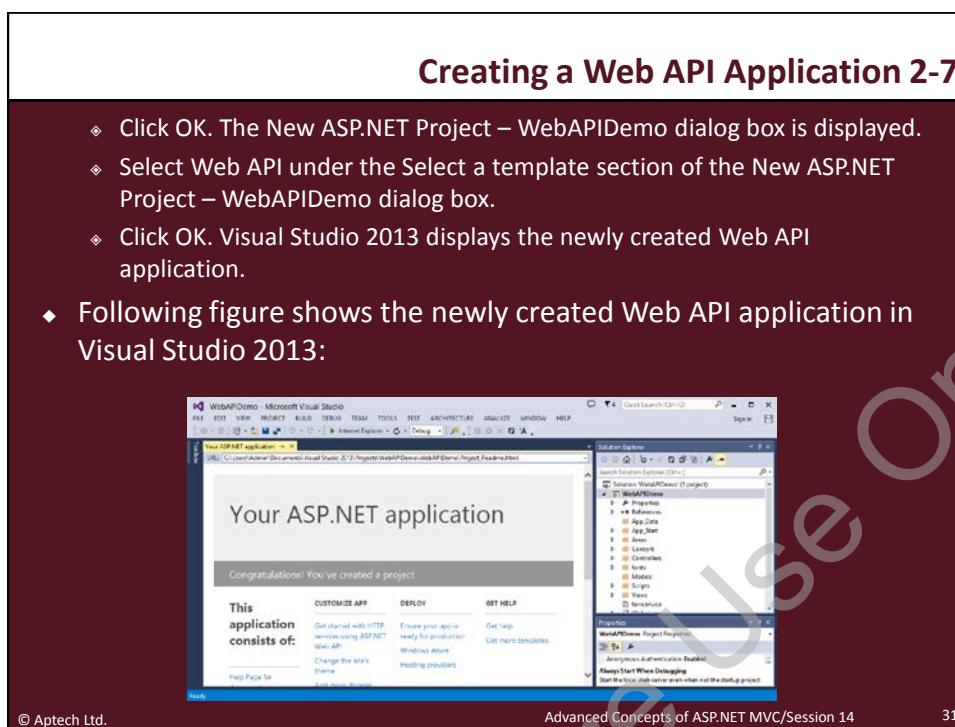
© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 30

In slide 30, explain the students that a Web API application is just a regular MVC Framework application with the addition of a special kind of controller. Visual Studio 2013 allows creating and building Web API application.

Then explain the steps to create a new Web API project in Visual Studio 2013, as displayed on slide 30.

## Creating a Web API Application 2-7

- ◆ Click OK. The New ASP.NET Project – WebAPIDemo dialog box is displayed.
- ◆ Select Web API under the Select a template section of the New ASP.NET Project – WebAPIDemo dialog box.
- ◆ Click OK. Visual Studio 2013 displays the newly created Web API application.
- ◆ Following figure shows the newly created Web API application in Visual Studio 2013:



In slide 31, refer to the figure that shows the newly created Web API application in Visual Studio 2013.

### Creating a Web API Application 3-7

- ◆ Once you have created the Web API application, you need to create a model.
- ◆ To create a model in Visual Studio 2013, you need to perform the following steps:
  - ❖ Right-click the Model folder in the Solution Explorer window and select Add→Class from the context menu that appears. The Add New Item – WebAPIDemo dialog box is displayed.
  - ❖ Type Product.cs in the Name text field of the Add New Item – WebAPIDemo dialog box.
  - ❖ Click Add. The Code Editor displays the newly created Product class.
  - ❖ In Code Editor add the code to the Product class to represent a product.

In slide 32, explain the students that once they have created the Web API application, they need to create a model.

Next, explain the steps to create a model in Visual Studio 2013, as displayed on slide 32.

## Creating a Web API Application 4-7

- Following code shows the Product class:

Snippet

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Web;
namespace WebAPIDemo.Models
{
    public class Product
    {
        public int Id { get; set; }
        public string Category { get; set; }
        public string Name { get; set; }
    }
}
```

- This code declares three variables named Id, Category, and Name along with the get and set methods.
- Once you have created the model named, Product.cs you can add a Web API controller to the application.

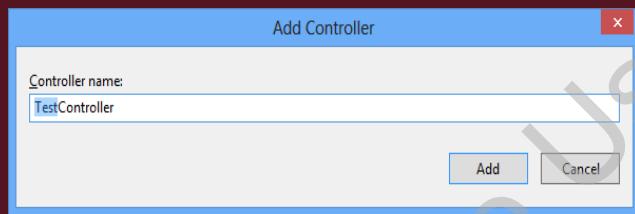
Display slide 33 and refer to the code displayed on the slide.

Then explain them that this code declares three variables named Id, Category, and Name along with the get and set methods.

Then tell them that once they have created the model named, Product.cs you can add a Web API controller to the application.

### Creating a Web API Application 5-7

- ◆ In Visual Studio 2013, to create a Web API controller, you need to perform the following steps:
  - ❖ Right-click the Controllers folder in the Solution Explorer window and select Add→Controller from the context menu that appears. The Add Scaffold dialog box is displayed.
  - ❖ Select the Web API 2 Controller – Empty template in the Add Scaffold dialog box.
  - ❖ Click Add. The Add Controller dialog box is displayed.
  - ❖ Type TestController in the Controller name field.
- ◆ Following figure shows the Add Controller dialog box:



© Aptech Ltd.

Advanced Concepts of ASP.NET MVC/Session 14

34

In slide 34, explain the steps to create a Web API controller in Visual Studio 2013, as displayed on the slide.

## Creating a Web API Application 6-7

- ◆ Click Add. The Code Editor displays the newly created TestController controller class.
- ◆ In the TestController controller class, initialize a Product array with Product objects. Then, create a Get() method to return the Product array
- ◆ Following code shows the TestController controller class:

Snippet

```
public class TestController : ApiController {
    Product[] products = new Product[] {
        new Product {Id = 1, Name = "Product 1", Category= "Category
1"},
        new Product {Id = 2, Name = "Product 2", Category= "Category
1"},
        new Product {Id = 3, Name = "Product 3", Category= "Category
2"},
    };
    public IEnumerable<Product> Get()
    {
        return products;
    }
}
```

In slide 35, refer to the code that shows the TestController controller class.

## Creating a Web API Application 7-7

- ◆ The preceding code:
  - ❖ Defines a Product array that contains three Product objects.
  - ❖ A Get() method is defined that will process GET requests that arrives to the controller.
  - ❖ The Get() method returns the Product array.

In slide 36, refer to the code on slide 35 and explain to the students that the preceding code defines a Product array that contains three Product objects. Then a Get( ) method is defined that will process GET requests that arrives to the controller. The Get( ) method returns the Product array.

**Slides 37 to 40**

Understand Routing in Web API.

**Defining Routing in Web API 1-4**

- ◆ Once you have created the Web API controller, you need to register it with the ASP.NET routing Framework.
- ◆ When the Web API application receives a request, the routing Framework tries to match the URI against one of the route templates defined in the WebApiConfig.cs file. If no route matches, the client receives a 404 error.
- ◆ When you create a Web API application in Visual Studio 2013, by default the IDE configures the route of the application in the WebApiConfig.cs file under the App\_Start folder.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 37

In slide 37, explain to the students once they have created the Web API controller, they need to register it with the ASP.NET routing Framework. When the Web API application receives a request, the routing Framework tries to match the URI against one of the route templates defined in the WebApiConfig.cs file. If no route matches, the client receives a 404 error.

Tell them that when they create a Web API application in Visual Studio 2013, by default the IDE configures the route of the application in the WebApiConfig.cs file under the App\_Start folder.

**Tips:**

Routing is how Web API matches a URI to an action. Web API 2 supports a new type of routing, called attribute routing. As the name implies, attribute routing uses attributes to define routes. Attribute routing gives you more control over the URIs in your Web API.

## Defining Routing in Web API 2-4

- ◆ Following code snippet shows the default route configuration of the Web API application:

Snippet

```
config.Routes.MapHttpRoute(  
    name: "DefaultApi",  
    routeTemplate: "api/{controller}/{id}",  
    defaults: new { id = RouteParameter.Optional }  
) ;
```

- ◆ This code shows the default route configuration for a Web API application.
- ◆ This route configuration contains a route template specified by the `routeTemplate` attribute that defines the following pattern:  
`"api/{controller}/{id}"`

In slide 38, refer to the code displayed on the slide that shows the default route configuration of the Web API application.

Next, explain them that this code shows the default route configuration for a Web API application. This route configuration contains a route template specified by the `routeTemplate` attribute that defines the following pattern:

`"api/{controller}/{id}"`

## Defining Routing in Web API 3-4

- ◆ In the preceding route pattern:
  - ❖ api: Is a literal path segment
  - ❖ {controller}: Is a placeholder for the name of the controller to access
  - ❖ {id}: Is an optional placeholder that the controller method accepts as parameter
- ◆ You can use the RouteTable.MapHttpRoute() method to configure routes of a Web API application.
- ◆ For example, consider the following URL:  
`http://localhost:9510/web/Test`
- ◆ Assuming that the preceding URL needs to access the TestController controller class of the WebAPIDemo application, you need to configure a route in the WebApiConfig.cs file.

In slide 39, refer to the code on slide 38 and explain that:

- api: Is a literal path segment
- {controller}: Is a placeholder for the name of the controller to access
- {id}: Is an optional placeholder that the controller method accepts as parameter

Tell them that they can use the `RouteTable.MapHttpRoute()` method to configure routes of a Web API application.

Explain the example as displayed on slide 39.

## Defining Routing in Web API 4-4

- ◆ Following code snippet shows how to configure a route:

Snippet

```
config.Routes.MapHttpRoute(  
    name: "TestDefaultApi",  
    routeTemplate: "web/{controller}/{id}",  
    defaults: new { id = RouteParameter.Optional }  
) ;
```

- ◆ In this code, a route named TestDefaultApi is created with the route pattern web/{controller}/{id}.

In slide 40, refer to the code that shows how to configure a route.

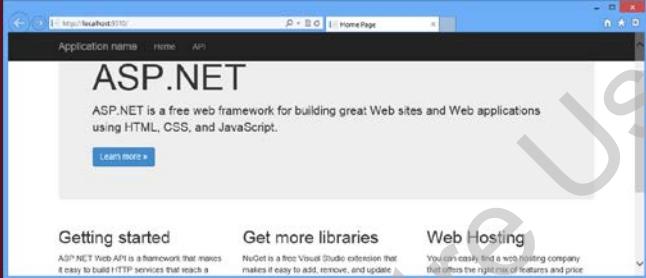
Then explain that in this code, a route named TestDefaultApi is created with the route pattern web/{controller}/{id}.

## Slides 41 and 42

Understand accessing an ASP.NET Web API application.

### Accessing an ASP.NET Web API Application 1-2

- ◆ Once you have created the controller class and configured the routing of the Web API application, you can access it over the browser.
- ◆ For that, you need to perform the following steps:
  - ❖ Click Debug → Start Without Debugging. The browser window displays the Home page of the application.
- ◆ Following figure shows the Home page of the application:



The screenshot shows a Microsoft Edge browser window with the URL `http://localhost:9110/` in the address bar. The title bar says "Application name: Home API". The main content area displays the "ASP.NET" logo and the text: "ASP.NET is a free web framework for building great Web sites and Web applications using HTML, CSS, and JavaScript." Below this, there are three links: "Getting started", "Get more libraries", and "Web Hosting". The "Getting started" link has a tooltip: "ASP.NET Web API is a framework that makes it easy to build HTTP services that reach a wide range of clients." The "Get more libraries" link has a tooltip: "NuGet is a free Visual Studio extension that makes it easy to add, remove, and update libraries and tools in your project." The "Web Hosting" link has a tooltip: "You can easily find a web hosting company that offers the right mix of features and price." At the bottom of the page, there is a copyright notice: "© Aptech Ltd." and a footer: "Advanced Concepts of ASP.NET MVC/Session 14".

In slide 41, explain the students that once they have created the controller class and configured the routing of the Web API application, they can access it over the browser.

Then explain the steps to access the application on browser, as displayed on slide 41.

You can then refer to the figure displayed on slide 41 that shows the Home page of the application.

## Accessing an ASP.NET Web API Application 2-2

- ❖ Type the following URL in the address bar of the browser:  
<http://localhost:9510/web/Test>
- ❖ Browser displays the product details, as shown in the following figure:

```
-<ArrayOfProduct>
- <Product>
  <Category>Category 1</Category>
  <Id>1</Id>
  <Name>Product 1</Name>
</Product>
- <Product>
  <Category>Category 1</Category>
  <Id>2</Id>
  <Name>Product 2</Name>
</Product>
- <Product>
  <Category>Category 2</Category>
  <Id>3</Id>
  <Name>Product 3</Name>
</Product>
</ArrayOfProduct>
```

In slide 42, refer to the figure that shows the browser displaying product details.

## Slide 43

Let us summarize the session.

**Summary**

- ◆ MVC Framework provides several components that together function to handle requests coming from a client.
- ◆ In an ASP.NET MVC application, when a request is received, the routing module matches the URL of the incoming request with route constraints defined for the application.
- ◆ The HTTP handler handles HTTP requests that the route handler receives as a URL.
- ◆ Authorization filters execute before an action method is invoked to make security decisions on whether to allow the execution of the action method.
- ◆ MVC action filters enable you to execute some business logic before and after an action method executes.
- ◆ Exception filters are additional exception handling component of MVC Framework besides the built-in .NET Framework exception handling mechanism comprising of try-catch block.
- ◆ Web API is a Framework that allows you to easily create Web services that provide an API for a wide range of clients, such as browsers and mobile devices using the HTTP protocol.

© Aptech Ltd. Advanced Concepts of ASP.NET MVC/Session 14 43

In slide 43, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- MVC Framework provides several components that together function to handle requests coming from a client.
- In an ASP.NET MVC application, when a request is received, the routing module matches the URL of the incoming request with route constraints defined for the application.
- The HTTP handler handles HTTP requests that the route handler receives as a URL.
- Authorization filters execute before an action method is invoked to make security decisions on whether to allow the execution of the action method.
- MVC action filters enable you to execute some business logic before and after an action method executes.
- Exception filters are additional exception handling component of MVC Framework besides the built-in .NET Framework exception handling mechanism comprising of try-catch block.
- Web API is a Framework that allows you to easily create Web services that provide an API for a wide range of clients, such as browsers and mobile devices using the HTTP protocol.

### **14.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. How will you create a custom filter?
2. What is Web API content negotiation?
3. What is the purpose of attribute routing?

# Session 15 -

## Testing and Deploying

### **15.1 Pre-Class Activities**

Before you commence the session, you should familiarize yourself with the topics of the current session in depth.

#### **15.1.1 Objectives**

After the session, the learners will be able to:

- Define and describe how to perform unit tests
- Explain how to prepare an application for deployment
- Define and describe how to deploy an application on IIS

#### **15.1.2 Teaching Skills**

To teach this session successfully, you must know about unit tests in an ASP.NET MVC application. You should have the knowledge of how to prepare an application for deployment. You should also be aware of how to deploy an application on IIS.

You should teach the concepts in the theory class using slides and LCD projectors.

#### **Tips:**

It is recommended that you test the understanding of the students by asking questions in between the class.

#### **In-Class Activities**

Follow the order given here during In-Class activities.

#### **Overview of the Session**

Give the students a brief overview of the current session in the form of session objectives. Show the students slide 2 of the presentation. Tell them that they will be introduced to unit testing in an ASP.NET MVC application. They will learn about preparing an application for deployment. This session will also discuss about how to deploy an application on IIS.

## 15.2 In-Class Explanation

### Slide 3

Understand testing.

**Testing**

- ◆ Once you have created an application, you should ensure that the application works properly by testing it thoroughly to provide the best quality possible.
- ◆ Sometimes, an application may work properly at the time of development.
- ◆ However, when it is deployed, you cannot be sure of the inputs that the users may provide to the application and the results that the application may generate in such a scenario.
- ◆ To avoid such problems, you should test the application before you deploy it.

© Aptech Ltd.      Testing and Deploying/Session 15      3

Use slide 3 to explain that once they have created an application, they should ensure that the application works properly by testing it thoroughly to provide the best quality possible.

Tell them that sometimes, an application may work properly at the time of development. However, when it is deployed, they cannot be sure of the inputs that the users may provide to the application and the results that the application may generate in such a scenario. To avoid such problems, they should test the application before deploying it.

**Slides 4 to 6**

Understand preparing for unit tests.

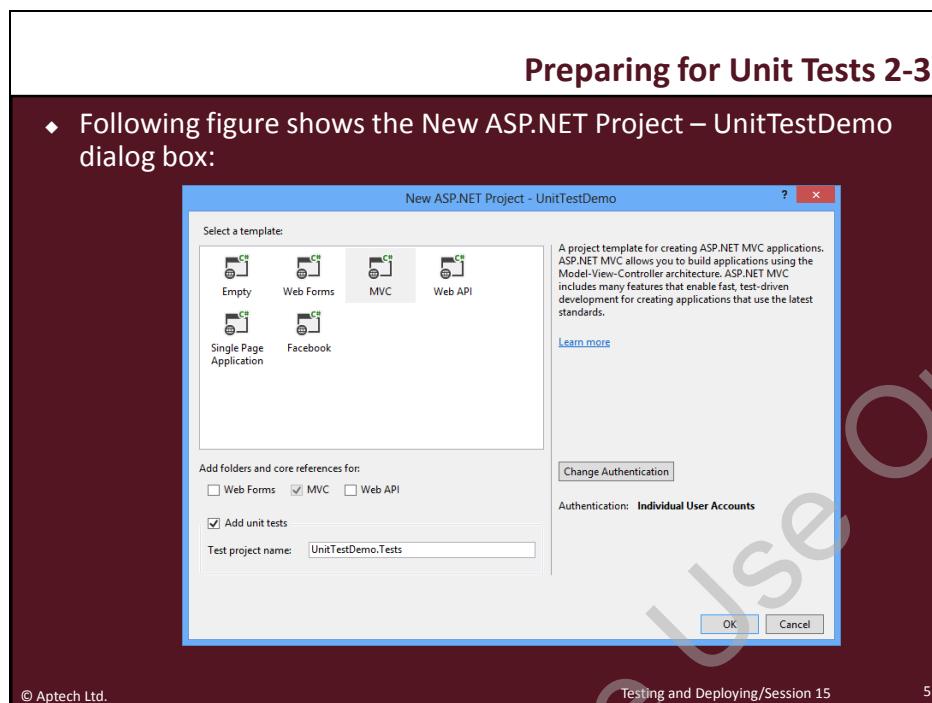
### Preparing for Unit Tests 1-3

- ◆ Unit testing:
  - ❖ Is a technique that allows you to create classes and methods in your application and test their intended functionality.
  - ❖ Is the smallest part of an application that can be tested.
  - ❖ Is concerned whether or not the individual units that make up the application functions as expected.
- ◆ When you create an ASP.NET MVC application in Visual Studio 2013, you can specify whether to create a unit test for the application.
- ◆ To create an application in Visual Studio 2013 with unit test, you need to perform the following steps:
  - ❖ Create a new ASP.NET Web Application project, named UnitTestDemo.
  - ❖ Click OK. The New ASP.NET Project – UnitTestDemp dialog box appears.
  - ❖ Select MVC under the Select a template section and select the Add unit tests check box.

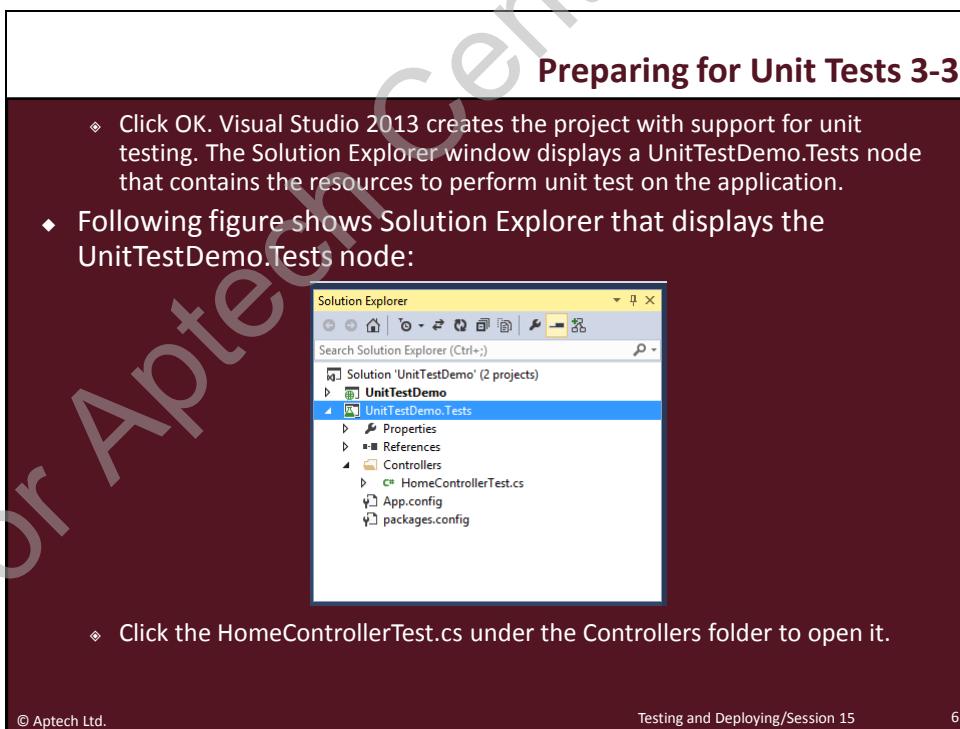
In slide 4, explain them that unit testing is a technique that allows creating classes and methods in an application and test their intended functionality. It is the smallest part of an application that can be tested. Unit testing is concerned whether or not the individual units that make up the application functions as expected.

Explain them that when they create an ASP.NET MVC application in Visual Studio 2013, they can specify whether to create a unit test for the application.

Then explain the steps to create an application in Visual Studio 2013 with unit test, as displayed on the slide 4.



In slide 5, refer to the figure that shows the New ASP.NET Project – UnitTestDemo dialog box.



In slide 6, refer to the figure that shows the Solution Explorer that displays the UnitTestDemo.Tests node.

**Slides 7 to 10**

Understand unit test classes and methods.

### Unit Test Classes and Methods 1-4

- ◆ A unit test class uses the [TestClass] attribute in the class declaration to indicate that it is a unit test class.
- ◆ Next, for each action method present in the target controller that needs to be tested, the unit test class has a corresponding method with the [TestMethod] attribute applied to it.
- ◆ To test an action method that passes data to the view using a ViewBag object, the test method uses the Assert.AreEqual() method.
- ◆ This method accepts the message being passed to the view as the first parameter and ViewResult.ViewBag.Message property as the second parameter.

Display slide 7 and explain that a unit test class uses the [TestClass] attribute in the class declaration to indicate that it is a unit test class. Next, for each action method present in the target controller that needs to be tested, the unit test class has a corresponding method with the [TestMethod] attribute applied to it.

Tell them that to test an action method that passes data to the view using a ViewBag object, the test method uses the Assert.AreEqual( ) method. This method accepts the message being passed to the view as the first parameter and ViewResult.ViewBag.Message property as the second parameter.

## Unit Test Classes and Methods 2-4

- Following code shows the unit test class, named HomeControllerTest:

Snippet

```
using System;
using System.Collections.Generic;
using System.Linq;
using System.Text;
using System.Web.Mvc;
using Microsoft.VisualStudio.TestTools.UnitTesting;
using UnitTestDemo;
using UnitTestDemo.Controllers;
namespace UnitTestDemo.Tests.Controllers{
    [TestClass]
    public class HomeControllerTest
    {
        [TestMethod]
        public void Index()
        {
            // Arrange
            HomeController controller = new HomeController();
            // Act
            ViewResult result = controller.Index() as ViewResult;
```

© Aptech Ltd.

Testing and Deploying/Session 15

8

Using slide 8, refer to the code that shows the unit test class, named HomeControllerTest.

## Unit Test Classes and Methods 3-4

### Snippet

```
// Assert
    Assert.IsNotNull(result);
}
[TestMethod]
public void About()
{
    // Arrange
    HomeController controller = new HomeController();
    // Act
    ViewResult result = controller.About() as ViewResult;
    // Assert
    Assert.AreEqual("Your application description page.",
        result.ViewBag.Message);
}
```

Using slide 9, refer to the code that shows the unit test class, named `HomeControllerTest`. It is the continuation of slide 8. The example in slide 8 is carried forward in slide 9.

## Unit Test Classes and Methods 4-4

### Snippet

```
[TestClass]
public void Contact()
{
    // Arrange
    HomeController controller = new HomeController();

    // Act
    ViewResult result = controller.Contact() as ViewResult;
    // Assert
    Assert.IsNotNull(result);
}
```

- ◆ This code uses the [TestClass] attribute in the class declaration and the [TestMethod] attributes for each action method that needs to be tested.

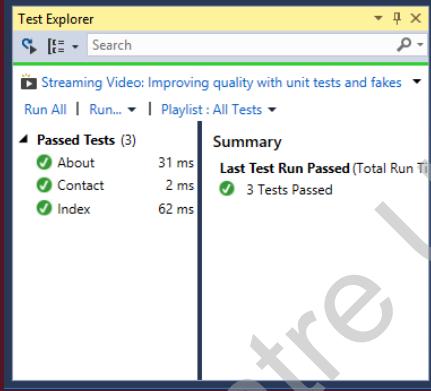
In slide 10, you continue to referring the code that shows the unit test class, named HomeControllerTest.

Then explain that this code uses the [TestClass] attribute in the class declaration and the [TestMethod] attributes for each action method that needs to be tested.

## Slide 11

Understand performing unit tests.

### Performing Unit Tests

- ◆ To unit test the Home controller class using the HomeControllerTest unit test, you need to select Test→Run→All Tests in Visual Studio 2013. The Test Explorer window displays the test results.
- ◆ Following figure shows the Test Explorer window:  


© Aptech Ltd. Testing and Deploying/Session 15 11

In slide 11, explain them that to unit test the Home controller class using the HomeControllerTest unit test, you need to select Test → Run → All Tests in Visual Studio 2013. The Test Explorer window displays the test results.

Then refer to the figure that shows the Test Explorer window.

**Slide 12**

Understand preparing the application for deployment.

### Preparing the Application for Deployment

- ◆ While developing and executing an ASP.NET MVC application using Visual Studio 2013, the application automatically deployed on Internet Information Server (IIS) Express.
- ◆ IIS Express makes deployment simple because it runs with your identity, and allows you to start and stop the Web server whenever required.
- ◆ To make your application accessible over the Internet, you need to host it on an IIS or any other Web server.
- ◆ Before deploying the application on a Web server, you first need to prepare the application for deployment.
- ◆ This process of preparing an application contains activities, such as identifying the files and folders to be copied on the Web server, configuring Web.config file, and precompiling the application.

In slide 12, explain the students that while developing and executing an ASP.NET MVC application using Visual Studio 2013, the application automatically deployed on Internet Information Server (IIS) Express. IIS Express makes deployment simple because it runs with their identity, and allows them to start and stop the Web server whenever required.

Tell them that to make an application accessible over the Internet; they need to host it on an IIS or any other Web server.

Then explain them that before deploying the application on a Web server, they first need to prepare the application for deployment. This process of preparing an application contains activities, such as identifying the files and folders to be copied on the Web server, configuring Web.config file, and precompiling the application.

**Slide 13**

Understand identifying the files and folders.

### Identifying the Files and Folders

- ◆ When you create an application using Visual Studio 2013, the application is saved on the path of the local computer that you can explicitly specify.
- ◆ Once you have created the application then, you need to deploy it on a Web server, such as IIS.
- ◆ To deploy the application on IIS, first you need to identify the files and folders that are created in the specified path and need to be copied to the destination server.
- ◆ While using Visual Studio 2013, by default it deploys only those files and folders that are required to run the application.
- ◆ However, sometimes there might be a requirement where you need to copy several other files and folders on the destination server.

Use slide 13 to explain that when they create an application using Visual Studio 2013, the application is saved on the path of the local computer that you can explicitly specify. Once they have created the application then, they need to deploy it on a Web server, such as IIS.

Tell them that to deploy the application on IIS, first they need to identify the files and folders that are created in the specified path and need to be copied to the destination server. While using Visual Studio 2013, by default it deploys only those files and folders that are required to run the application. However, sometimes there might be a requirement where they need to copy several other files and folders on the destination server.

## Slides 14 and 15

Understand configuring the Web.config file.

### Configuring the Web.config File 1-2

- ◆ Once you have deployed an ASP.NET MVC application on a Web server, some of the settings of the Web.config file vary in the deployed application.
- ◆ A transform file is associated with a build configuration.
- ◆ When you compile and execute an application in Visual Studio 2013, by default it creates the Debug and Release build configurations files named Web.Debug.config and Web.Release.config respectively.
- ◆ When you compile the application in Release mode, the Web.release.config file contains the changes that Visual Studio 2013 made in the Web.config file.
- ◆ On the other hand, when you compile the application in the Debug mode, the Web.debug.config file contains the changes that Visual Studio 2013 made in the Web.config file.

Display slide 14 and explain that once they have deployed an ASP.NET MVC application on a Web server, some of the settings of the Web.config file vary in the deployed application. A transform file is associated with a build configuration.

Tell them that when they compile and execute an application in Visual Studio 2013, by default it creates the Debug and Release build configurations files named Web.Debug.config and Web.Release.config, respectively. When they compile the application in Release mode, the Web.release.config file contains the changes that Visual Studio 2013 made in the Web.config file. On the other hand, when they compile the application in the Debug mode, the Web.debug.config file contains the changes that Visual Studio 2013 made in the Web.config file.

## Configuring the Web.config File 2-2

- ◆ To publish the application using release configuration, you need to remove the debug attribute from the `<compilation>` element in the Web.config file.
- ◆ Following code snippet shows how to remove the debug attribute:

Snippet

```
<system.web>
<compilation xdt:Transform="RemoveAttributes(debug)" />
</system.web>
```

- ◆ In this code, the `xdt:Transform` attribute is used to remove the debug attribute from the Web.config file.

In slide 15, explain them that to publish the application using release configuration, they need to remove the debug attribute from the `<compilation>` element in the Web.config file.

Next, refer to the code that shows how to remove the debug attribute.

Then explain the students that in this code, the `xdt : Transform` attribute is used to remove the debug attribute from the Web.config file.

**Slides 16 and 17**

Understand Precompilation.

## Precompilation 1-2

- ◆ To deploy an application, you need to copy the files and folders of the application to the hard drive of a Web server.
- ◆ In this process, most of the files are deployed to the Web server without compilation.
- ◆ Deploying an application in this way typically contains the following issues:
  - ◆ The application might get deployed with compilation errors
  - ◆ The source code of the application is exposed
  - ◆ The application loads slowly because files are required to be compiled when it is accessed for the first time

© Aptech Ltd.

Testing and Deploying/Session 15      16

In slide 16, explain the students that to deploy an application, they need to copy the files and folders of the application to the hard drive of a Web server. In this process, most of the files are deployed to the Web server without compilation.

Then explain them that deploying an application in this way typically contains the following issues:

- The application might get deployed with compilation errors
- The source code of the application is exposed
- The application loads slowly because files are required to be compiled when it is accessed for the first time

## Precompilation 2-2

- ◆ Precompiling a Web application is a process that involves compilation of the source code into DLL assemblies before deployment.
- ◆ The process of precompiling provides the following advantages:
  - ❖ It provides faster response because the files of the application do not need to be compiled the first time it is accessed.
  - ❖ It helps in identifying the errors that can occur when a page is requested, because errors are rectified at the time of compiling the application.
  - ❖ It secures the source code of the application from malicious users.

In slide 17, tell them that precompiling a Web application is a process that involves compilation of the source code into DLL assemblies before deployment.

Then explain them about the advantages of the process of precompiling as displayed on the slide 17.

**Slide 18**

Understand deploying on IIS.

### Deploying on IIS

- ◆ IIS is a Web server that allows you to develop, host, and manage your application.
- ◆ Before deploying an application on IIS, you first need to install it on your computer.
- ◆ To deploy an ASP.NET MVC application, you need to perform the following tasks:
  - ◆ Install IIS
  - ◆ Create an application in IIS
  - ◆ Create a publish profile for the application
  - ◆ Publish the project

© Aptech Ltd. Testing and Deploying/Session 15 18

In slide 18, explain that IIS is a Web server that allows developing, hosting, and managing applications. Before deploying an application on IIS, they first need to install it on the computer.

Then explain them about the tasks to be performed to deploy an ASP.NET MVC application as displayed on the slide 18.

**Additional Information:**

ASP.NET provides you with the tracing feature that enables you to track the program execution, thereby ensuring that your Web application runs properly. You can use this feature to view diagnostic information about a particular Web page. This information includes the execution time of page methods, the time spent rendering page controls, and the contents of various collections, such as the query string, HTTP header, and session state.

**Slides 19 and 20**

Understand installing IIS.

### Installing IIS 1-2

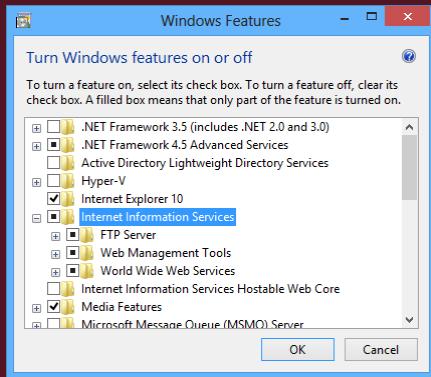
- ◆ The first step to deploy an application is to install IIS.
- ◆ To install IIS, you need to perform the following tasks:
  - ◆ Open Control Panel.
  - ◆ Click Uninstall a program under the Programs icon. The Program and Features window displays the Uninstall or change a program screen.
  - ◆ Click the Turn Windows features On or Off link in the left pane. The Windows Features window is displayed.
  - ◆ Ensure that all the check boxes are selected under the Internet Information Services node.

© Aptech Ltd. Testing and Deploying/Session 15 19

In slide 19, explain to the students that the first step to deploy an application is to install IIS. Then explain the steps to install IIS as displayed on the slide 19.

## Installing IIS 2-2

- ◆ Following figure shows the Windows Features window:



- ◆ Click OK. The Windows Features message box is displayed.
- ◆ Wait until the changes are applied to the system.
- ◆ Click the Close button.
- ◆ Close the Programs and Features window.

In slide 20, you keep explaining the steps to install IIS, as displayed on the slide. You can then refer to the figure that shows the Windows Features window.

**Slides 21 to 24**

Understand creating an application on IIS.

**Creating an Application on IIS 1-4**

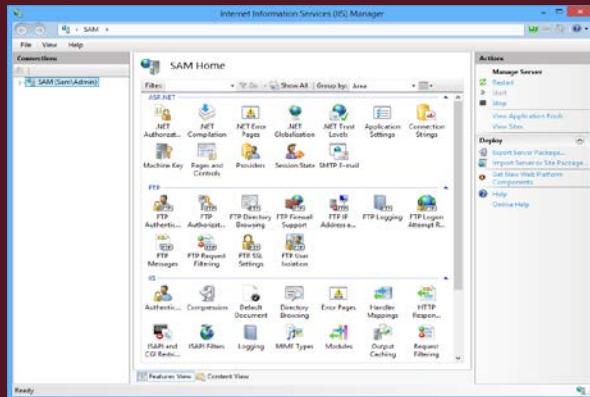
- ◆ Once you have installed IIS, the next step that you need to perform is creating a publish profile.
- ◆ To create an application on IIS, you need to perform the following steps:
  - ◆ Create a folder with the name of your project, for example MVCDemo in the C:\inetpub\wwwroot folder.
  - ◆ Press the Windows+R keys. The Run dialog box is displayed.
  - ◆ Type inetmgr in the Run dialog box.
  - ◆ Click OK. The Internet Information Services (IIS) Manager window is displayed.

In slide 21, explain the students that once they have installed IIS, the next step that they need to perform is creating a publish profile.

Then you explain the steps to create an application on IIS, as displayed on the slide 21.

## Creating an Application on IIS 2-4

- ◆ Following figure shows the Internet Information Services (IIS) Manager window:

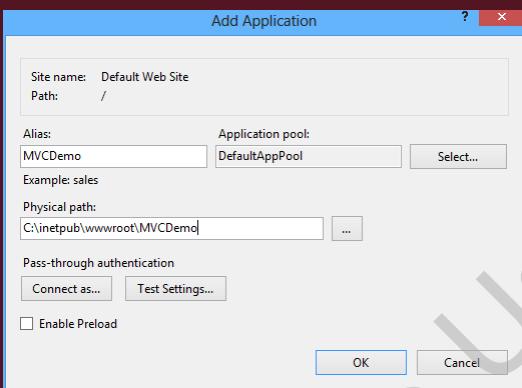


- ◆ Expand the node under the Connections pane.
- ◆ Expand the Sites node.

In slide 22, refer to the figure that shows the Internet Information Services (IIS) Manager window.

### Creating an Application on IIS 3-4

- ❖ Right-click the Default Website node under the Sites node, and then, select Add Application. The Add Application dialog box is displayed.
- ❖ In the Add Application dialog box, type MVCDemo in the Alias text field and type C:\inetpub\wwwroot\MVCDemo in the Physical path text field.
- ◆ Following figure shows the Add Application dialog box:



© Aptech Ltd.

Testing and Deploying/Session 15

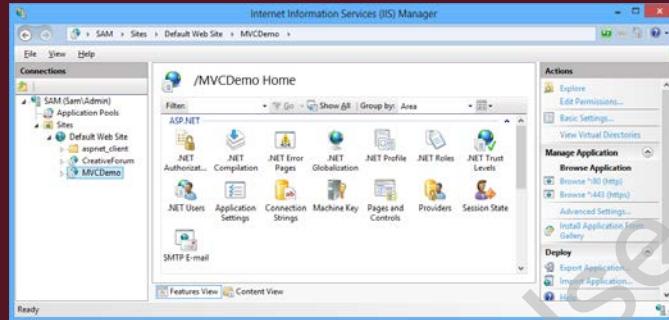
23

In slide 23, continue explaining the steps to create an application on IIS, as displayed on the slide.

You can refer to the figure that shows the Add Application dialog box.

### Creating an Application on IIS 4-4

- ❖ Click the OK button in the Add Application dialog box. The MVC node is displayed under the Connection node of the Internet Information Services (IIS) Manager window.
- ◆ Following figure shows the MVC node under the Connection node:



- ❖ Close the Internet Information Services (IIS) Manager window.

In slide 24, refer to the figure that shows the MVC node under the Connection node.

**Slides 25 to 29**

Understand creating a publish profile.

**Creating a Publish Profile 1-5**

- ◆ After creating the application on IIS, you need to create a publish profile.
- ◆ A publish profile represents various deployment options, such as the target server to be used for deployment, the credentials needed to log on to the server to deploy.
- ◆ To create a publish profile in Visual Studio 2013, you need to perform the following tasks:
  - ◆ Start Visual Studio 2013 with Administrator privilege.
  - ◆ Open the MVCDemo project to publish.
  - ◆ Right-click the project in the Solution Explorer window and select Publish. The Publish Web dialog box is displayed.
  - ◆ From the Select or import a publish profile drop-down list, select the <New...> option.
  - ◆ The New Profile dialog box is displayed.
  - ◆ Type MVCDemoPublishProfile in the Profile Name text field.

© Aptech Ltd. Testing and Deploying/Session 15 25

In slide 25, explain that after creating the application on IIS, they need to create a publish profile. A publish profile represents various deployment options, such as the target server to be used for deployment, the credentials needed to log on to the server to deploy.

Then you explain the steps to create a publish profile in Visual Studio 2013, as displayed on the slide 25.

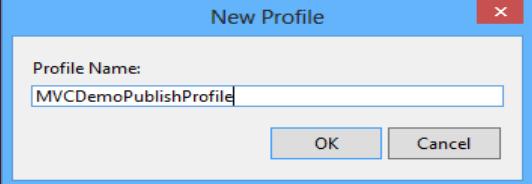
**Additional Information:**

If you are deploying a Web application to a target server owned by a service provider, the service provider will provide you a .publish settings file. You can automatically create the publish profile by importing this .publish settings file while using the Publish Web wizard.

For Aptech Centre Use Only

### Creating a Publish Profile 2-5

◆ Following figure shows the New Profile dialog box:



- ◆ Click OK. The Publish Web dialog box is displayed with the specified profile name.
- ◆ Ensure that Web Deploy is selected in the Publish method drop-down list.
- ◆ In the Service URL text field, type localhost and in the Site/application text field, type Default Web Site/MVCDEMOPUBLISHPROFILE.

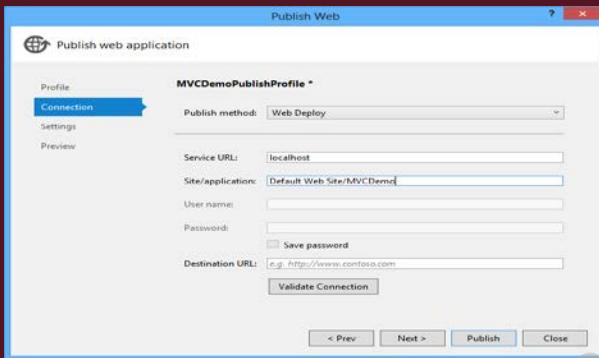
© Aptech Ltd. Testing and Deploying/Session 15 26

Use slide 26, to refer to the figure that shows the New Profile dialog box.

Then continue explaining the steps to create a publish profile in Visual Studio 2013, as displayed on the slide 26.

### Creating a Publish Profile 3-5

- Following figure shows the specifying the Service URL and Site/application text fields:



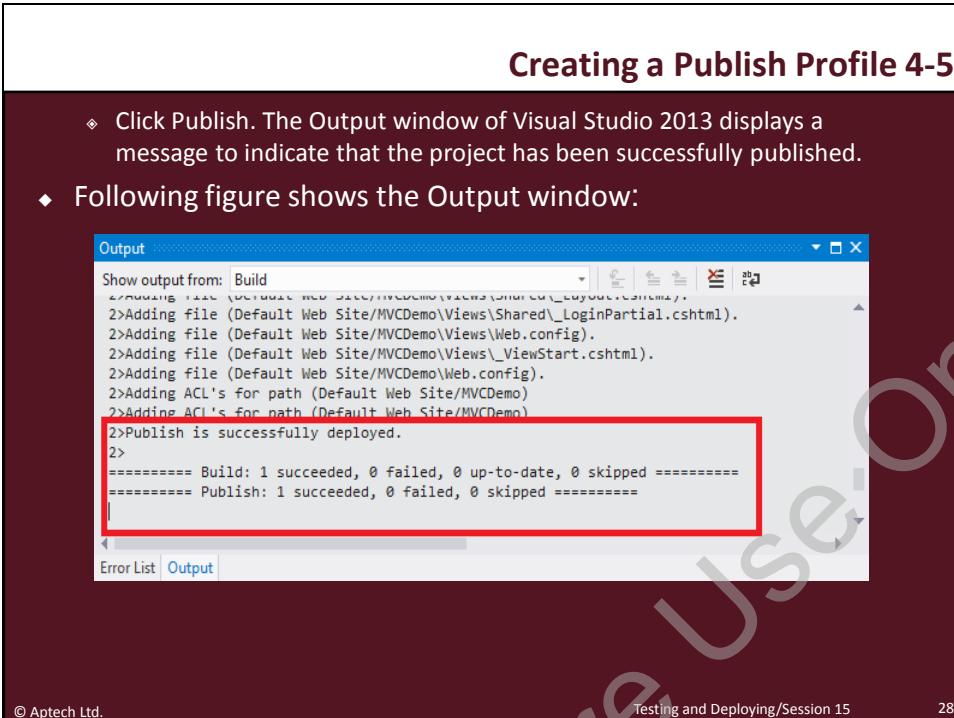
- Click Validate Connection. When the connection is valid a Correct mark is displayed by the side of the Validate Connection button.
- Click Next. The Publish Web dialog box is displayed.

Use slide 27, to refer to the figure that shows specifying the Service URL and Site/application text fields.

Then continue explaining the steps to create a publish profile in Visual Studio 2013, as displayed on the slide 27.

### Creating a Publish Profile 4-5

- ❖ Click Publish. The Output window of Visual Studio 2013 displays a message to indicate that the project has been successfully published.
- ◆ Following figure shows the Output window:



The screenshot shows the Visual Studio 2013 Output window with the title 'Output' and 'Show output from: Build'. The window displays the following log entries:

```
2>Moving file 'C:\Users\Aptech\Documents\Visual Studio 2013\Projects\MVCDemo\Views\Shared\_LoginPartial.cshtml' to 'C:\Users\Aptech\Documents\Visual Studio 2013\Projects\MVCDemo\bin\MVCDemo\Views\Shared\_LoginPartial.cshtml'.
2>Adding file (Default Web Site/MVCDemo\Views\Shared\_LoginPartial.cshtml).
2>Adding file (Default Web Site/MVCDemo\Views\Web.config).
2>Adding file (Default Web Site/MVCDemo\Views\_ViewStart.cshtml).
2>Adding file (Default Web Site/MVCDemo\Web.config).
2>Adding ACL's for path (Default Web Site/MVCDemo)
2>Adding ACL's for path (Default Web Site/MVCDemo)
2>Publish is successfully deployed.
2>
===== Build: 1 succeeded, 0 failed, 0 up-to-date, 0 skipped =====
===== Publish: 1 succeeded, 0 failed, 0 skipped =====
```

The last two lines of the log, which indicate the successful deployment, are highlighted with a red rectangular box.

© Aptech Ltd.

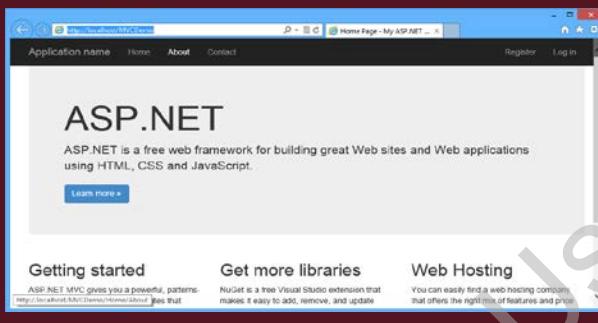
Testing and Deploying/Session 15

28

In slide 28, you refer to the figure that shows the Output window.

### Creating a Publish Profile 5-5

- ◆ To test the published application type the following URL in the address bar of the browser.  
`http://localhost/MVCDemo`
- ◆ The Home page of the application published on IIS is displayed on the browser.
- ◆ Following figure shows the Home page of the published application on the browser:



In slide 29, tell the students that to test the published application, type the following URL in the Address bar of the browser:

`http://localhost/MVCDemo`

Then tell them that the Home page of the application published on IIS is displayed on the browser.

Then refer to the figure that shows the Home page of the published application on the browser.

## Slide 30

Let us summarize the session.

**Summary**

- ◆ Unit testing is a technique that allows you to create classes and methods in your application and test their intended functionality.
- ◆ While developing and executing an ASP.NET MVC application using Visual Studio 2013, the application is automatically deployed on IIS Express.
- ◆ To deploy the application on IIS, first you need to identify the files and folders that are required to be copied on the destination server.
- ◆ While using Visual Studio 2013, by default it deploys only those files and folders that are required to run the application.
- ◆ Precompiling a Web application is a process that involves compilation of the source code into DLL assemblies before deployment.
- ◆ To deploy an ASP.NET MVC application, you need to install IIS, create an application in IIS, create a publish profile for the application, and finally publish the project.
- ◆ A publish profile represents various deployment options, such as the target server to be used for deployment, the credentials needed to log on to the server to deploy.

© Aptech Ltd. Testing and Deploying/Session 15 30

In slide 30, you will summarize the session. You will end the session, with a brief summary of what has been taught in the session. Tell the students pointers of the session. This will be a revision of the current session and it will be related to the next session. Explain each of the following points in brief. Tell them that:

- Unit testing is a technique that allows you to create classes and methods in your application and test their intended functionality.
- While developing and executing an ASP.NET MVC application using Visual Studio 2013, the application is automatically deployed on IIS Express.
- To deploy the application on IIS, first you need to identify the files and folders that are required to be copied on the destination server.
- While using Visual Studio 2013, by default it deploys only those files and folders that are required to run the application.
- Precompiling a Web application is a process that involves compilation of the source code into DLL assemblies before deployment.
- To deploy an ASP.NET MVC application, you need to install IIS, create an application in IIS, create a publish profile for the application, and finally publish the project.
- A publish profile represents various deployment options, such as the target server to be used for deployment, the credentials needed to log on to the server to deploy.

### **15.3 Post Class Activities for Faculty**

You should familiarize yourself with the topics of the next session. You should also explore and identify the **OnlineVarsity** accessories and components that are offered for the next session.

**Tips:** You can also check the **Articles/Blogs/Expert Videos** uploaded on the **OnlineVarsity** site to gain additional information related to the topics covered in the next session. You can also connect to online tutors on the **OnlineVarsity** site to ask queries related to the sessions. You can refer any of the related books to provide much more information about the topic. You may analyze the students understanding capability towards the subject by giving them some live task related to the session concepts.

You can also put a few questions to students to search additional information, such as:

1. What is the purpose of tracing feature?
2. How will you create a publish profile when you need to deploy a Web application to a target server owned by a service provider?