- UNIT II
  - The razor view engine
  - what is razor?
  - Razor syntax
  - Strongly typed views,

Model Introduction

Model represents domain specific data and business logic in MVC architecture. It maintains the data of the application.

Model objects retrieve and store model state in the persistence store like a database.

Model class holds data in public properties. All the Model classes reside in the Model folder in MVC folder structure.

Example: Model class

```
namespace MVC_BasicTutorials.Models
{
    public class Student
    {
        public int StudentId { get; set; }
        public string StudentName { get; set;  }
        public int Age { get; set;  }
    }
}
```

Razor Introduction

Razor is a syntax used in Microsoft .NET technologies to develop web pages and applications that let us embed the server based code (like C# and Visual Basic) into the web pages.

All the server side code in Razor starts with @{ and ends with }.

We can mix the html and server based code inside the Razor block and that is the beauty of the Razor markup.

Declare a variable in Razor View

To declare a variable in the View using Razor syntax, we need to first create a code block by using @{ and } and then we can use the same syntax we use in the C#.

```
@{
  var fileName = "dotnetfunda";
  var a = 20;
  var b = 30;
}
```

Comment in Razor View

To comment in the code block of Razor view, we use the same syntax as we use in C#.

Like for single line // and for multiline /* and */.

To comment, HTML along with other code, we need to use razor comment block that starts with @* and ends with *@.

For Loop in Razor View

following represents syntax of for loop in asp.net mvc razor view engine

```
@for (int i = 0; i < 5; i++)
{
    //Code Block
}
```

Foreach Loop in Razor View

Following is the syntax of using foreach loop in asp.net mvc razor view

```
@ foreach (var item in marks)
{
        //Code Block
 }
```

If condition in Razor view

```
@if (condition)
{
        //statements
}
else
{
        //statements
}
```

Switch case in Razor View

```
@switch (variable)
{
        case constant:
                //statements
                 break;
        default :
                //statments
                 break;
 }
```

Variable data inside URL

Example

HTML Helper Classes

HTML helpers are the way by which we can render html on the view page of MVC.

These Helpers are simple functions that let the developer specify the type of HTML needed on the view.

HTML Helper can be **loosely typed view** and **strongly typed view**.

HtmlHelper - TextBox and TextBoxFor

HtmlHelper class includes two extension methods which creates a textbox (<input type="text">) element in razor view: TextBox() and TextBoxFor().

The TextBox() method is loosely typed method whereas TextBoxFor() is a strongly typed method.

Syntax

Html.TextBox(string name, string value, object htmlAttributes)

Syntax

Html.TextBoxFor(lambda expression, object htmlAttributes)

Other Controls of Helper

- TextArea(string name, string value, object htmlAttributes)
- TextAreaFor(lambda expression, object htmlAttributes)

- RadioButton(string name, object value, bool isChecked, object htmlAttributes)
- RadioButtonFor(expression, object value, object htmlAttributes)

Other Controls of Helper

- CheckBox(string name, bool isChecked, object htmlAttributes)
- CheckBoxFor(lambda expression, object htmlAttributes)

Other Controls of Helper

- DropDownList(string name, IEnumerable<SelectLestItem> selectList, string optionLabel, object htmlAttributes)
- DropDownListFor(expression, IEnumerable<SelectLestItem> selectList, string optionLabel, object htmlAttributes)

Other Controls of Helper

- Hidden(string name, object value, object htmlAttributes)
- HiddenFor(Expression)
- Password(string name, object value, object htmlAttributes)
- PasswordFor(Expression, object htmlAttributes)

Other Controls of Helper

- Display(string expression)
- DisplayFor(expression)
- Label(string expression, string labelText, object htmlAttributes)
- LabelFor(expression)

Editor

- We have seen different HtmlHelper methods used to generated different html elements in the previous sections.
- ASP.NET MVC also includes a method that generates html input elements based on the datatype.
- EditorFor() extension method generates html elements based on the data type of the model object's property.

Property Data Type	Html Element
string	<input type="text" >
Int	<input type="number" >
decimal, float	<input type="text" >
Boolean	<input type="checkbox" >
Enum	<input type="text" >
DateTime	<input type="datetime" >

References

Web Site:
https://docs.microsoft.com/en-us/aspnet/mvc/
https://www.tutorialspoint.com/mvc_framework/mvc_framework_introduction.htm
https://www.guru99.com/mvc-tutorial.html
https://en.wikipedia.org/wiki/ModelViewController
https://www.guru99.com/mvc-tutorial.html
https://www.geeksforgeeks.org/mvc-design-pattern/

Book:
Pro ASP.NET MVC 5.0