

QUEZON CITY UNIVERSITY

COLLEGE OF COMPUTER STUDIES



WEEK 5

DEVELOPING VIEWS

IPT102 – INTEGRATIVE PROGRAMMING AND TECHNOLOGIES 2

1ST SEMESTER

LEARNING OUTCOMES:

- To create views with Razor Syntax
- To declare the variable in Views using Razor Syntax

Main Razor Syntax Rules for C#

- Razor code blocks are enclosed in `@{ ... }`
- Inline expressions (variables and functions) start with `@`
- Code statements end with semicolon
- Variables are declared with the `var` keyword
- Strings are enclosed with quotation marks
- C# code is case sensitive
- C# files have the extension `.cshtml`

How it works:

Razor web pages can be described as HTML pages with two kinds of content: HTML content and Razor code.

When the server reads the page, it runs the Razor code first, before it sends the HTML page to the browser. The code that is executed on the server can perform tasks that cannot be done in the browser, for example accessing a server database. Server code can create dynamic HTML content on the fly, before it is sent to the browser. Seen from the browser, the HTML generated by server code is no different than static HTML content.

Syntax:

```
<h1>Razor syntax demo</h1>
```

```
<h2>@DateTime.Now.ToShortDateString()</h2>
```

Output:

Razor syntax demo
“ Current date “

Multi-statement Code block:

You can write multiple lines of server-side code enclosed in braces `@{ ... }`. Each line must end with a semicolon the same as C#.

Example: Server side Code in Razor

Syntax

```
@{  
    var date =  
    DateTime.Now.ToShortDateString();  
    var message = "Hello World";  
}
```

```
<h2>Today's date is: @date </h2>  
<h3> @message</h3>
```

Output:

Today's date is: 08-09-2014
Hello World!

Declare Variables

Declare a variable in a code block enclosed in brackets and then use those variables inside HTML with @ symbol.

Example: Text in Razor Syntax

// Using the var keyword:

```
var greeting = "Welcome to my Class";
```

```
var counter = "Number of Students:" + 50;
```

```
var today = DateTime.Today;
```

// Using data types:

```
string greeting = "Welcome to my Class";
```

```
int counter = 50;
```

```
var count = "Students:" + counter;
```

```
DateTime today = DateTime.Today;
```

Output:

Welcome to my Class

Number of Students:50

9/7/2021 12:00:00 AM

If and Else Conditions

The common way to do this is with the if ... else statements:

Example:

```
@{
var txt = "";
if(DateTime.Now.Hour > 12)
    {txt = "Good Evening";}
else
    {txt = "Good Morning";}
}
<html>
<body>
<p>The message is @txt</p>
</body>
</html>
```

Output:

The message is Good Evening

For Loops

This kind of loop is especially useful for counting up or counting down:

Example:

```
<ul>  
  @for (int i = 0; i < 5; i++)  
  {  
    <li>@i</li>  
  }  
</ul>
```

Output:

```
0  
1  
2  
3  
4
```

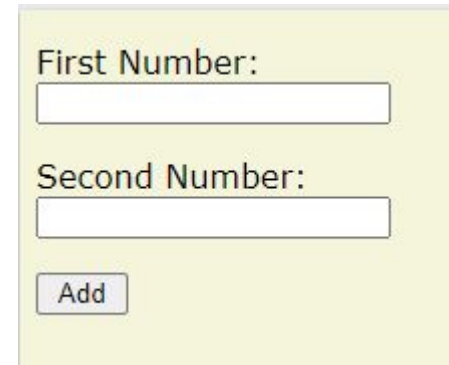
Reading User Input

Input is read by the `Request[]` function, and posting (input) is tested by the `IsPost` condition:

Example:

```
@{
var totalMessage = "";
if(IsPost)
{
var num1 = Request["text1"];
var num2 = Request["text2"];
var total = num1.AsInt() + num2.AsInt();
totalMessage = "Total = " + total;
}
}
```

Output:

A screenshot of a web form with a light yellow background. It contains two text input fields. The first field is labeled "First Number:" and the second field is labeled "Second Number:". Below the second field is a button labeled "Add".

END OF PRESENTATION.
THANK YOU!