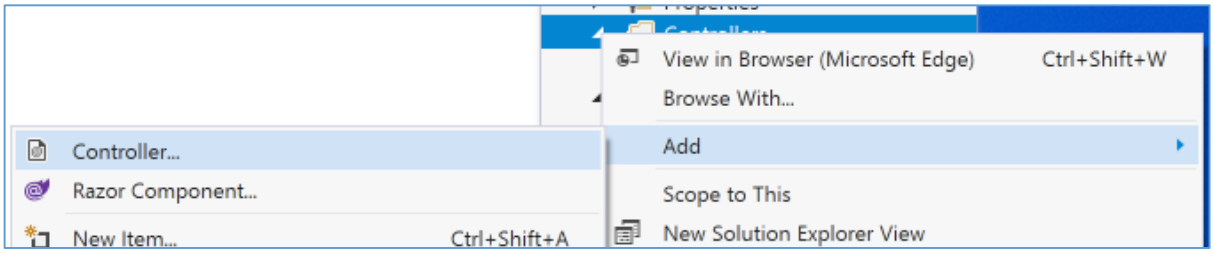
Web Development .NET
Passing Data From The Controller To The View (ViewData / ViewBag)

Let's return to the MVC application (MVCPets) that we created earlier.
We did create a controller, a model and a view, but we are going to create some more now and look at other ways in which we can pass data from a controller to a view.

From the Controllers folder
Add → Controller → MVC Controller (Empty)




```
public class HelloWorldController : Controller
{
    // GET: /HelloWorld
    public IActionResult Index()
    {
        return View();
    }

    // GET: /HelloWord/Hello/
    public string Hello()
    {
        return "This is the Hello method";
    }
}
```

We have an Index method that returns a view. But we have not created this view yet.

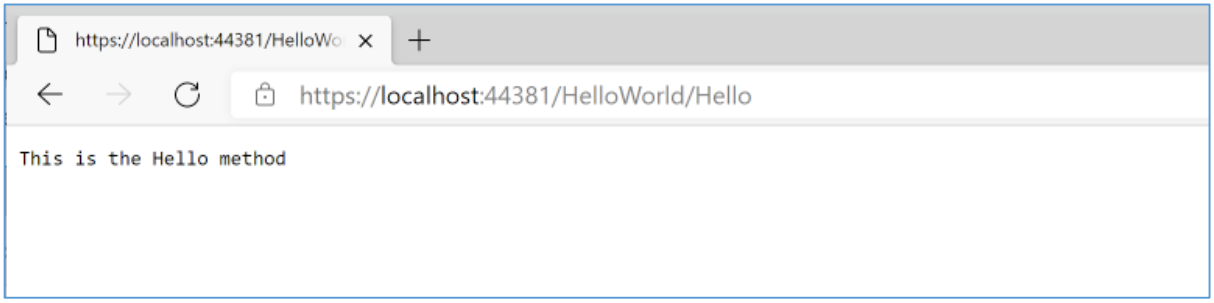
Ignore that method for now…

We are going to create a new method that returns a string. This is because we are going to learn how to pass data in the URL to a controller.

Create the Hello() method as shown so it returns a string.

HelloWorldController.cs

Run the application and browse to the following /HelloWorld/Hello so that you see the string.



The Controller is **HelloWorld** and **Hello** is the action method. Now we are going to add some parameters (in the URL).

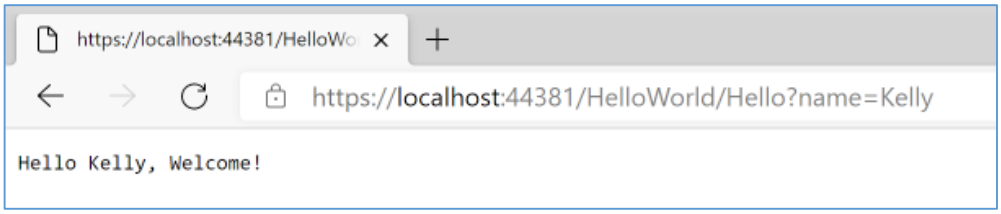
Change the Hello method to accept a string parameter and uses it in the return string.

You will need to add the using statement using System.Text.Encodings.Web; in order to use the Encode method. We need this to encode the string being returned into suitable HTML format (in case it contains any special characters, e.g. < or >)

```
using System.Text.Encodings.Web;
```

```
// GET: /HelloWord/Hello/
public string Hello(string name)
{
    return HtmlEncoder.Default.Encode($"Hello {name}, Welcome!");
}
```

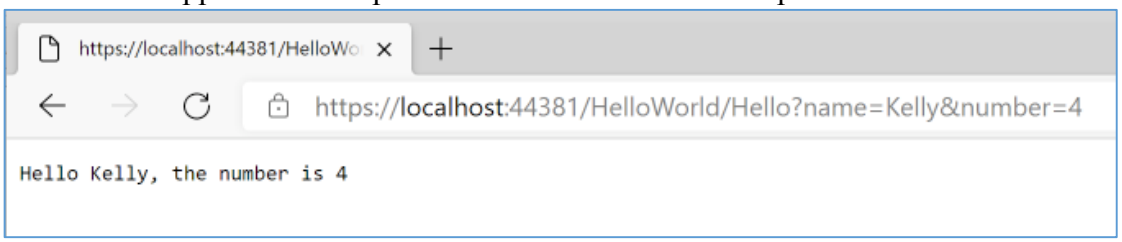
Now run the application and pass the “name” parameter in the URL.



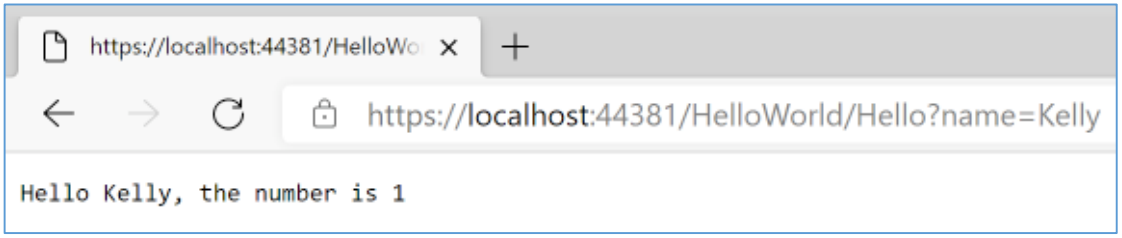
We can pass multiple parameters (and we can have a default for optional parameters). Change the Hello() method as shown,

```
// GET: /HelloWord/Hello/
public string Hello(string name, int number=1)
{
    return HtmlEncoder.Default.Encode($"Hello {name}, the number is {number}");
}
```

Now run the application and pass the “name” and “number” parameters in the URL.



If you miss out the number parameter, the default kicks in.



The MVC model binding system is mapping the named parameters in the query string (URL) to parameters in the method.

Passing Data From Controller To View:
Using ViewData

The example we just looked at uses the Hello method to take in two parameters and output the result directly to the browser. Of course, to do this properly we should have the controller pass the data to the View instead, and the View should be creating the response that the user sees.

So we will get the Controller to store the data (the parameters) in a ViewData dictionary so that the View can access this data and use it.

Change the Hello method as shown:

```
// GET: /HelloWord/Hello/
public IActionResult Hello(string name, int number=1)
{
    // Store the parameters in a ViewData dictionary
    ViewData["name"] = name;
    ViewData["number"] = number;
    // Return the View
    return View();
}
```

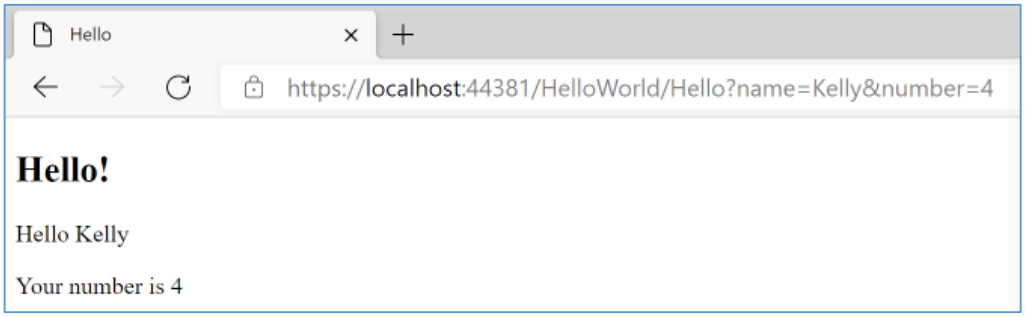
HelloWorldController.cs

Now add a HelloWorld folder inside Views, and create a new Razor View called Hello. Inside this View, you can add code that embeds the name and number into a HTML page.

```
@{
    Layout = null;
}

<!DOCTYPE html>

<html>
<head>
    <meta name="viewport" content="width=device-width" />
    <title>Hello</title>
</head>
<body>
    <h2>Hello!</h2>

    <p>Hello @ViewData["name"]</p>
    <p>Your number is  @ViewData["number"]</p>
</body>
</html>
```

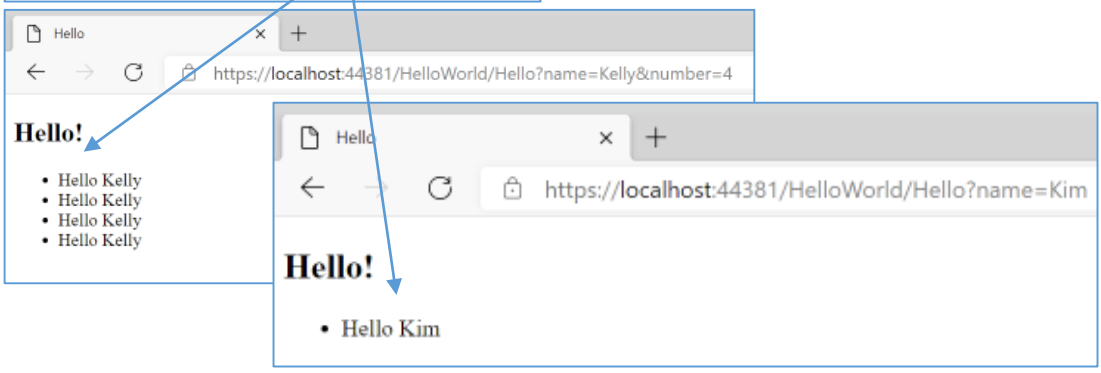
Hello.cshtml

Now run the application and pass the “name” and “number” parameters in the URL.



We can use the parameters anywhere in our C# code.
e.g. Use the number in a loop that determines how many times we say hello to the user.

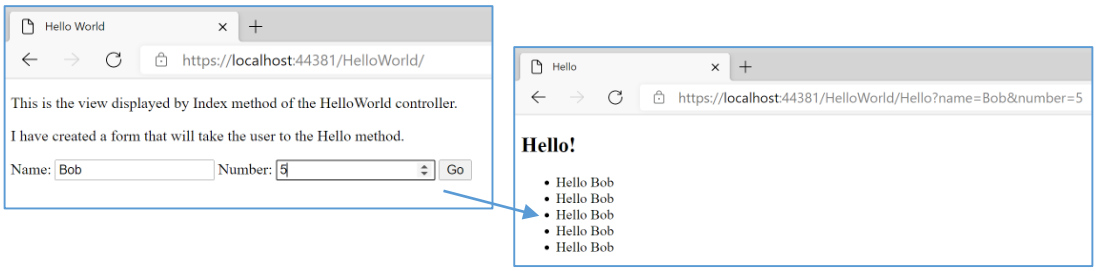
```
<ul>
    @for (int i = 0; i < (int)ViewData["number"]; i++)
    {
        <li>Hello @ViewData["name"]</li>
    }
</ul>
```



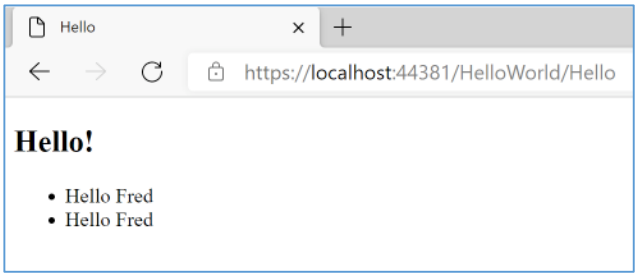
We can create a View for the Index method in the HelloWorld controller and use a form for the user to enter data (e.g. name and number). This data is passed using ViewData to the View.

```
<p>This is the view displayed by Index method of the HelloWorld controller.</p>
<p>I have created a form that will take the user to the Hello method.</p>
<form method="get">
    Name: <input type="text" id="name" name="name">
    Number: <input type="number" id="number" name="number">
    <button formaction="/HelloWorld/Hello">Go</button>
</form>
```

Index.cshtml (inside HelloWorld)



This will work if the form method is set to “get” or “post”.
But with “post” we do not see the parameters in the URL.



In this example I changed the method to “post” and I entered the values “Fred” and 2. The controller still passed the data correctly to the view using ViewData, but the only difference is we cannot see it in the URL on the browser window.

ViewData is a weakly-typed dictionary which uses the string balue as the key to retrieve data. It dynamically resolves at runtime so does not give us IntelliSense or compile-time checking.

Passing Data From Controller To View:
Using ViewBag

ViewBag is a wrapper around ViewData.
It uses dynamic properties which are resolved at run-time.
Just like ViewData, it does not give us IntelliSense or compile-time checking.

The same example with the form using ViewBag instead of ViewData is as follows:

```
// GET: /HelloWord/Hello/
public IActionResult Hello(string name, int number=1)
{
    // Store the parameters in ViewBag properties
    ViewBag.Name = name;
    ViewBag.Number = number;
    // Return the View
    return View();
}
```

HelloWorldController.cs

```
<ul>
    @for (int i = 0; i < (int)ViewBag.Number; i++)
    {
        <li>Hello @ViewBag.Name</li>
    }
</ul>
```

Hello.cshtml

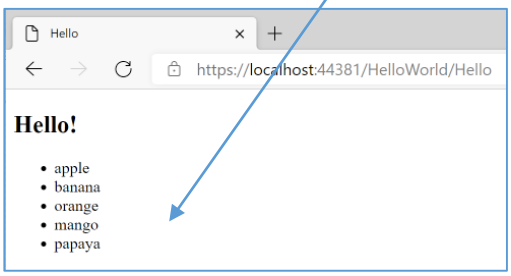
We can also pass lists inside ViewBag.

```
// GET: /HelloWord/Hello/
public IActionResult Hello()
{
    // Create a list of fruits
    var fruits = new List<string>
    {
        "apple",
        "banana",
        "orange",
        "mango",
        "papaya"
    };

    ViewBag.Fruits = fruits;

    // Return the View
    return View();
}
```

HelloWorldController.cs

```
<ul>
    @foreach (var fruit in ViewBag.Fruits)
    {
        <li>@fruit</li>
    }
</ul>
```

Hello.cshtml



Next Steps:

- Understand HTMLEncoder https://www.w3schools.com/asp/met_htmlencode.asp
- Understanding C# dictionaries (because ViewData is a dictionary).