This worksheet is going to look at model binding and tag helpers. The aim of today is to play with the different options available when using tag helpers to try and understand what they can do.

There are a number of questions asked with each of the tasks. Try and answer these on this page.

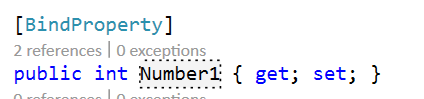
We will discuss some of the answers.

If you notice anything interesting or unusual please take note of it and we will discuss it and try and understand what is happening.

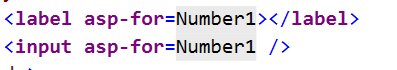
**Task 1**

Open your existing .Net core web application. Create a new razor page with a page model call it taghelpplay.

Add a property to your model called number1.



In your cshtml page add the following within a html form with a submit button.



**Note**: If you don’t get a colour change for the HTML label and input element tags you need to make sure your page is using tag helpers.

What happens when you submit the form?

Number 1 becomes the value from the form

**Note**: if the number submitted is replaced by a zero remember that by default model binding only works with requests using HTTP POST.

Look at the html which is generated can you understand where each of the attributes comes from.

Add some content to the label element “Enter your first number”.

Look at the html which is generated, what is the difference?

The label changed

Comment out the [BindProperty] decoration of the Number1 property.

What difference does this make?

Value is not allocated anymore to Number 1 – the property is not linked

**Task 2**

Add some more properties to your razor page. Add the [BindProperty] decoration to each of them.

Int?

Double

Double?

Decimal

Decimal?

Float

Float?

Add labels and input elements to your razor page for each of these and see what html is generated when you request the page.

Look at the html which is generated for each of these. What type do they have? Do they have a default value?

|  |  |  |
| --- | --- | --- |
| C# Type | HTML input element type | Default Value |
| Int |  | 0 |
| Int? |  | Null |
| Double |  | 0 |
| Double? |  | Null |
| Decimal |  | 0.00 |
| Decimal? |  | Null |
| Float |  | 0 |
| Float? |  | Null |

**Task 3**

Add a property with type string to the Model and suitable markup to the razor page.

What happens when you run the page?

String value is empty

Why do you think there is no default value generated for the input element when bound to a string?

It is an empty string

**Task 4**

Add a property with type **DATETIME** to the Model and suitable markup to the razor page.

What happens when you run the page?

**localhost** is currently unable to handle this request.

HTTP ERROR 500

What happens if you use type **DATETIME?** Instead?

Note: Dates can get quite complicated it is an area we will return to. In the meantime you can investigate different methods yourself.

**localhost** is currently unable to handle this request.

HTTP ERROR 500

**Task 5**

Add a property with type bool called IsContactAllowed and suitable markup to the razor page.

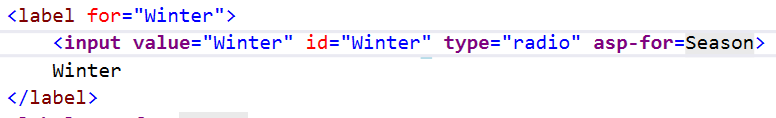
What does it produce?

**localhost** is currently unable to handle this request.

HTTP ERROR 500

**Task 6: Radio Buttons**

Radio buttons are more complicated than other input elements. You need to include more attributes manually on the razor page. See the example below. In this case you need a property Season declared in the PageModel page.



Add a set of radio button with a choice of seasons and echo the user’s choice back to them in a message. When testing this make sure.

* The data persisits. i.e. we want the radio button to stay selected.
* You can click on the label and have the appropriate button selected. It gives the user a much bigger click target.
* The radio buttons must act like a set. You should only be able to select one.