CSY1018
Web Programming
Topic 8

Tom Butler
thomas.butler@northampton.ac.uk

Topic 8

- User input using HTML forms

HTML Forms

- HTML provides a method of allowing user input:
  - Forms
- HTML forms have the following interactive elements:
  - Text (single line)
  - Text area (multi line)
  - Checkbox
  - Drop down (select)
  - Radio button
  - File upload
  - Buttons

HTML Forms

- All form elements must be placed inside a <form> tag
- The form tag has two required attributes:
  - *Method* -This is either GET or POST
  - *Action* – URL to which the form is *submitted*

HTML Forms

- Each element in a form has a *name* attribute
- This name is a unique identifier which can be used when the form has been *submitted*
- Each element should have a unique name

Form actions

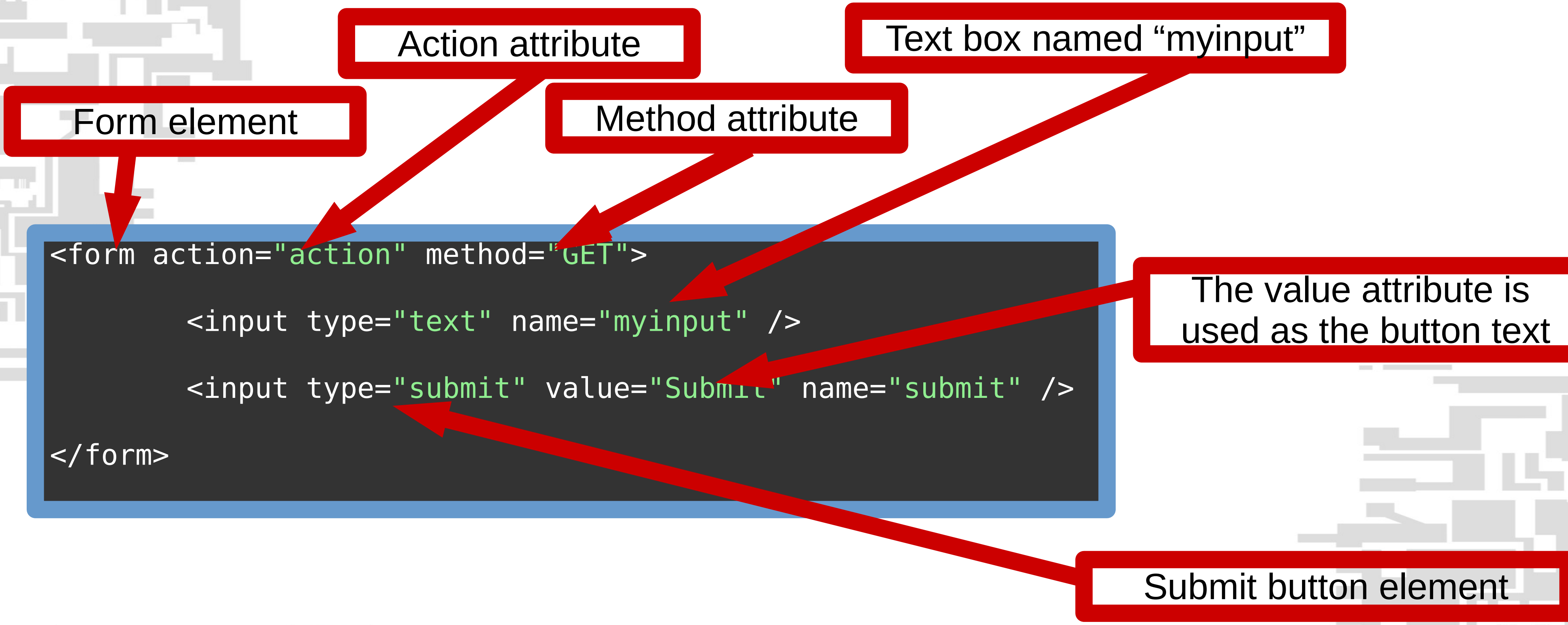
- Each form has an “action”
- This is what happens when the form is submitted
- There are a couple of different things that can go there
- Normally this is the URL of a page, on the server, which will handle the submission
  - e.g. posting a comment, adding a news article, uploading a file
- This requires advanced code which is covered in year 2
- For this module we will just design the form to look nice

Basic HTML form

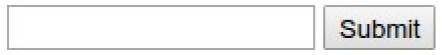
- A form needs at least one *input element* (e.g. a text box)
- Each form needs a *submit button*

```
<form action="action" method="GET">

        <input type="text" name="myinput" />

        <input type="submit" value="Submit" name="submit" />

</form>
```

Basic HTML form



```
<form action="action" method="GET">

        <input type="text" name="myinput" />

        <input type="submit" value="Submit" name="submit" />

</form>
```

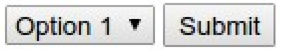


Form input types

HTML Forms – Select Boxes

- The HTML <select> tag enforces selection of a specific value
- The user cannot enter *free text*. They are forced to choose between one of the supplied *options*
- You must specify the available options inside the HTML
- Each option has a *value* which is sent back to the server and a *label* which is displayed to the user
  - The user never sees the *value*
  - The *label* is never sent to the server

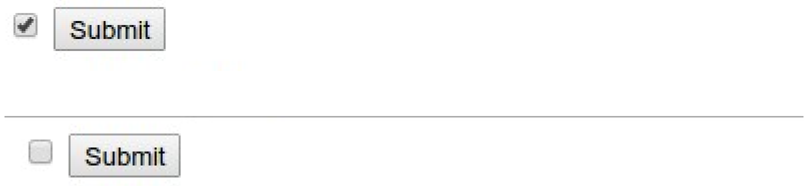
```html
<form action="action" method="POST">

        <select name="myselect">
                <option value="One">Option 1</option>
                <option value="Two">Option 2</option>
                <option value="Three">Option 3</option>
        </select>
        <input type="submit" value="Submit" name="submit" />

</form>
```



Checkboxes

- A checkbox is an input element that allows the user to *tick* a box. This represents a *boolean* on a HTML form, the user can choose to tick the box or not
- This is quite often used for “Do you accept the terms and conditions” or questionnaires

```
<form action="action" method="POST">

        <input type="checkbox" name="mycheckbox" value="ticked" />
        <input type="submit" value="Submit" name="submit" />

</form>
```

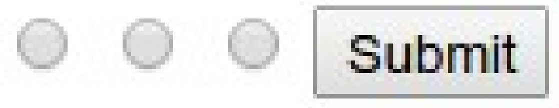

☑ Submit
☐ Submit

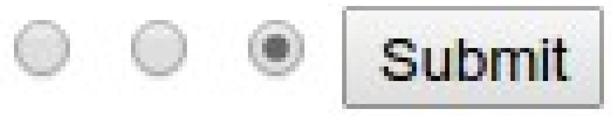
Radio Buttons

- Radio buttons are similar to checkboxes
- You can *tick* and *untick* them
- However, only one radio button can be ticked at the same time
- You must create multiple radio buttons with the same *name.*
- Only one radio button with that *name* can be ticked at any time

Radio Buttons

```
<form action="action" method="POST">

        <input type="radio" name="myradio" value="radio button 1" />
        <input type="radio" name="myradio" value="radio button 2" />
        <input type="radio" name="myradio" value="radio button 3" />

        <input type="submit" value="Submit" name="submit" />

</form>
```

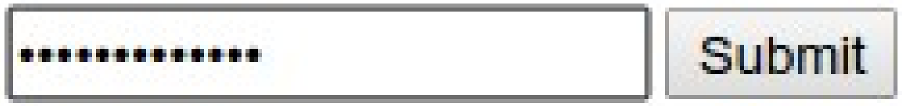




Password input

- Password boxes allow entering of *free text*
- However, the value is never displayed on screen, it's replaced with asterisks:

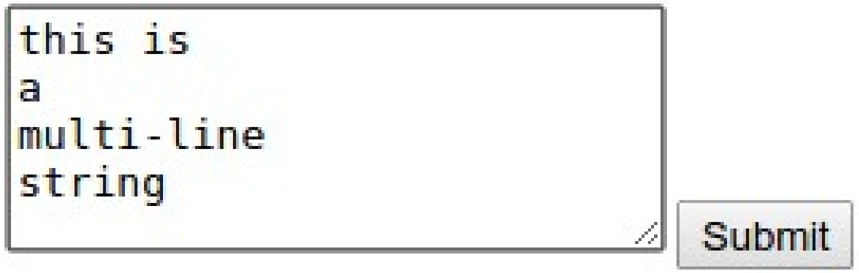
```
<form action="action" method="POST">
        <input type="password" name="mypassword" />
        <input type="submit" value="Submit" name="submit" />

</form>
```



Textarea

- Textarea inputs allow *multi-line* user input
- Textarea, like <select> has an opening and closing tag (input doesn't!)

```
<form action="action" method="POST">
        <textarea name="mytextarea">
        </textarea>
        <input type="submit" value="Submit" name="submit" />

</form>
```


```
this is
a
multi-line
string
```

Submitting Forms

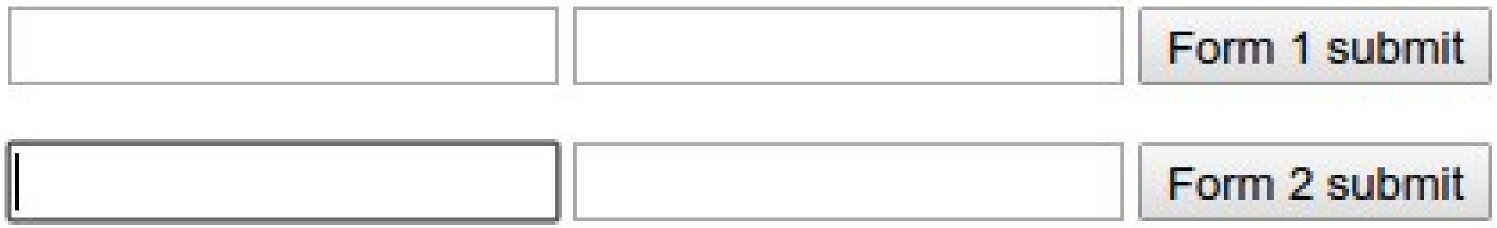
- Forms must be submitted by a user action
- That user action can be:
  - Clicking the *Submit* button
  - Pressing *Enter* while a field has *focus* (is currently selected)

Forms

- A HTML form may have as many elements as you like
- There can be more than one form on a page
- However, only one can be submitted

Multiple forms on one page

```
<form action="action" method="POST">
        <input type="text" name="form1text1" />
        <input type="text" name="form1text2" />
        <input type="submit" name="submit" value="Form 1 submit" />

</form>

<form action="action" method="POST">
        <input type="text" name="form2text1" />
        <input type="text" name="form2text2" />
        <input type="submit" name="submit" value="Form 2 submit" />

</form>
```



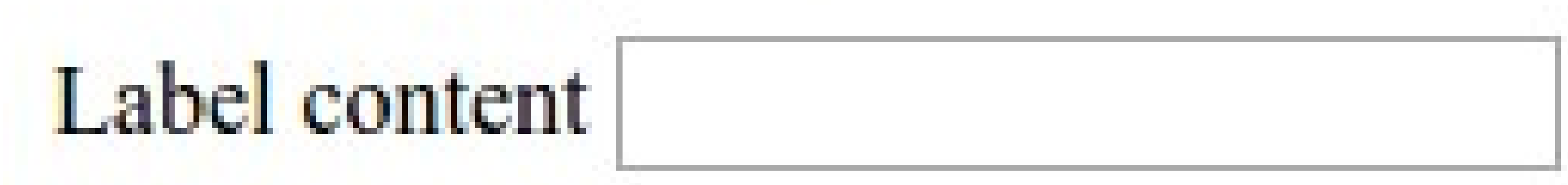
Labels

- Any form field can have a *label*
- Labels use the <label> element and can be linked to a form element using the *for* attribute
- To do this, the element must be given an *ID* attribute which is then referenced in the label's *for* attribute

Labels

- Once a label is assigned to an element, you can click on it and it will act like clicking on the referenced element

```
<label for="myinput">Label content</label> <input type="text" name="myinput"  id="myinput" />
```

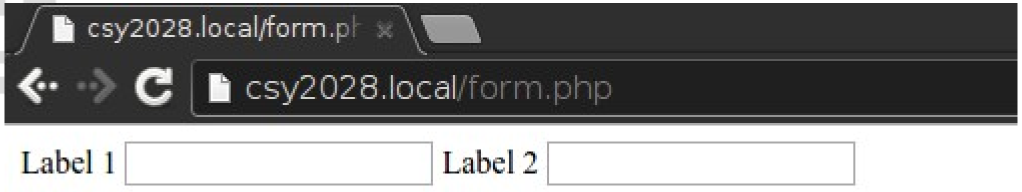


Labels

- Labels can be styled using CSS
- Without CSS, all inputs and labels will appear on the same line

Labels

```
<form action="form.php" method="POST">
                <label for="myinput1">Label 1</label> <input type="text" id="myinput1" />
                <label for="myinput2">Label 2</label> <input type="text" id="myinput2" />
</form>
```



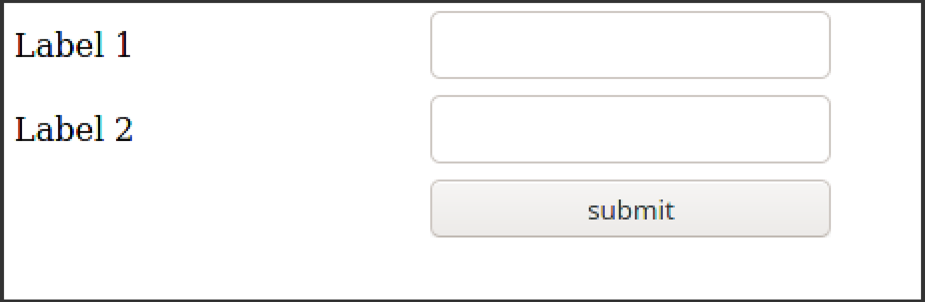
Label 1 Label 2

Labels

- By using display: grid you can position the elements on the same line (You don't need to use tables or divs to achieve this!)

```
<form action="action" method="POST">
                <label for="myinput1">Label 1</label> <input type="text" name="input1" id="myinput1" />
                <label for="myinput2">Label 2</label> <input type="text" name="input2" id="myinput2" />
                <input type="submit" class="submit" name="submit" value="submit" />
</form>
```

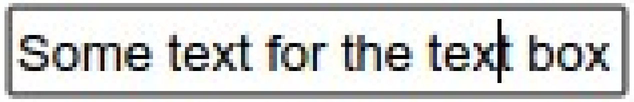
```
       form {
                display: grid;
                width: 400px;
                grid-template-columns: 50% 50%;
                align-items: center;
                grid-gap: 0.5em;
        }
        .submit {
                grid-column: 2;
        }
```



Setting element values

- You can pre-populate forms using the *value* attribute.
- This will place some text in the field when the page loads, without the user having to type it in
- The user can overwrite it or leave it

```
<input type="text" name="myinput" value="Some text for the text box" />
```


Some text for the text box

Setting textarea values

- For textareas you must specify the content between the opening and closing tags:

```
<textarea name="mytextarea">
Some text for the text box
</textarea>
```

- Note: Textareas do not use the value attribute!

Setting select values

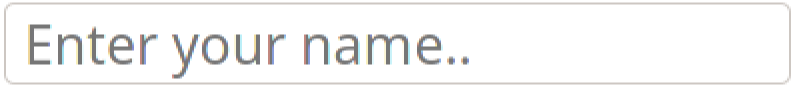
- To set the value of a <select> when the page loads element you must add selected=”selected” to the relevant option

```
<form action="form.php" method="POST">

        <select name="myselect">
                <option value="One">Option 1</option>
                <option selected=”selected” value="Two">Option 2</option>
                <option value="Three">Option 3</option>
        </select>
        <input type="submit" value="Submit" name="submit" />

</form>
```

Placeholders

- Placeholders can be used instead of labels to prompt the user what they should write into a text box
- These are useful form more compact designs

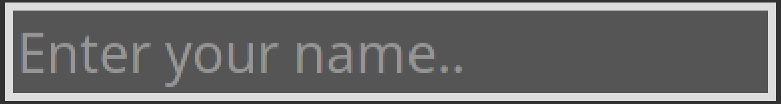
```
<input type="text" name="textbox" placeholder="Enter your name.." />
```


Enter your name..

Styling placeholder

- You can add styles to the placeholder text to match your website theme e.g. a dark theme:

```css
body {
        background-color: #333;
}
input {
        border: 2px solid #ddd;
        color: #fff;
        background-color: #555;
}
input::placeholder {
        color: #ccc;
}
```



Placeholder vs label

- Either placeholders or labels can be used to prompt the user, however, placeholders are only available on text boxes
- Any design which uses checkboxes, radio buttons, etc in addition to text boxes, will need labels for those elements – be consistent!
  - As a rule of thumb If it’s a short form like a login form, use placeholders otherwise use labels

Email forms

- A very crude way of creating an email contact form is using *mailto*
  - **This should generally be avoided, but to do so you'd need a *server side language – something we're not doing until next year!***

Creating a mailto form

- Firstly you need to set the form's *action* to mailto:email@address?subject=Contact Form
- e.g.

```
<form action="mailto:thomas.butler@northampton.ac.uk?subject=Contact Form" method="POST">
```

Mailto forms

- You can then add any fields to the form you like
- When the form is submitted it will open the user’s email client (e.g. Outlook, Thunderbird, etc)
- The email message will be created from elements from the form
- They can then press “send” to send the email

Mailto

- This is a poor solution, but the best available until we do *server side scripting* in year 2
- It is unreliable: depending on the user’s configured mail client, this may not work at all.

Exercise 1

- 1. Create a *login form* asking for
  - Username
  - Password
  - Stay logged in (checkbox)
  - Submit
- 2. Use labels to prompt the user
- 3. Style the form with CSS so it looks user friendly
- Note: Nothing needs to happen when you press submit, just make the form look nice

Exercise 2

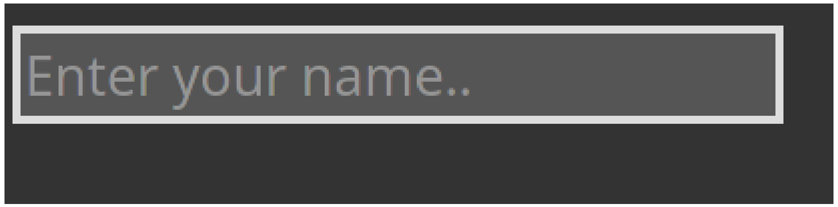
- 1. Use placeholders instead of labels to prompt the user what they should enter into each box.

Exercise 3

- 1. Create a registration form that asks users for the following information:
  - Firstname
  - Surname
  - Email address
  - Password
  - Country (use a <select> box! Enter at least 5 countries as options)
  - Do you accept the terms and conditions (checkbox)
  - Submit button
- 2. Style the form using display: grid. Each label and input should be on its own row
- **Do not use <table> elements, <p> elements, or <div> elements for positioning in your form! The <form> element should only contain the following elements: <input>, <label>, <select> or <textarea>**
- **You can give certain elements class names if they need specific styling**

Exercise 4

- 1. Amend the form from exercise 3 to use a dark theme (light text on a dark background) e.g.


Enter your name..