Appendix 16: HTML — forms (WDD)

The HTML <form> element defines a form that is used to collect user input:

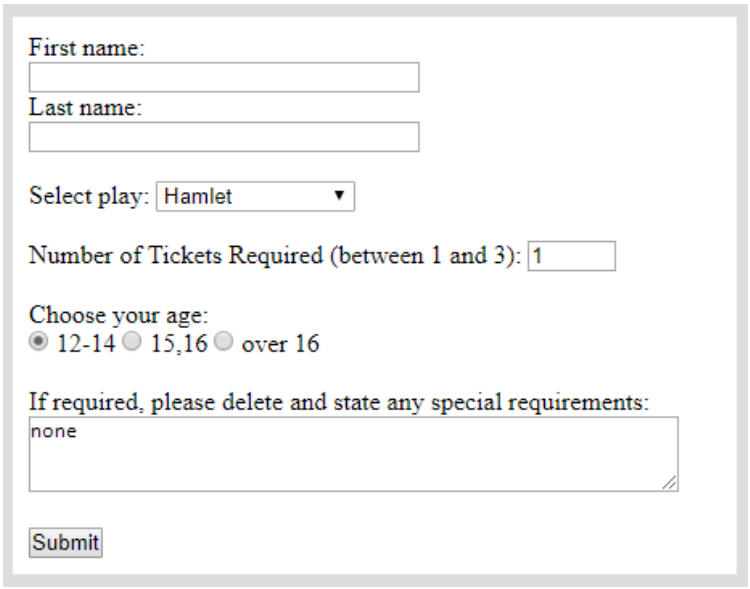
```
<form>
.
form elements
.
</form>
```

HTML 5 can be used to create and perform client-side validation on forms, without the need for JavaScript. Form elements implemented at Higher level are:

- input
  - text
  - number
  - textarea
  - radio
  - submit
- select

Where appropriate, form inputs will be validated using length, presence and range checks.

The following examples have been taken from the form used within the drama page of the *Higher Computing Science example website*. The example website can be downloaded as a zip file from the Higher Computing Science page on SQA's website. To view the full code in context, open and view the source code from the drama page.



Input: example 1 — text
The example below implements two text input boxes, including length and presence checks:

```
<form>
First name:<br>
<input type="text" name="firstname" size="30" maxlength="15"
required><br>
Last name:<br>
<input type="text" name="lastname" size="30" maxlength="15"
required>
</form>
```

The above code includes the following:

`type="text"`	Identifies input type of the form element as text.
`name="firstname"`	The name attribute is required when a form's data is submitted to a server for processing. Although submitting a form to a server is not required until Advanced Higher, it is good practice to include the name attribute in all form elements.
`size="30"`	Width of the input box, in characters, when displayed in a browser.
`maxlength="15"`	Length check limiting input to 15 characters.
`required`	A presence check is applied to the input.

Input: example 2 — number
Number input may be limited to a minimum value, maximum value or both:

```
Number of Tickets Required (between 1 and 3):
<input type="number" name="tickets" min="1" max="3">
```

The above code includes the following:

`type="number"`	Identifies input type of the form element as numeric.
`min="1" max="3"`	A range check to ensure values entered are >=1 and <=3.

Input: example 3 — textarea
A larger text box, for use with extended text input, can be implemented using the textarea form element:

```
If required, please delete and state any special requirements:
<textarea name="message" rows="3" cols="55"> </textarea>
```

The width and height of the textarea element is set using rows and columns.
If required, a length and presence check can be applied to the above input element.

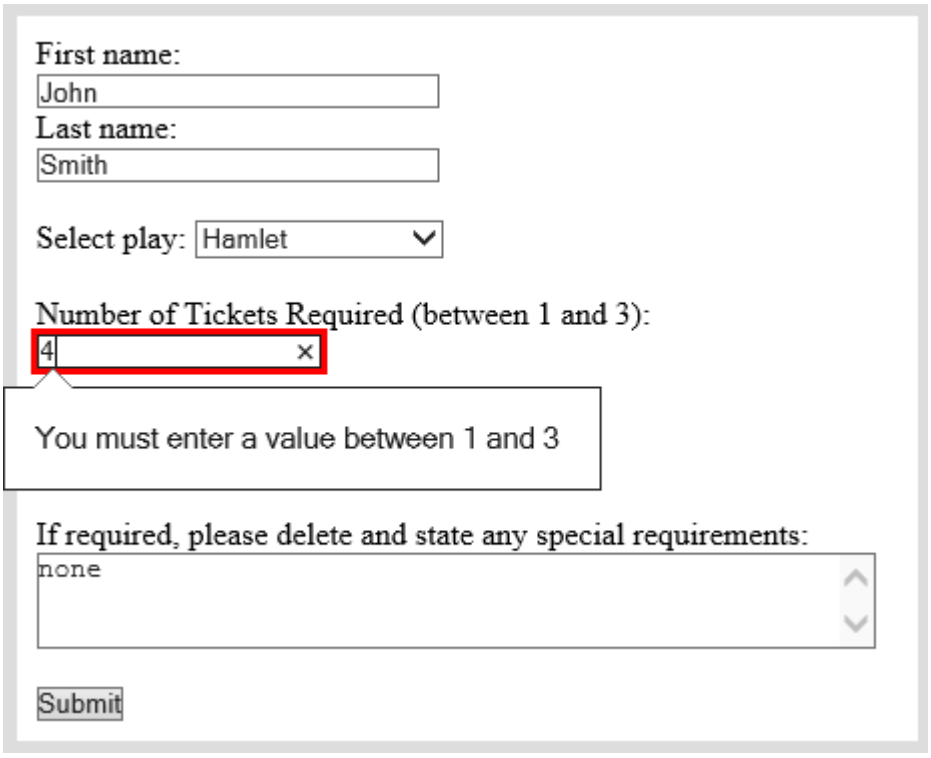
Input: example 4 — radio buttons
Radio input can be implemented using multiple input elements of type radio:

```
Choose your age:<br>
<input type="radio" name="age" value="12 to 14"> 12-14
<br>
<input type="radio" name="age" value="15 or 16"> 15,16
<br>
<input type="radio" name="age" value="17 or over"> over 16
```

When submitted, the above form would return the name attribute “age” along with one of the listed values: 12 to 14, 15 or 16, 17 or over. Although Higher forms are not submitted to a server, it is good practice to include both the name and value attributes.

If the
 elements are omitted from the form, the radio buttons align horizontally.

Input: example 5 — submit
When a form is submitted, the browser shows any validation errors (check browser versions, as this is browser dependent).



To allow candidates visual acknowledgement that an action is performed when the submit button is clicked, a JavaScript onclick event can be applied to the button as shown below:

```
<input type="submit" onclick="alert('Form Entered')" value="Submit">
```

Visual acknowledgement that the form is submitted can be helpful to candidates who are not yet experiencing a response generated by server-side processing.

Select: example 1 — drop-down menu
The select element is used to create a list of possible inputs in the form of a drop-down menu. Input choices are placed inside option elements:

```
Select play:
  <select name="play">
    <option value="hamlet">Hamlet</option>
    <option value="godo">Waiting for Godo</option>
    <option value="brothers">Blood Brothers</option>
    <option value="curious">Curious Incident</option>
  </select>
```

As previously stated, the name and value attributes are not required at Higher, but it is good practice to include both.

Select: example 2 — drop-down menu with size attribute
The size attribute can be used within the select element, to display a set number of options. If the number of options is larger than the size attribute, a scroll bar will automatically appear:

```
<select name="play" size="3">
```

Select: example 3 — drop-down menu with multiple attribute
To allow users to select more than one option, the multiple attribute is used with the select element:

```
<select name="play" size="4" multiple>
```

Pre-populating form input
To aid user input, form elements can be given values that are displayed when the web page loads. These can be left unchanged, deleted or edited by the user, when they are completing the form.

Example 1: text
The value attribute can be used to pre-populate text input elements:

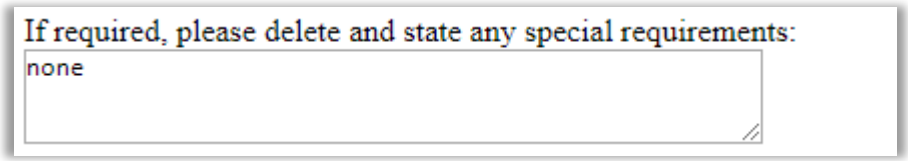
```
<input type="text" name="firstname" value="forename"
size="30" maxlength="15" required>
```

Example 2: number
The value attribute is also used to pre-populate numeric input elements:

```
<input type="number" name="tickets" value="1" min="1"
max="3">
```

Example 3: textarea
To pre-populate a textarea element, the text is included between the start and end elements:

```
<textarea name="message" rows="3" cols="55">none</textarea>
```



Example 4: radio
Checked is used to initially check one of the radio buttons in a form:

```
<input type="radio" name="age" value="12 to 14" checked>
12-14 <br>
```