**University of Wolverhampton**

**School of Engineering, Computational and Mathematical Sciences**

**5CS020 Human-Computer Interaction**

**Workshop 3 – HTML, CSS and JavaScript**

As you will be creating and experimenting with user interface via web pages, it is important that you understand HTML, CSS, and JavaScript.

This workshop continues from the previous workshop in exploring these topics further.

**HTML - Forms**

HTML Forms are required, when you want to collect some data from the site visitor. For example, during user registration you would like to collect information such as name, email address, credit card, etc.

A form will take input from the site visitor and then will post it to a back-end application such as CGI, ASP Script or PHP script etc. The back-end application will perform required processing on the passed data based on defined business logic inside the application.

There are various form elements available like text fields, textarea fields, drop-down menus, radio buttons, checkboxes, etc.

The HTML <form> tag is used to create an HTML form and it has following syntax

<form action = "Script URL" method = "GET|POST">

form elements like input, textarea etc.

</form>

**Form Attributes**

Apart from common attributes, following is a list of the most frequently used form attributes

* action - Backend script ready to process your passed data.
* method - Method to be used to upload data. The most frequently used are GET and POST methods.
* target - Specify the target window or frame where the result of the script will be displayed. It takes values like \_blank, \_self, \_parent etc.
* enctype - You can use the enctype attribute to specify how the browser encodes the data before it sends it to the server. Possible values are (1) application/x-www-form-urlencoded − This is the standard method most forms use in simple scenarios. (2) mutlipart/form-data − This is used when you want to upload binary data in the form of files like image, word file etc.

**HTML Form Controls**

There are different types of form controls that you can use to collect data using HTML form

* Text Input Controls
* Checkboxes Controls
* Radio Box Controls
* Select Box Controls
* File Select boxes
* Hidden Controls
* Clickable Buttons
* Submit and Reset Button
* Text Input Controls

There are three types of text input used on forms

* Single-line text input controls − This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag.
* Password input controls − This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTMl <input> tag.
* Multi-line text input controls − This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag.

**Single-line text input controls**

This control is used for items that require only one line of user input, such as search boxes or names. They are created using HTML <input> tag. Here is a basic example of a single-line text input used to take first name and last name:

<!DOCTYPE html>

<html>

<head>

<title>Text Input Control</title>

</head>

<body>

<form >

First name: <input type = "text" name = "first\_name" />

<br>

Last name: <input type = "text" name = "last\_name" />

</form>

</body>

</html>

Will produce

A picture containing graphical user interface

Description automatically generated

**Attributes**

Following is the list of attributes for <input> tag for creating text field.

* type - indicates the type of input control and for text input control it will be set to text.
* name - used to give a name to the control which is sent to the server to be recognized and get the value.
* value - this can be used to provide an initial value inside the control.
* size - allows to specify the width of the text-input control in terms of characters.
* maxlength - allows to specify the maximum number of characters a user can enter into the text box.

**Password input controls**

This is also a single-line text input but it masks the character as soon as a user enters it. They are also created using HTML <input>tag but type attribute is set to password. Here is an example of a single-line password input used to take user password

<!DOCTYPE html>

<html>

<head>

<title>Password Input Control</title>

</head>

<body>

<form >

User ID : <input type = "text" name = "user\_id" />

<br>

Password: <input type = "password" name = "password" />

</form>

</body>

</html>

This will produce

A picture containing graphical user interface

Description automatically generated

* type - indicates the type of input control and for password input control it will be set to password.
* name - used to give a name to the control which is sent to the server to be recognized and get the value.
* value - this can be used to provide an initial value inside the control.
* size - allows to specify the width of the text-input control in terms of characters.
* maxlength - allows to specify the maximum number of characters a user can enter into the text box.

**Multiple-Line Text Input Controls**

This is used when the user is required to give details that may be longer than a single sentence. Multi-line input controls are created using HTML <textarea> tag. Here is an example of a multi-line text input used to take item description

<!DOCTYPE html>

<html>

<head>

<title>Multiple-Line Input Control</title>

</head>

<body>

<form>

Description : <br />

<textarea rows = "5" cols = "50" name = "description">

Enter description here...

</textarea>

</form>

</body>

</html>

This will produce the following

Text

Description automatically generated with medium confidence

Following is the list of attributes for <textarea> tag.

* Name - Used to give a name to the control which is sent to the server to be recognized and get the value.
* Rows - Indicates the number of rows of text area box.
* Cols - Indicates the number of columns of text area box

**Checkbox Control**

Checkboxes are used when more than one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to checkbox. Here is an example HTML code for a form with two checkboxes

<!DOCTYPE html>

<html>

<head>

<title>Checkbox Control</title>

</head>

<body>

<form>

<input type = "checkbox" name = "maths" value = "on"> Maths

<input type = "checkbox" name = "physics" value = "on"> Physics

</form>

</body>

</html>

This will produce the following



Following is the list of attributes for <checkbox> tag.

* type - indicates the type of input control and for checkbox input control it will be set to checkbox..
* name - used to give a name to the control which is sent to the server to be recognized and get the value.
* value - the value that will be used if the checkbox is selected.
* checked - set to checked if you want to select it by default.

**Radio Button Control**

Radio buttons are used when out of many options, just one option is required to be selected. They are also created using HTML <input> tag but type attribute is set to radio. Here is example HTML code for a form with two radio buttons

<!DOCTYPE html>

<html>

<head>

<title>Radio Box Control</title>

</head>

<body>

<form>

<input type = "radio" name = "subject" value = "maths"> Maths

<input type = "radio" name = "subject" value = "physics"> Physics

</form>

</body>

</html>

Produces



Following is the list of attributes for radio button.

* type - indicates the type of input control and for checkbox input control it will be set to radio.
* name - used to give a name to the control which is sent to the server to be recognized and get the value.
* value - the value that will be used if the radio box is selected.
* checked - set to checked if you want to select it by default.

**Select Box Control**

A select box, also called drop down box which provides option to list down various options in the form of drop down list, from where a user can select one or more options. Here is example HTML code for a form with one drop down box

<!DOCTYPE html>

<html>

<head>

<title>Select Box Control</title>

</head>

<body>

<form>

<select name = "dropdown">

<option value = "Maths" selected>Maths</option>

<option value = "Physics">Physics</option>

</select>

</form>

</body>

</html>

Produce

Graphical user interface, text, application

Description automatically generated

Following is the list of important attributes of <select> tag −

* Name - Used to give a name to the control which is sent to the server to be recognized and get the value.
* Size - This can be used to present a scrolling list box.
* Multiple - If set to "multiple" then allows a user to select multiple items from the menu.
* Following is the list of important attributes of <option> tag
* Value - The value that will be used if an option in the select box box is selected.
* Selected - Specifies that this option should be the initially selected value when the page loads.
* Label - An alternative way of labelling options

**Cascading Style Sheets (CSS)**

Cascading Style Sheets (CSS) describe how documents are presented on screens, in print, or perhaps how they are pronounced. W3C has actively promoted the use of style sheets on the Web since the consortium was founded in 1994.

Cascading Style Sheets (CSS) provide easy and effective alternatives to specify various attributes for the HTML tags. Using CSS, you can specify a number of style properties for a given HTML element. Each property has a name and a value, separated by a colon (:). Each property declaration is separated by a semi-colon (;).

First let's consider an example of HTML document which makes use of <font> tag and associated attributes to specify text colour and font size

<!DOCTYPE html>

<html>

<head>

<title>HTML CSS</title>

</head>

<body>

<p><font color = "green" size = "5">Hello, World!</font></p>

</body>

</html>

We can re-write above example with the help of Style Sheet as follows

<!DOCTYPE html>

<html>

<head>

<title>HTML CSS</title>

</head>

<body>

<p style = "color:green; font-size:24px;" >Hello, World!</p>

</body>

</html>

Produces

Logo, company name

Description automatically generated

You can use CSS in three ways in your HTML document −

* External Style Sheet − Define style sheet rules in a separate .css file and then include that file in your HTML document using HTML <link> tag.
* Internal Style Sheet − Define style sheet rules in header section of the HTML document using <style> tag.
* Inline Style Sheet − Define style sheet rules directly along-with the HTML elements using style attribute.

**External Style Sheet**

If you need to use your style sheet to various pages, then its always recommended to define a common style sheet in a separate file. A cascading style sheet file will have extension as .css and it will be included in HTML files using <link> tag.

Consider we define a style sheet file style.css which has following rules

.red {

color: red;

}

.thick {

font-size:20px;

}

.green {

color:green;

}

Here we defined three CSS rules which will be applicable to three different classes defined for the HTML tags. Now let's make use of the above external CSS file in our following HTML document

<!DOCTYPE html>

<html>

<head>

<title>HTML External CSS</title>

<link rel = "stylesheet" type = "text/css" href = "style.css">

</head>

<body>

<p class = "red">This is red</p>

<p class = "thick">This is thick</p>

<p class = "green">This is green</p>

<p class = "thick green">This is thick and green</p>

</body>

</html>

Produces

Graphical user interface, text, application

Description automatically generated

Internal Style Sheet

If you want to apply Style Sheet rules to a single document only, then you can include those rules in header section of the HTML document using <style> tag. Rules defined in internal style sheet overrides the rules defined in an external CSS file. Let's re-write above example once again, but here we will write style sheet rules in the same HTML document using <style> tag

<!DOCTYPE html>

<html>

<head>

<title>HTML Internal CSS</title>

<style type = "text/css">

.red {

color: red;

}

.thick{

font-size:20px;

}

.green {

color:green;

}

</style>

</head>

<body>

<p class = "red">This is red</p>

<p class = "thick">This is thick</p>

<p class = "green">This is green</p>

<p class = "thick green">This is thick and green</p>

</body>

</html>

Produces the same:

Graphical user interface, text

Description automatically generated

**Inline Style Sheet**

You can apply style sheet rules directly to any HTML element using style attribute of the relevant tag. This should be done only when you are interested to make a particular change in any HTML element only. Rules defined inline with the element overrides the rules defined in an external CSS file as well as the rules defined in <style> element. Let's re-write above example once again, but here we will write style sheet rules along with the HTML elements using style attribute of those elements.

<!DOCTYPE html>

<html>

<head>

<title>HTML Inline CSS</title>

</head>

<body>

<p style = "color:red;">This is red</p>

<p style = "font-size:20px;">This is thick</p>

<p style = "color:green;">This is green</p>

<p style = "color:green;font-size:20px;">This is thick and green</p>

</body>

</html>

Produces the same:

Graphical user interface, text

Description automatically generated