Topic 4
Using Advanced HTML

CST4013 | Website Designing

Learning Outcomes

1

Discuss the HTML Forms control and structures.

2

Explain the concept of CSS and types of CSS Selectors.

3

Describe the concept and uses of JavaScript.

HTML Formatting Text

- There are two categories of markup in HTML formatting text tags:
  - Structural markup: the elements that can use to describe both headings and paragraphs.
  - Semantic markup: which provides extra information; such as where emphasis is placed in a sentence or that something that have written is a quotation.

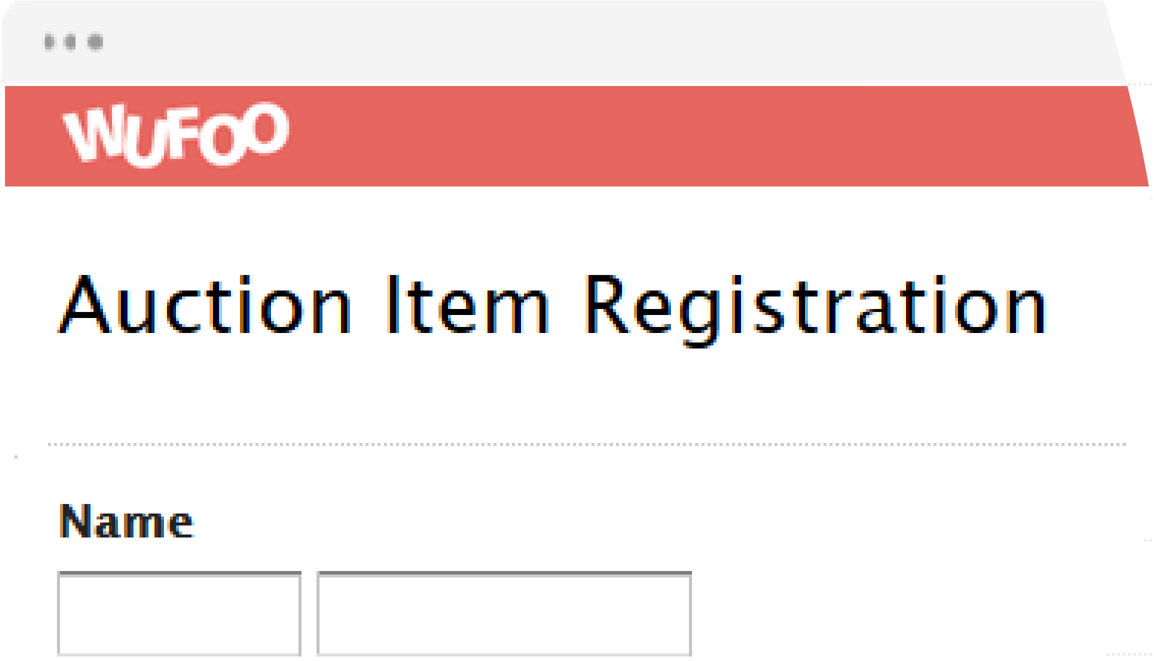
HTML Formatting Text

Bold `<b>` `</b>`	By enclosing words in the tags <b> and </b> we can make characters appear bold.
Italic `<i>` `</i>`	By enclosing words in the tags <i> and </i> we can make characters appear italic.
Sup `<sup>` `</sup>`	The <sup> element is used to contain characters that should be superscript such as the suffixes of dates or mathematical concepts like raising a number to a power such as 2^2.
Sub `<sub>` `</sub>`	The <sub> element is used to contain characters that should be subscript. It is commonly used with foot notes or chemical formulas such as H_20.
Strong `<strong>` `</strong>`	The use of the <strong> element indicates that its content has strong importance. For example, the words contained in this element might be said with strong emphasis

HTML Formatting Text

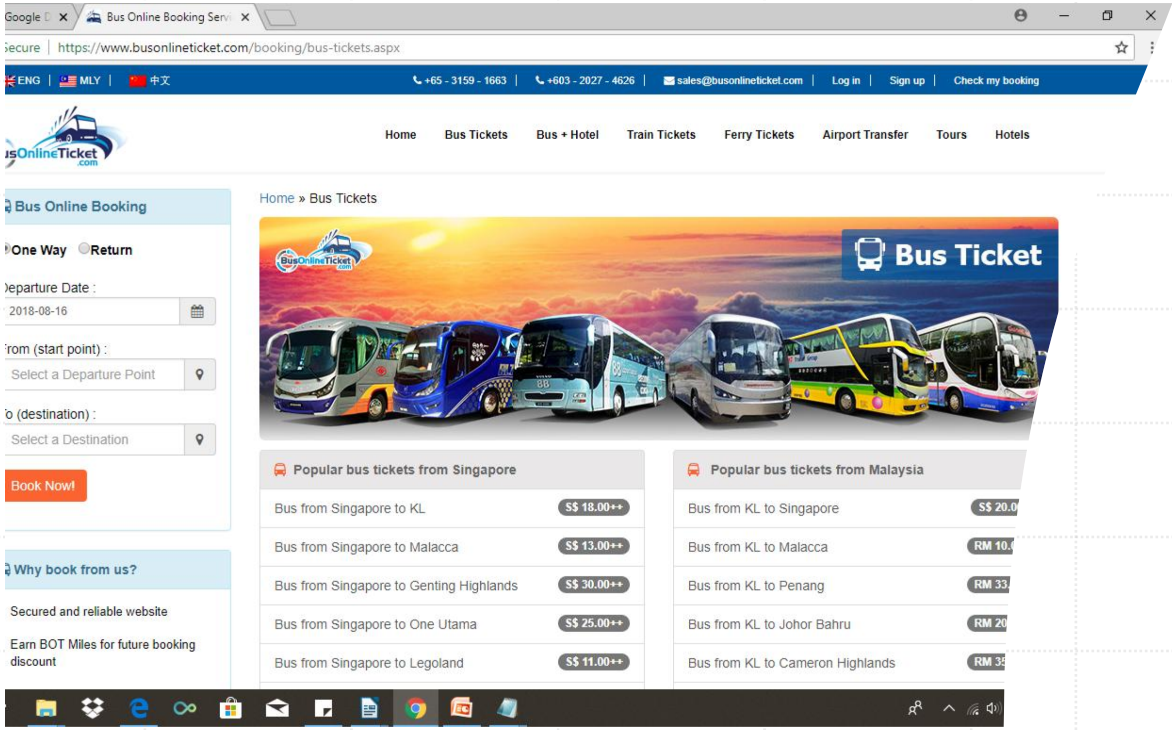
em <em> </em>	The <em> element indicates emphasis that subtly changes the meaning of a sentence.
Blockquote <blockquote> </blockquote>	The <blockquote> element is used for longer quotes that take up an entire paragraph. Note how the <p> element is still used inside the <blockquote> element.
Quote <q> </q>	The <q> element is used for shorter quotes that sit within a paragraph.
Ins <ins> </ins>	The <ins> element can be used to show content that has been inserted into a document,
Del <del> </del>	The <del> element can show text that has been deleted from it.

HTML Forms



Auction Item Registration

Name


Google D ×
Bus Online Booking Serv ×
ecure | https://www.busonlineticket.com/booking/bus-tickets.aspx
ENG | MLY | 中文
+65 - 3159 - 1663 | +603 - 2027 - 4626 | sales@busonlineticket.com | Log in | Sign up | Check my booking
usOnlineTicket .com
Home
Bus Tickets
Bus + Hotel
Train Tickets
Ferry Tickets
Airport Transfer
Tours
Hotels
Bus Ticket

Home » Bus Tickets

Bus Online Booking

○ One Way ○ Return

eparture Date :

2018-08-16

rom (start point) :

Select a Departure Point

o (destination) :

Select a Destination

Book Now!

Why book from us?

Secured and reliable website

Earn BOT Miles for future booking discount

Popular bus tickets from Singapore	
Bus from Singapore to KL	S$ 18.00++
Bus from Singapore to Malacca	S$ 13.00++
Bus from Singapore to Genting Highlands	S$ 30.00++
Bus from Singapore to One Utama	S$ 25.00++
Bus from Singapore to Legoland	S$ 11.00++

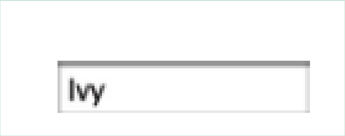
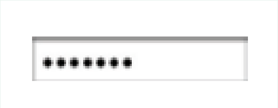
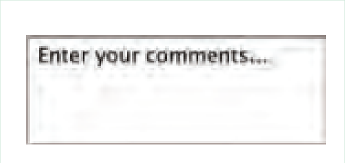
Popular bus tickets from Malaysia	
Bus from KL to Singapore	S$ 20.0
Bus from KL to Malacca	RM 10.
Bus from KL to Penang	RM 33.
Bus from KL to Johor Bahru	RM 20
Bus from KL to Cameron Highlands	RM 3

HTML Forms

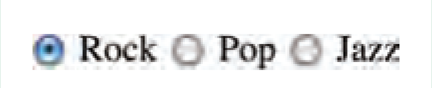
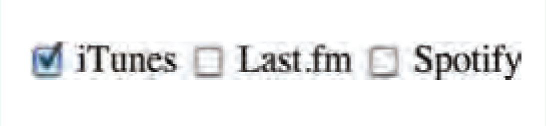
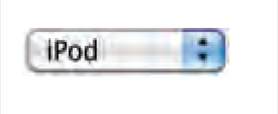
- The HTML `<form>` element defines a form that is used to collect user input.
- An HTML form contains **form elements**.
- Form elements are different types of input elements, like text fields, checkboxes, radio buttons, submit buttons, and more.
  - Syntax:

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

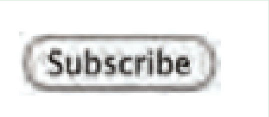
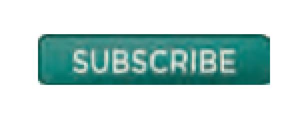
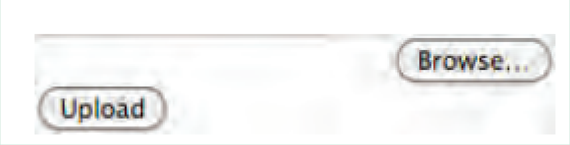
HTML Forms Controls

Adding Text	Description	Example
Text input (single-line)	Used for a single line of text such as email addresses and names.	 Ivy
Password input	Like a single line text box but it masks the characters entered.	 •••••••
Text area (multi-line)	For longer areas of text, such as messages and comments.	 Enter your comments...

HTML Forms Controls

Making Choices	Description	Example
Radio buttons	For use when a user must select one of a number of options.	 ● Rock ○ Pop ○ Jazz
Checkboxes	When a user can select and unselect one or more optionsText area (multi-line).	 ☑ iTunes □ Last.fm □ Spotify
Drop-down boxes	When a user must pick one of a number of options.	 iPod

HTML Forms Controls

Submitting Forms	Description	Example
Submit buttons	To submit data from your form to another web page.	
Image buttons	Similar to submit buttons but they allow you to use an image.	
File upload	Allows users to upload files (e.g. images) to a website.	

HTML Form Structure

<form action="/submit" method="POST">	
<form>	• Form controls live inside a <form> element. • This element should always carry the action attribute and will usually have a method and id attribute.
action	• Every <form> element requires an action attribute. • Its value is the URL for the page on the server that will receive the information in the form when it is submitted.
method	• Forms can be sent using one of two methods: get or post. • With the get method, the values from the form are added to the end of the URL specified in the action attribute. • With the post method the values are sent in what are known as HTTP headers.

HTML Form Elements

- The `<input>` Element
- The `<select>` Element
- The `<textarea>` Element
- The `<button>` Element

HTML Form Elements

Tag	Description
<form>	Defines an HTML form for user input
<input>	Defines an input control
<textarea>	Defines a multiline input control (text area)
<label>	Defines a label for an <input> element
<fieldset>	Groups related elements in a form
<legend>	The <legend> element can come directly after the opening <fieldset> tag and contains a caption which helps identify the purpose of that group of form controls.

HTML Form Elements

Tag	Description
`<select>`	Defines a drop-down list
`<optgroup>`	Defines a group of related options in a drop-down list
`<option>`	Defines an option in a drop-down list
`<button>`	Defines a clickable button
`<datalist>`	Specifies a list of pre-defined options for input controls
`<output>`	Defines the result of a calculation

HTML Input Types

Types	Scripting	Description
Text input	<input type="text" name="username" size="15" maxlength="30"/>	Defines a single-line text input field.
Password input	<input type="password" name="password" size="15" maxlength="30"/>	Defines a single-line text input field with the hidden characters.
Textarea	<textarea name="comments" cols="20" rows="4">Enter your comments... </textarea>	Defines a multi-line text input field.

HTML Input Element

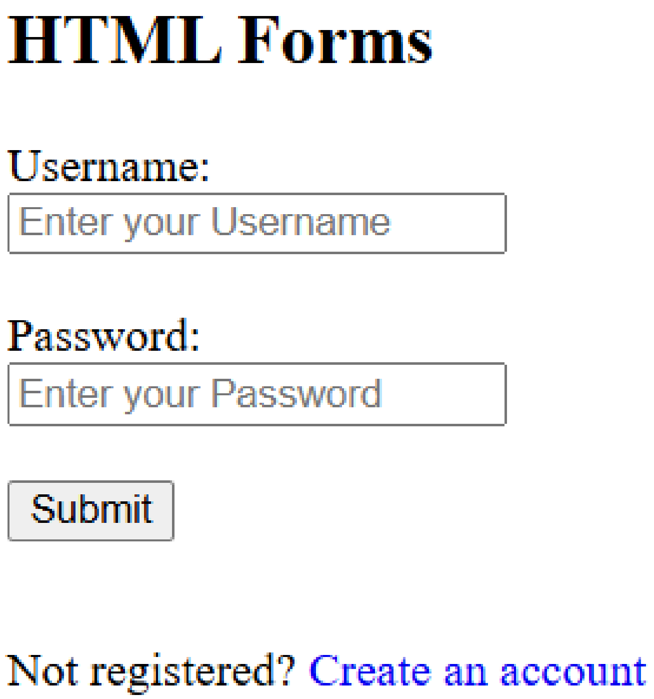
Types	Scripting	Description
Radio Button	`<input type="radio" name="genre" value="rock" checked="checked"/> Rock`	Radio buttons allow users to pick just one of a number of options.
Checkbox	`<input type="checkbox" name="service" value="itunes" checked="checked"/> iTunes`	Checkboxes allow users to select (and unselect) one or more options in answer to a question
Dropdown Listbox	`<select name="devices"> <option value="ipod">iPod</option> <option value="radio">Radio</option> </select>`	A drop down list box (also known as a select box) allows users to select one option from a drop down list.

HTML Input Element

Types	Scripting	Description
Submit Button	`<input type="submit" name="subscribe" value="Subscribe" />`	The submit button is used to send a form to the server.
Image Button	`<input type="image" src="images/subscribe.jpg" width="100" height="20" />`	Used an image for submit button.
File Input Box	`<input type="file" name="user-song" /> <br/> <input type="submit" value="Upload" />`	Allow users to upload a file for example an image, video, mp3, or a PDF.

Exercise

- Write HTML scripting based on the following HTML forms diagram.



HTML Forms

Username:

Enter your Username

Password:

Enter your Password

Not registered? Create an account

HTML Input Restrictions

Attribute	Description
disabled	Specifies that an input field should be disabled
max	Specifies the maximum value for an input field
maxlength	Specifies the maximum number of character for an input field
min	Specifies the minimum value for an input field
pattern	Specifies a regular expression to check the input value against
types	Specifies the type of input, which automatically includes certain validations (e.g., type="email" ensures a valid email format).

HTML Input Restrictions

Attribute	Description
readonly	Specifies that an input field is read only (cannot be changed)
required	Specifies that an input field is required (must be filled out)
size	Specifies the width (in characters) of an input field
step	Specifies the legal number intervals for an input field
value	Specifies the default value for an input field

CSS Introduction

- Cascading Style Sheets (CSS) is a style sheet language used for describing describe the presentation and layout of HTML documents.
- CSS describes **how HTML elements are to be displayed on screen, paper, or in other media**
- CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

Key Features of CSS

Selectors and Properties

1. CSS uses selectors to target HTML elements and apply styles to them.

2. Each style consists of a property (like color, font-size, or margin) and a value (like red, 16px, or 10px).

Responsive Design

1. CSS enables responsive design techniques, allowing web pages to adapt to different screen sizes and devices using media queries and flexible layouts.

Key Features of CSS

Separation of Content and Style

1. CSS allows you to keep your HTML structure (content) separate from its presentation (style).

2. This separation makes it easier to maintain and update websites.

Cascading Order

1. The "cascading" aspect of CSS means that styles can be applied in layers.

2. If there are conflicting styles, the browser applies the rules based on specificity and the order they are defined.

Visual Effects

1. CSS includes features like animations, transitions, and transformations, allowing developers to create dynamic and engaging user experiences.

CSS Flexbox

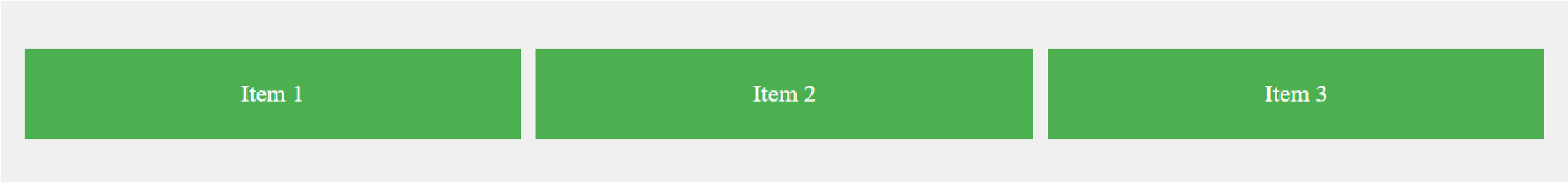
- CSS Flexbox (Flexible Box Layout) is a powerful layout model in CSS that allows you to design responsive and flexible web layouts with ease.
- Flexbox is a layout module that enables the arrangement of elements in a container to be flexible and efficient, even when the size of the items is unknown or dynamic.
- Key concepts:
  - Flex Container: The parent element that uses the display: flex; property.
  - Flex Items: The child elements inside the flex container.
- Syntax:

```
.container {
    display: flex;
   /*or display: inline-flex; */
}
```

Example of CSS Flexbox

```
<style>
  .container {
  display: flex;
  flex-direction: row;
  justify-content: space-between;
  align-items: center;
  height: 100px;
  background-color: #f0f0f0;
  padding: 10px;
  }
```

```
.item {
  flex: 1;
  margin: 5px;
  padding: 20px;
  background-color: #4caf50;
  color: white;
  text-align: center;
}
 </style>
```



Item 1

Item 2

Item 3

Flex Container Properties

flex-direction:	**row**	**row-reverse**	**column**	**column-reverse**
Defines the direction flex items are placed in the flex container.	Default; items are placed in a row.	Items are placed in a row in reverse order.	Items are placed in a column.	Items are placed in a column in reverse order.
flex-wrap:	**nowrap**	**wrap**	**wrap-reverse**	
Controls whether flex items should wrap onto multiple lines.	Default; items are laid out in a single line.	Items will wrap onto multiple lines.	Items will wrap onto multiple lines in reverse order.	

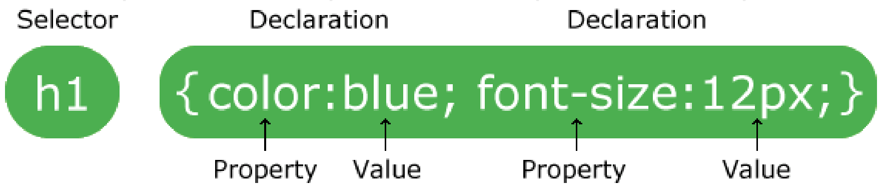
Flex Container Properties

justify-content:	**flex-start**	**flex-end**	**center**	**space-between**	**space-around**
Aligns flex items along the main axis (horizontal by default).	Aligns items to the start of the container.	Aligns items to the end of the container.	Centers items in the container.	Distributes items evenly; first item at start, last at end.	Distributes items evenly with space around them
align-item	**flex-start**	**flex-end**	**center**	**baseline**	**stretch**
Aligns flex items along the cross axis (vertical by default).	Aligns items to the start of the container.	Aligns items to the end of the container.	Centers items in the container.	Aligns items based on their baseline.	Stretches items to fill the container (default).

CSS Syntax

- A CSS rule-set consists of a **selector** and a **declaration block.**
- The selector indicate which element the rule applies to.
- The declaration indicate how the elements referred to in the selector should be styled.
- The declaration block contains one or more declarations separated by semicolons.

- Each declaration includes a CSS **property name** and **a value**, separated by a colon.
- A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.



Selector Declaration Declaration

```
h1 {color:blue; font-size:12px;}
```

Property Value Property Value

CSS Syntax Example

```
<!DOCTYPE html>
 <html>
  <head>
   <style>
      p {
         color:red;
         text-align:center;
        }
</style>
   </head>
     <body>
        <p>Hello World!</p>
        <p>These paragraphs are styled with CSS.</p>
    </body>
</html>
```

In the following example all <p> elements will be center-aligned, with a red text color:

Element Selector

- CSS selectors are used to find or select HTML elements based on their element name, id, class, attribute, and more.
- Element Selector
  - The element selector selects elements based on the element name.
  - Example: All <p> elements will be center-aligned, with a red text color).

```
p {
    text-align:center;
    color:red;
}
```

ID Selector

- The id selector uses the id attribute of an HTML element to select a specific element.
- The id of an element should be unique within a page, so the id selector is used to select one unique element.
- To select an element with a specific id, write a hash (#) prefix, followed by the id of the element.
- The style rule below will be applied to the HTML element with id="para1":

```
#para1 {
    text-align:center;
    color:red;
}
```

Class Selector

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period (.) prefix, followed by the name of the class.
- In the example below, all HTML elements with class="center" will be red and center-aligned:

```
.center {
    text-align:center;
    color:red;
}
```

Class Selector

- You can also specify that only specific HTML elements should be affected by a class.
- In the example below, only <p> elements with class="center" will be center-aligned:

```
p.center {
    text-align:center;
    color:red;
}
```

CSS Types

There are three ways of inserting a style sheet:

1. External style sheet
2. Internal style sheet
3. Inline style

External Style Sheet

- An external style sheet is used to define the style for many HTML pages.
- Each page must include a reference to the external style sheet file inside the <link> element.
- To use an external style sheet, add a link to it in the <head> section of each HTML page:

```
<head>
<link rel="stylesheet" type="text/
css" href="mystyle.css">
</head>
```

External Style Sheet

- An external style sheet can be written in any text editor.
- The file should not contain any html tags.
- The style sheet file must be saved with a .css extension.
- Here is how the "mystyle.css" looks:

```
body {
     background-color:lightblue;
  }
h1  {
     color: navy;
     margin-left:20px;
 }
```

Note: Do not add a space between the property value and the unit (such as `margin-left: 20 px;`).
The correct way is: `margin-left: 20px;`

Internal Style Sheet

- An internal style sheet may be used if one single page has a unique style.
- Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

```
<head>
 <style>
  body {
    background-color:linen;
 }
   h1 {
     color:maroon;
     margin-left:40px;
 }
 </style>
</head>
```

Inline Styles

- An inline style may be used to apply a unique style for a single element.
- To use inline styles, add the style attribute to the relevant element.
- The style attribute can contain any CSS property.
- The example below shows how to change the color and the left margin of a <h1> element:

```
<h1 style="color:blue;margin-
left:30px;">
This is a heading</h1>
```

Tip: An inline style loses many of the advantages of a style sheet (by mixing content with presentation). Use this method sparingly.

JavaScript Introduction

- JavaScript is a high-level, dynamic, interpreted programming language primarily used for client-side web development.

- It allows developers to create interactive and dynamic web content, enhancing the user experience by providing features such as interactivity, animations, and dynamic updates without the need to reload the entire web page.

- JavaScript was initially created by Brendan Eich at Netscape Communications Corporation in 1995, originally named LiveScript, but later renamed JavaScript to capitalize on the popularity of Java.

JavaScript Functions

- Interactivity
  - Making web pages interactive through event handling.
- Dynamic Content Manipulation
  - Changing content and styles on the fly.
- Form Validation
  - Validating user inputs before submission.
- Animation
  - Creating animations for a better user experience.
- AJAX
  - Enabling asynchronous communication with the server.
- Data Handling and Storage
  - Managing data and providing persistence.

JavaScript Methods

- ✓ Inline JavaScript
- Internal JavaScript
- External JavaScript
- Defer and Async Attributes
- DOM Methods

Inline JavaScript

- Inline JavaScript refers to JavaScript code that is written directly within HTML elements, typically using the onclick, onchange, onmouseover, or other event attributes.

```
<!DOCTYPE html>
<html>
<head>
    <title>Inline JavaScript Example</title>
</head>
<body>
    <button onclick="alert('Hello,
world!')">Click me</button>
</body>
</html>
```

Internal JavaScript

- Internal JavaScript refers to JavaScript code that is placed within a <script> tag inside the HTML document, typically within the <head> or <body> section. .
- This is useful for including JavaScript that is only relevant to a specific page.

```
<!DOCTYPE html>
<html>
<head>
    <title>Internal JavaScript Example</title>
    <script>
        function showAlert() {
            alert('Hello, world!');
        }
    </script>
</head>
<body>
    <button onclick="showAlert()">Click
me</button>
</body>
</html>
```

External JavaScript

- External JavaScript is written in a separate .js file and linked to the HTML document using the <script> tag with the src attribute.
- This is the preferred method for larger scripts and for maintaining clean and manageable code.

HTML file (index.html)

```
<!DOCTYPE html>
<html>
<head>
    <title>External JavaScript Example</title>
    <script src="script.js"></script>
</head>
<body>
    <button onclick="showAlert()">Click
me</button>
</body>
</html>
```

JavaScript File
(script.js)

```
function showAlert() {
    alert('Hello, world!');
}
```

Using the Defer Attribute (External JavaScript attributes)

- The defer attribute tells the browser to download the JavaScript file during the HTML parsing process but wait to execute the script until the HTML document has been fully parsed.
- This ensures that the script does not block the rendering of the page.

```
<!DOCTYPE html>
<html>
<head>
    <title>Defer Attribute Example</title>
    <script src="script.js" defer></script>
</head>
<body>
    <button onclick="showAlert()">Click
me</button>
</body>
</html>
```

Using the Async Attribute (External JavaScript attributes)

- The async attribute also allows the browser to download the JavaScript file during the HTML parsing process, but it does not guarantee the order of execution.

```
<!DOCTYPE html>
<html>
<head>
    <title>Async Attribute Example</title>
    <script src="script.js" async></script>
</head>
<body>
    <button onclick="showAlert()">Click
me</button>
</body>
</html>
```

Document Object Model (DOM) Methods

- DOM (Document Object Model) is a programming interface for web documents.
- It represents the structure of the document as a tree of objects, allowing scripts to manipulate the content, structure, and style of web pages dynamically.
- JavaScript can be used to interact with the DOM to change the content and style of HTML elements, respond to user events, and modify the document structure..

```
<!DOCTYPE html>
<html>
<head>
    <title>DOM Method Example</title>
</head>
<body>
    <button id="loadScript">Load Script</button>

    <script>
        document.getElementById('loadScript').onclick = 
function() {
            var script = document.createElement('script');
            script.src = 'script.js';
            document.head.appendChild(script);
        };
    </script>
</body>
</html>
```

Thank you