Lecture No: 6
Topics: Iframes and Forms

Block and Inline Elements

- Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is block or inline

The <div> Element

- The <div> element is often used as a container for other HTML elements.
- The <div> element has no required attributes, but style, class and id are common.
- When used together with CSS, the <div> element can be used to style blocks of content:
- **Example**
- <div style="background-color:royalblue;color:white;padding:20px;">
  <h2>Rizal Park</h2>
  <p>Rizal Park (Filipino: Liwasang Rizal), also known as Luneta Park or simply Luneta, is a historical urban park in the Philippines.</p>
  </div>

Block level elements in HTML:

<address>	<dd>	<figcaption>	<header>	<noscript>	<section>
<article>	<div>	<figure>	<hr>	<p>	<tfoot>
<aside>	<dl>	<footer>	<li>	<pre>	<ul>
<blockquote>	<dt>	<form>	<main>	<output>	<table>
<canvas>	<fieldset>	<h1>-<h6>	<nav>	<ol>	<video>

Inline Elements

- An inline element does not start on a new line and only takes up as much width as necessary.
- This is an inline <span> element inside a paragraph.
  Example
- <span>Hello</span>
  <span>World</span>

The <span> Element

- The <span> element is often used as a container for some text.
- The <span> element has no required attributes, but style, class and id are common.
- When used together with CSS, the <span> element can be used to style parts of the text:
  Example
- **<h1>My <span style="color:red">Important</span> Heading</h1>**

The class Attribute

- The class attribute specifies one or more class names for an HTML element.
- The class name can be used by CSS and JavaScript to perform certain tasks for elements with the specified class name.
- In CSS, to select elements with a specific class, write a period (.) character, followed by the name of the class.
- **Note:** The class name is case sensitive!

- Use CSS to style all elements with the class name "city":

PLACE THIS IN THE HEAD SECTION

```
<style>
.city {
  background-color: tomato;
    color: white;
    padding: 10px;
}
</style>
```

PLACE THIS IN THE BODY SECTION

```
<h2 class="city">Manila</h2>
<p>Manila is the capital of Philippines.</p>
<h2 class="city">Paris</h2>
<p>Paris is the capital of France.</p>
<h2 class="city">Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>
```

The id Attribute

- The id attribute specifies a unique id for an HTML element (the value must be unique within the HTML document).
- The id value can be used by CSS and JavaScript to perform certain tasks for a unique element with the specified id value.
- In CSS, to select an element with a specific id, write a hash (#) character, followed by the id of the element.
- **Note:** The id value is case-sensitive.
- **Note:** The id value must contain at least **one** character, and must **not** contain whitespace (spaces, tabs, etc.).

Use CSS to style an element with the id "myHeader":

PLACE THIS IN THE HEAD SECTION

-
  ```
  <style>
  #myHeader {
     background-color: lightblue;
     color: black;
     padding: 40px;
     text-align: center;
  }
  </style>
  ```
  PLACE THIS IN THE BODY SECTION
- `<h1 id="myHeader">My Header</h1>`

Difference Between Class and ID

- An HTML element can only have one unique id that belongs to that single element, while a class name can be used by multiple elements:

-
  ```
  <style>
  /* Style the element with the id "myHeader" */
  #myHeader {
     background-color: lightblue;
  ```

```
  color: black;
  padding: 40px;
  text-align: center;
}
/* Style all elements with the class name "city" */
.city {
  background-color: tomato;
  color: white;
  padding: 10px;
}
</style>
  <!-- A unique element -->
<h1 id="myHeader">My Cities</h1>

<!-- Multiple similar elements -->
<h2 class="city">Manila</h2>
<p>Manila is the capital of Philippines.</p>

<h2 class="city">Paris</h2>
<p>Paris is the capital of France.</p>

<h2 class="city">Tokyo</h2>
<p>Tokyo is the capital of Japan.</p>
```

Bookmarks with ID and Links

- HTML bookmarks are used to allow readers to jump to specific parts of a Web page.
- Bookmarks can be useful if your webpage is very long.
- To make a bookmark, you must first create the bookmark, and then add a link to it.
- When the link is clicked, the page will scroll to the location with the bookmark.

Example

- First, create a bookmark with the id attribute:

```
<h2 id="C4">Chapter 4</h2>
```

- Then, add a link to the bookmark ("Jump to Chapter 4"), from within the same page:

```
<a href="#C4">Jump to Chapter 4</a>
```

HTML Iframes

- An iframe is used to display a web page within a web page.
- **Iframe Syntax**

An HTML iframe is defined with the <iframe> tag:

```
<iframe src="URL"></iframe>
```

The src attribute specifies the URL (web address) of the inline frame page.

Set Height and Width

- Use the height and width attributes to specify the size of the iframe.

- The attribute values are specified in pixels by default, but they can also be in percent (like "80%").
- Example
- **<iframe src="demo_iframe.htm" height="200" width="300"></iframe>**
- Or you can use CSS to set the height and width of the iframe:
- Example
- **<iframe src="demo_iframe.htm" style="height:200px;width:300px;"></iframe>**

Remove the Border

- By default, an iframe has a border around it.
- To remove the border, add the style attribute and use the CSS border property:

Example

<iframe src="demo_iframe.htm" style="border:none;"></iframe>

- With CSS, you can also change the size, style and color of the iframe's border:

Example

- <iframe src="demo_iframe.htm" style="border:2px solid red;"></iframe>

Target for a Link

- An iframe can be used as the target frame for a link.
- The target attribute of the link must refer to the name attribute of the iframe:
- Example
- **<iframe src="demo_iframe.htm" name="iframe_a"></iframe>**

Forms

Form Fundamentals

- are composed of one or more text-input boxes, clickable buttons, multiple-choice checkboxes, and even pull-down menus and image maps, all placed inside the <form> tag.

The <form> Tag

Function	Defines a form
Attributes	accept, action, charset, class, dir, enctype, id, lang, method, name, onClick, onDblClick, onKeyDown, onKeyPress, onKeyUp, onMouseDown, onMouseMove, onMouseOut, onMouseOver, onMouseUp, onReset, onSubmit, style, target, title
End tag	</form>; never omitted
Contains	form_content
Used in	block

The <input> Element

- The <input> element is the most important form element.
- The <input> element can be displayed in several ways, depending on the type attribute.

Type	**Description**
<input type="text">	Defines a one-line text input field
<input type="radio">	Defines a radio button (for selecting one of many choices)
<input type="submit">	Defines a submit button (for submitting the form)

<u>**Text Input**</u>

<input type="text"> defines a one-line input field for **text input**:

Example

```
<form>
  First name:<br>
  <input type="text" name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
</form>
```

Note: The form itself is not visible. Also note that the default width of a text field is 20 characters.

Radio Button Input

<input type="radio"> defines a **radio button**.

Radio buttons let a user select ONE of a limited number of choices:

Example

```
<form>
  <input type="radio" name="gender" value="male"      checked> Male<br>
  <input type="radio" name="gender" value="female"> Female<br>
  <input type="radio" name="gender" value="other">   Other
</form>
```

The Submit Button

- <input type="submit"> defines a button for submitting the form data to a form-handler.
- The form-handler is typically a server page with a script for processing input data.
- The form-handler is specified in the form's action attribute.
- **Example**

-
  ```
  <form action="/action_page.php">
    First name:<br>
    <input type="text" name="firstname" value=“ Nicole"><br>
    Last name:<br>
    <input type="text" name="lastname" value=“  Pasqual"><br><br>
    <input type="submit" value="Submit">
  </form>
  ```

The action Attribute

- The required action attribute for the <form> tag gives the URL of the application that is to receive and process the form's data.
- Syntax:

A typical <form> tag with the action attribute looks like this:

- <form action="/action_page.php">

NOTE: If the action attribute is omitted, the action is set to the current page.

Target Attribute

- The target attribute specifies if the submitted result will open in a new browser tab, a frame, or in the current window.
- The default value is "_self" which means the form will be submitted in the current window.
- To make the form result open in a new browser tab, use the value "_blank":
- Example
- **<form action="/action_page.php" target="_blank">**
- Other legal values are "_parent", "_top", or a name representing the name of an iframe.

The method Attribute

- This attribute for the <form> tag sets the method by which the browser sends the form's data to the server for processing. There are two ways: the POST method and the GET method. If method is not specified, GET is used. Example
- <form action="/action_page.php" **method="get"**>
  or:
  Example
- <form action="/action_page.php" **method="post"**>

The POST and GET method

- With the POST method, the browser sends the data in two steps: the browser first contacts the forms-processing server specified in the action attribute and then, once contact is made, sends the data to the server in a separate transmission.

- The GET method, on the other hand, contacts the forms-processing server and sends the form data in a single transmission step: the browser appends the data to the form's action URL, separated by the question mark character.

- Which one should you use if your forms-processing server supports both the POST and GET methods? Here are some rules of thumb:

    - For best form-transmission performance, send small forms with a few short fields via the GET method.

    - Because some server operating systems limit the number and length of command-line arguments that can be passed to an application at once, use the POST method to send forms that have many fields or that have long text fields.

- If you are inexperienced in writing server-side forms-processing applications, choose GET. The extra steps involved in reading and decoding POST-style transmitted parameters, while not too difficult, may be more than you are willing to tackle.

- If security is an issue, choose POST. GET places the form parameters directly in the application URL, where they easily can be captured by network sniffers or extracted from a server logfile. If the parameters contain sensitive information like credit card numbers, you may be compromising your users without their knowledge. While POST applications are not without their security holes, they can at least take advantage of encryption when transmitting the parameters as a separate transaction with the server.

Name Attribute

- Each input field must have a name attribute to be submitted.

- If the name attribute is omitted, the data of that input field will not be sent at all.
- This example will only submit the "Last name" input field:
- Example
-
```
<form action="/action_page.php">
  First name:<br>
  <input type="text" value="Nicole"><br>
  Last name:<br>
  <input type="text" name="lastname"        value="Pasqual"><br><br>
  <input type="submit" value="Submit">
</form>
```

HTML Input Types

<u>Input Type Text</u>

<input type="text"> defines a one-line text input field.

Example:

```
<form>
  First name:<br>
  <input type="text" name="firstname"><br>
  Last name:<br>
  <input type="text" name="lastname">
</form>
```

<u>Input Type Password</u>

<input type="password"> defines a password field:

Example

```
<form>
  User name:<br>
  <input type="text" name="username"><br>
  User password:<br>
  <input type="password" name="psw">
</form>
```

<u>Input Type Submit</u>

- <input type="submit"> defines a button for **submitting** form data to a **form-handler**.
- The form-handler is typically a server page with a script for processing input data.
  Example

-
```
<form action="/action_page.php">
  First name:<br>
  <input type="text" name="firstname" value="Nicole"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Pasqual"><br><br>
  <input type="submit" value="Submit">
</form>
```

Input Type Reset

- <input type="reset"> defines a reset button that will reset all form values to their default values.
- Example
-
```
<form action="/action_page.php">
  First name:<br>
  <input type="text" name="firstname" value="Nicole"><br>
  Last name:<br>
  <input type="text" name="lastname" value="Pasqual"><br><br>
  <input type="submit" value="Submit">
  <input type="reset">
</form>
```

Radio Buttons

- Radio button form controls are similar in behavior to checkboxes, except that the user can select only one in the group.
- Create a radio button by setting the type attribute of the <input> tag to radio. As with checkbox controls, radio buttons each require a name and value attribute. Radio buttons with the same name are members of a group. One of them may be checked by including the checked attribute with that element. If you don't check one in the group, the browser does it automatically for you by checking the first element in the group.

Input Type Radio

<input type="radio"> defines a radio button.

Radio buttons let a user select ONLY ONE of a limited number of choices:

Example

```
<form>
  <input type="radio" name="gender" value="male" checked> Male<br>
  <input type="radio" name="gender" value="female"> Female<br>
  <input type="radio" name="gender" value="other"> Other
</form>
```

Checkboxes

- The checkbox form control gives users a way to select or deselect an item quickly and easily in your form. Checkboxes also may be grouped to create a set of choices, any and all of which the user may select or deselect.

Input Type Checkbox

<input type="checkbox"> defines a checkbox.

Checkboxes let a user select ZERO or MORE options of a limited number of choices.

Example

```
<form>
  <input type="checkbox" name="vehicle1" value="Bike"> I have a bike<br>
  <input type="checkbox" name="vehicle2" value="Car"> I have a car
</form>
```

Input Type Button

- <input type="button"> defines a **button**:
- Example
- <input type="button" onclick="alert('Hello World!')" value="Click Me!">

<u>HTML5 Input Types:</u> HTML5 added several new input types:

- color
- date
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

New input types that are not supported by older web browsers, will behave as <input type="text">.

EXAMPLE:

<u>Input Type Color</u>

The <input type="color"> is used for input fields that should contain a color.
Depending on browser support, a color picker can show up in the input field.
Example

```
<form>
  Select your favorite color:
  <input type="color" name="favcolor">
</form>
```

HTML Input Attributes

<u>The value Attribute</u>

The value attribute specifies the initial value for an input field:

Example

```
<form action="">
First name:<br>
<input type="text" name="firstname" value="Nicole">
</form>
```

The readonly Attribute

The readonly attribute specifies that the input field is read only (cannot be changed):

Example

```
<form action="">
First name:<br>
<input type="text" name="firstname" value="Nicole" readonly>
</form>
```

The disabled Attribute

The disabled attribute specifies that the input field is disabled.

A disabled input field is unusable and un-clickable, and its value will not be sent when submitting the form:

Example

```
<form action="">
First name:<br>
<input type="text" name="firstname" value="Nicole" disabled>
</form>
```

The size Attribute

The size attribute specifies the size (in characters) for the input field:

Example

```
<form action="">
First name:<br>
<input type="text" name="firstname" value="Nicole" size="40">
</form>
```

The maxlength Attribute

The maxlength attribute specifies the maximum allowed length for the input field:

Example

```
<form action="">
First name:<br>
<input type="text" name="firstname" maxlength="10">
</form>
```

HTML5 Attributes

HTML5 added the following attributes for <input>:

- autocomplete
- autofocus
- form
- formaction

- formenctype
- formmethod
- formnovalidate
- formtarget
- height and width
- list
- min and max
- multiple
- pattern (regexp)
- placeholder
- required
- step

and the following attributes for <form>:

- autocomplete
- novalidate

- An HTML form with **autocomplete** on (and off for one input field):
- <form action="/action_page.php" autocomplete="on">

```
  First name:<input type="text" name="fname"><br>
  Last name: <input type="text" name="lname"><br>
  Email: <input type="email" name="email" autocomplete="off"><br>
  <input type="submit">
</form>
```

The <select> Element

The <select> element defines a drop-down list:

Example

```
<select name="Movies">
 <option value="Ten">Ten Things I Hate About You
        </option>
 <option value="Fifty">Fifty First Dates</option>
 <option value="Dreams">What Dreams May Come
              </option>
</select>
```

- The <option> elements defines an option that can be selected.
- By default, the first item in the drop-down list is selected.
- To define a pre-selected option, add the selected attribute to the option

Example

<option value="fiat" selected>Fiat</option>

Visible Values:

Use the size attribute to specify the number of visible values:

Example

<select name="cars" size="3">

Allow Multiple Selections:

- Use the multiple attribute to allow the user to select more than one value:
- Example
- <select name="cars" size="4" multiple>

The <textarea> Tag

- the <textarea> tag creates a multiline text-entry area in the user's browser display. In it, the user may type a nearly unlimited number of lines of text. Upon submission of the form, the browser collects all the lines of text, each separated by %0D%0A (carriage return/line feed), and sends them to the server as the value of this form element, using the name specified by the required name attribute.
- The <textarea> element defines a multi-line input field (a text area):
- Example
- <textarea name="message" rows="10" cols="30">
  The cat was playing in the garden.
  </textarea>
- The rows attribute specifies the visible number of lines in a text area.
- The cols attribute specifies the visible width of a text area.

- You can also define the size of the text area by using CSS:
- Example
- <textarea name="message" style="width:200px; height:600px">
  The cat was playing in the garden.
  </textarea>

Creating Effective Forms

- **Browser Constraints**
  - Make sure your forms assist users as much as possible. For example, adjust the size of text-input fields to give clues on acceptable input; five-character (or nine-character) zip codes, for instance. Use checkboxes, radio buttons, and selection lists whenever possible to narrow the list of choices the user must make.
- **Handling Limited Displays**
  - You should structure your form to scroll naturally into two or three logical sections. The user can fill out the first section, page down; fill out the second section, page down; and so forth.
  - You should also avoid wide input elements. It is difficult enough to deal with a scrolling text field or text area without having to scroll the document itself horizontally to see additional portions of the input element.
- **User-Interface Considerations**
  - A good form accommodates all three of these perceptive needs. Input elements should be organized in logical groups so that your brain can process the form layout in chunks of related fields. Consistent, well-written prompts and supporting text assist and lead the user to enter the correct information. Text prompts also remind users of the task at hand and reinforce the form's goal.
- **Creating Forms That Flow**
  - Your forms should lead the user naturally through the process of supplying the necessary data for the application. You wouldn't ask for a street address before asking for the user's name; other rules may dictate the ordering of other groups of input elements. To see whether your form really works, make sure you view it on several browsers and have several people fill it out and comment on its effectiveness.

References:

- Carey. P (2018). *New perspectives on HTML5, CSS3, and Javascript.* Australia : Cengage Learning
- Jenkins, S. (2013). Web Design All-in-One for Dummies. John Wiley & Sons.
- Parker, J. (2021). HTML for Beginners: A Complete Beginners Guide to Learn Html in 1 Hour and Master Your Web Designing.
- W3Schools online web tutorials. (n.d.). https://www.w3schools.com/
- Sklar, J. (2012). *Web Design Principles*.
- Castro, E., & Hyslop, B. (2013). *HTML and CSS: Visual QuickStart Guide*. Peachpit Press.
- Plumley, G. (2011). *Website design and development :100 questions to ask before building a website.* Indianapolis, IN : Wiley Pub.