Lecture No: 6
Topics: Forms and Multimedia Elements

Forms

Web Forms

- are interactive elements on web pages that allow users to input data, such as text, choices, or selections, and submit it for processing.

Parts of a Web Form

A form contains controls,also known as widgets,which are the objects that allow the user to interact with the form.HTML supports several types of controls and widgets,including;

Controls

- **input boxes** for inserting text strings and numeric values
- **option buttons**,also called radio buttons,for selecting data values from a small predefined set of options
- **selection lists** for selecting data values from a more extensive list of options
- **check boxes** for selecting data values limited to two possibilities,such as "yes"or"no"
- **text area boxes** for entering text strings that may include several lines of content

Widgets

- **spin boxes** for entering integer values confined to a specified range
- **slider controls** for entering numeric values confined to a specified range
- **calendar controls** for selecting date and time values
- **color pickers** for choosing color values

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)

☑ Text Input

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

Syntax:

<input type="text" id="text-box-identifier"

placeholder="user-entry-description"

size="box-width" maxlength="maximum-typed-characters">

Example

```
<form>
  <label for="fname">First name:</label><br>
  <input type="text" id="fname" name="fname"><br>
  <label for="lname">Last name:</label><br>
  <input type="text" id="lname" name="lname">
</form>
```

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

The <label> Element

- The <label> tag defines a label for many form elements.
- The <label> element is useful for screen-reader users, because the screen-reader will read out loud the label when the user focuses on the input element.
- The <label> element also helps users who have difficulty clicking on very small regions (such as radio buttons or checkboxes) - because when the user clicks the text within the <label> element, it toggles the radio button/checkbox.
- The for attribute of the <label> tag should be equal to the id attribute of the <input> element to bind them together.

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.
- Example

```
<form action="/action_page.php">
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" value="Nicole"><br><br>
        <input type="submit" value="Submit">
</form>
```

☑ **HTML Input Types**

Input Type Text

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

Example:

```
<form>
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname"><br>
        <label for="lname">Last name:</label><br>
        <input type="text" id="lname" name="lname">
</form>
```

Input Type Password

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

Example

```
<form>
        <label for="username">Username:</label><br>
        <input type="text" id="username" name="username"><br>
        <label for="pwd">Password:</label><br>
        <input type="password" id="pwd" name="pwd">
</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">

```
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname" value="Nicole"><br>
        <label for="lname">Last name:</label><br>
        <input type="text" id="lname" name="lname" value="Pascual"><br><br>
        <input type="submit" value="Submit">

      <input type="reset" value="Reset">
    </form>
```

Input Type 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"> defines a radio button.
- Radio buttons let a user select ONLY ONE of a limited number of choices:

Example

```
<form>
        <input type="radio" id=”male” name="gender" value="male" checked>
        <label for="male">MaleL</label><br>
        <input type="radio" id=”female” name="gender" value="female">
        <label for="female">Female</label><br>
        <input type="radio" id=”other” name="gender" value="other">
        <label for="other">Other</label><br>
</form>
```

<u>Input Type Checkboxes</u>

- 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"> defines a checkbox.

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

Example

```
<form>
  <input type="checkbox" id="vehicle1" name="vehicle1" value="Bike">
  <label for="vehicle1"> I have a bike</label><br>
  <input type="checkbox" id="vehicle2" name="vehicle2" value="Car">
  <label for="vehicle2"> I have a car</label><br>
  <input type="checkbox" id="vehicle3" name="vehicle3" value="Boat">
  <label for="vehicle3"> I have a boat</label>
</form>
```

<u>Input Type Submit Button</u>

- <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">
    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname" value="Nicole"><br>
    <label for="lname">Last name:</label><br>
    <input type="text" id="lname" name="lname" value="Pascual"><br><br>
    <input type="submit" value="Submit">
  </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>
  <label for="favcolor">Select your favorite color:</label>
  <input type="color" id="favcolor" name="favcolor">
</form>
```

☑ HTML Input Attributes

<u>The value Attribute</u>

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

Example

```
<form>
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname" value="Nicole"><br>
</form>
```

The readonly Attribute

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

Example

```
<form>
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname" 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>
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname" value="Nicole" disabled>
</form>
```

The size Attribute

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

Example

```
<form>
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname" size="50"><br>
</form>
```

The maxlength Attribute

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

Example

```
<form>
        <label for="fname">First name:</label><br>
        <input type="text" id="fname" name="fname" size="50" maxlength="50">
</form>
```

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

The <select> Element

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

Example

```
<select name="movies" id="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

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 id="cars" name="cars" size="4" multiple>

The <textarea> Tag

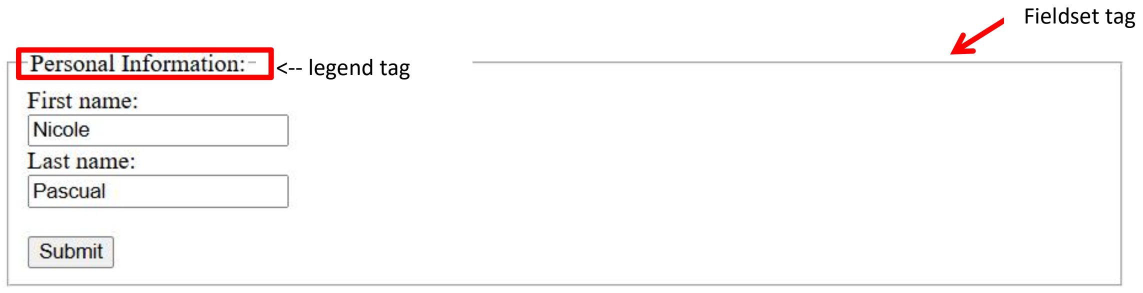
- the <textarea> tag creates a multiline text-entry area in the user's browser display. In it, the user may type a nearly unlimited number of lines of text. Upon submission of the form, the browser collects all the lines of text, each separated by %0D%0A (carriage return/line feed), and sends them to 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 id="message" 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 id="message" name="message" style="width:200px; height:600px">
  The cat was playing in the garden.
  </textarea>

The <fieldset> and <legend> Elements

- The <fieldset> element is used to group related data in a form.
- The <legend> element defines a caption for the <fieldset> element.
- Example:
-
```
<form action="/action_page.php">
  <fieldset>
    <legend>Personal Information:</legend>
    <label for="fname">First name:</label><br>
    <input type="text" id="fname" name="fname" value="Nicole"><br>
    <label for="lname">Last name:</label><br>
    <input type="text" id="lname" name="lname" value="Pascual"><br><br>
    <input type="submit" value="Submit">
  </fieldset>
</form>
```



-

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.

HTML Multimedia

- Multimedia on the web is sound, music, videos, movies, and animations.
- Web pages often contain multimedia elements of different types and formats.

Browser Support

- The first web browsers had support for text only, limited to a single font in a single color.
- Later came browsers with support for colors and fonts, and images!
- Audio, video, and animation have been handled differently by the major browsers. Different formats have been supported, and some formats require extra helper programs (plug-ins) to work.

Multimedia Formats

- Multimedia elements (like audio or video) are stored in media files.
- The most common way to discover the type of a file, is to look at the file extension.
- Multimedia files have formats and different extensions like: .swf, .wav, .mp3, .mp4, .mpg, .wmv, and .avi

AUDIO

Audio Formats

Format	File	Description
MIDI	.mid .midi	MIDI (Musical Instrument Digital Interface). Main format for all electronic music devices like synthesizers and PC sound cards. MIDI files do not contain sound, but digital notes that can be played by electronics. Plays well on all computers and music hardware, but not in web browsers.
AAC	.aac	AAC (Advanced Audio Coding). Developed by Apple as the default format for iTunes. Plays well on Apple computers, but not in web browsers.
Ogg	.ogg	Ogg. Developed by the Xiph.Org Foundation. Supported by HTML5.
MP3	.mp3	MP3 files are actually the sound part of MPEG files. MP3 is the most popular format for music players. Combines good compression (small files) with high quality. Supported by all browsers. MP3 is the abbreviation for 'MPEG Audio Layer III' ('MPEG' stands for 'Motion Pictures Expert Group')
MP4	.mp4	MP4 is a video format, but can also be used for audio. MP4 video is the upcoming video format on the internet. This leads to automatic support for MP4 audio by all browsers.

The HTML <audio> Element

- Before HTML5, audio files could only be played in a browser with a plug-in (like flash).
- The HTML5 <audio> element specifies a standard way to embed audio in a web page.
- Example:
- <audio controls>
-   <source src="RememberMe.mp3" type="audio/mpeg">
-   Your browser does not support the audio element.
- </audio>

controls attribute

- The controls attribute adds audio controls, like play, pause, and volume.
- The <source> element allows you to specify alternative audio files which the browser may choose from. The browser will use the first recognized format.
- The text between the <audio> and </audio> tags will only be displayed in browsers that do not support the <audio> element

HTML Audio - Browser Support

Browser	**MP3**	**Wav**	**Ogg**
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	YES	NO

VIDEOS

Video Formats

- **MP4 is the format for internet video.**
- **MP4 is recommended by YouTube.**
- **MP4 is supported by Flash Players.**
- **MP4 is supported by HTML5**

Format	File	Description
Ogg	.ogg	Theora Ogg. Developed by the Xiph.Org Foundation. Supported by HTML5.
WebM	.webm	WebM. Developed by the web giants, Mozilla, Opera, Adobe, and Google. Supported by HTML5.
MPEG-4 or MP4	.mp4	MP4. Developed by the Moving Pictures Expert Group. Based on QuickTime. Commonly used in newer video cameras and TV hardware. Supported by all HTML5 browsers. Recommended by YouTube.

The HTML <video> Element

- The HTML5 <video> element specifies a standard way to embed a video in a web page.

Example:

- <video width="320" height="240" controls>
  <source src="Feast.mp4" type="video/mp4">
  Your browser does not support the video tag.
  </video>

controls attribute

- The controls attribute adds video controls, like play, pause, and volume.
- It is a good idea to always include width and height attributes. If height and width are not set, the page might flicker while the video loads.

- The <source> element allows you to specify alternative video files which the browser may choose from. The browser will use the first recognized format.
- The text between the <video> and </video> tags will only be displayed in browsers that do not support the <video> element.

HTML <video> Autoplay

- To start a video automatically use the autoplay attribute:
- <video width="320" height="240" autoplay>
    <source src="movie.mp4" type="video/mp4">
    Your browser does not support the video tag.
  </video>
- The autoplay attribute does not work in mobile devices.

HTML Video - Browser Support

- In HTML5, there are 3 supported video formats: MP4, WebM, and Ogg

Browser	MP4	WebM	Ogg
Internet Explorer	YES	NO	NO
Chrome	YES	YES	YES
Firefox	YES	YES	YES
Safari	YES	NO	NO

HTML Helpers (Plug-ins)

- Helper applications (plug-ins) are computer programs that extend the standard functionality of a web browser.
- Examples of well-known plug-ins are Java applets.
- Plug-ins can be added to web pages with the <object> tag or the <embed> tag.
- Plug-ins can be used for many purposes: display maps, scan for viruses, verify your bank id, etc.
- To display video and audio: Use the <video> and <audio> tags.

The <object> Element

- The <object> element is supported by all browsers.
- The <object> element defines an embedded object within an HTML document.
- It is used to embed plug-ins (like Java applets, PDF readers, Flash Players) in web pages.
  Example:
  <object width="400" height="200" data="banners.swf"></object>
- The <object> element can also be used to include HTML in HTML:
  Example:
  <object width="100%" height="500px" data="images1.html"></object>
- Or images if you like:
  Example:
  <object data="panda.png"></object>

The <embed> Element

- The <embed> element is supported in all major browsers.
- The <embed> element also defines an embedded object within an HTML document.
- Web browsers have supported the <embed> element for a long time. However, it has not been a part of the HTML specification before HTML5.
  Example:
  <embed width="400" height="50" src="banners.swf">
  - Note that the <embed> element does not have a closing tag. It cannot contain alternative text.
- The <embed> element can also be used to include HTML in HTML:
  Example:
  <embed width="100%" height="500px" src="images1.html">
- Or images if you like:

Example:

```
<embed src="panda.png">
```

References:

- Dean, J. (2018). *Web Programming with HTML5, CSS, and JavaScript*. https://openlibrary.telkomuniversity.ac.id/home/catalog/id/164829/slug/web-programming-with-html5-css-and-javascript.html
- Castro, E., & Hyslop, B. (2013). *HTML and CSS: Visual QuickStart Guide*. Peachpit Press.
- Carey. P (2018). *New perspectives on HTML5, CSS3, and Javascript*. Australia : Cengage Learning
- 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*.
- Plumley, G. (2011). *Website design and development :100 questions to ask before building a website.* Indianapolis, IN : Wiley Pub.