Lecture 4: XHTML Lists, Tables, Forms
Tables

Tables are not only used to present tabular information, but can be used as a presentation aid. They can also be used to create the structure of an entire web page

Example 1. Tabular information

Example 2. Presentation aid

Example 3. Structure of a web page
What is a Table?

• a matrix of horizontal rows and vertical columns
• intersection of a row and column is a cell

*Cells* can contain almost anything:
text, headings, horizontal rules, images, nested tables

```html
<table border="1">
  <tr>
    <td>row 1, cell 1</td>
    <td>row 1, cell 2</td>
  </tr>
  <tr>
    <td>row 2, cell 1</td>
    <td>row 2, cell 2</td>
  </tr>
</table>
```
Table Element

Starts with tag \(<table>\) ends with \(</table>\)

Rows begin with label \(<tr>\) end with \(</tr>\)

Columns begin with \(<td>\) end with \(</td>\)
  - these tags define a cell
  - must be inside the \(<tr>\)  \(</tr>\) tags

See simple table example
Some table attributes

<table> tag has optional attributes:

**width** in px or %

```html
<table width="100%">
<table width="300px">
```

**border** in px

```html
<table border="1">
```

There are many others that control things like alignment and cell spacing etc. Since these are better controlled through CSS we will not discuss them here. For complete list see w3schools.com or other online references.
Tables with headings

Use the `<th>` tag to indicate header cells

By default bold

Works much like a `<td>` tag

See example
<caption> element

caption is displayed on top of the table

<table>
  <caption>Scores</caption>
  <tr>
    <th>Name</th>
    <th>Score</th>
  </tr>
  <tr>
    <td>Joe</td>
    <td>80</td>
  </tr>
  <tr>
    <th>Average</th>
    <td>80</td>
  </tr>
</table>

See example
Cell Attributes

- **rowspan**
  - stretches a cell to occupy cells in rows below

- **colspan**
  - expands cell to occupy cells to the right

See example
What kind of content can cells contain?

Just about anything!

See example
Lists

- Ordered (Enumerated)
- Unordered (Bullet points)

Real life example: Our class website schedule page
Unordered Lists

An unordered list starts with the `<ul>` tag. Each list item starts with the `<li>` tag.

The list items are marked with bullets (typically small black circles)

• Great for creating bullet points!
• Used for lists where order does not matter!
• Can be nested

Example:
```
<h1>Unordered Lists</h1>
<ul>
  <li>Great for creating bullet points!</li>
  <li>Used for lists where order does not matter!</li>
  <li>Can be nested</li>
</ul>
```
Ordered lists

- Used to list items when order is important
- List items are automatically numbered

<ol>
  <li>Go to store</li>
  <li>Buy the food</li>
  <li>Make dinner</li>
</ol>

1. Go to the store
2. Buy the food
3. Make dinner
List Item `<li>` Elements

- are block elements so each will display starting on a new line
- may contain either block level or inline elements, or both
- may contain text, other lists, or both as content (nesting)

See example
Forms

• allow users to interact with your site by
  – collecting data
  – responding

Forms can have several features (widgets):
  – checkboxes
  – radio buttons
  – text fields
  – password fields
  – file selection mechanisms
  – text areas
  – scrolling menus
  – buttons
Forms

• start with a `<form>` block tag

  - `action` required attribute (for now "" value)
  - `method` attribute “get” or “post”

```html
<form action="" method="post">
  <p>
    <input type="password" name="mypasswd"/>
    <input type="checkbox" name="mempwd" value="remember" checked="checked"/>
    Remember Password
  </p>
</form>
```
Method and post attributes

```html
<form method = "post" action = "">
or
<form method="get" action="">

method specifies how data is sent to the server
- post sends the information without user being able to see it
- get appends the information to the URL

action specifies the URL of the script on the server that will process the form data
The input element

The most important form element is the input element.

Controls (widgets) specified by `input` tag and `type` attribute

Syntax:
```html
<input type="control type" att1="att value" att2="att value"
... />
```

Possible control types:
text field,
password,
radio button,
checkbox,
file
hidden
image
submit button
Text fields

Syntax:
<input type="text" name="some_name" />

Attributes:
size - width of the field
maxlength - max number of characters
name - identifier
value - default value.

Default width of a text field is 20 characters

Example:
<form method="get" action="">
  <p>
    First name: <input type="text" name="firstname" />
    <br />
    Last name: <input type="text" name="lastname" />
  </p>
</form>
Password fields

Syntax:

```html
<input type="password" name="pw" />
```

When you want to hide user input

Used mostly for password entry boxes, credit cards and other account numbers

Has same attributes as text field
Submit button

A submit button is used to send form data to a server. The data is sent to the page specified in the form's action attribute.

```html
<input type="submit" value="Send" />
```

value attribute sets the text on the button.
Radio Buttons (only one can be selected in a group)

Important attributes are **name** and **value**

**name** identifies the radio button group  
**value** identifies a specific radio button - This is the value sent to the form's action URL

Optional attribute:  
checked - designates which radio button is initially selected

**Example:**

```html
<form>
  <p>
    <input type="radio" name="sex" value="male" />Male<br />
    <input type="radio" name="sex" value="female" />Female
  </p>
</form>
```
Check Boxes

Each has a checked or unchecked state

Usually a group of related check boxes have same name

Example:

<p>This summer I plan to (check all that apply)</p>
<form method="get" action="">
  <p>
    <input type="checkbox" name="summer" value="Travel"/> Travel
    <input type="checkbox" name="summer" value="Exercise"/> Exercise
    <input type="checkbox" name="summer" value="Study"/> Study
    <input type="checkbox" name="summer" value="Nothing"/> Do nothing!
  </p>
</form>
Reset button

Resets form widgets to their default

<input type="reset" />
Buttons

Custom buttons

<input type="button" value="Click me" onclick="msg()" />

Creates a button with text in value attribute
Runs a javascript function specified in the onclick attribute

Buttons with images

<input type="image" src="../Images/s.gif"
    alt="happy sender" name="happypic" value="Send" />

• src attribute is the url of the image
• alt is the alt text of the image
• name is the identifier
• value is default data sent to the server
The `<textarea>` tag defines a multi-line text input control

A text area can hold an unlimited number of characters, and the text renders in a fixed-width font (usually Courier).

The size of a textarea can be specified by the cols and rows attributes, or even better; through CSS' height and width properties.

Unlike the text type for input, there is no value attribute

```html
<textarea name="essay" rows="10" cols="80">
Default Text
</textarea>
```
Scrolling menu

Starts with a `<select>` tag (ends with `</select>`)

Choices are indicated by `<option>` elements

`<option>` tag has optional attribute `selected` that indicates what item is by default selected.

The choices can be broken down into groups by using `optgroup` tag. The `optgroups` get their names from a label attribute. See example.

Example:

```
<select>
  <optgroup label="Swedish Cars">
    <option value="volvo">Volvo</option>
    <option value="saab">Saab</option>
  </optgroup>
  <optgroup label="German Cars">
    <option value="mercedes">Mercedes</option>
    <option value="audi">Audi</option>
  </optgroup>
</select>
```
Sometimes we want to brake a form into sections.

We can use fieldset element to do this.

legend element creates a caption for the fieldset element.

```html
<form>
  <fieldset>
    <legend>Buyers info</legend>
    <!-- ...Buyers info form... -->
  </fieldset>
  <fieldset>
    <legend>Shipping info</legend>
    <!-- ...Shipping form... -->
  </fieldset>
</form>
```
In general text has to be always contained inside an "appropriate element". Most often this is a <p>. To be able to label your input elements inside a form, you should put the text either inside a <p> or use the <label> element.

<form action="" method="get">

    <label>My text field</label><input type="text" />

</form>
Big example

See examples page for everything we have learned put together.
# Reference for input elements attributes

<table>
<thead>
<tr>
<th>Attribute</th>
<th>Value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>accept</td>
<td>MIME_type</td>
<td>Specifies the types of files that can be submitted through a file upload (only for type=&quot;file&quot;)</td>
</tr>
<tr>
<td>alt</td>
<td>text</td>
<td>Specifies an alternate text for an image input (only for type=&quot;image&quot;)</td>
</tr>
<tr>
<td>checked</td>
<td>checked</td>
<td>Specifies that an input element should be preselected when the page loads (for type=&quot;checkbox&quot; or type=&quot;radio&quot;)</td>
</tr>
<tr>
<td>disabled</td>
<td>disabled</td>
<td>Specifies that an input element should be disabled when the page loads</td>
</tr>
<tr>
<td>maxlength</td>
<td>number</td>
<td>Specifies the maximum length (in characters) of an input field (for type=&quot;text&quot; or type=&quot;password&quot;)</td>
</tr>
<tr>
<td>name</td>
<td>name</td>
<td>Specifies a name for an input element</td>
</tr>
<tr>
<td>readonly</td>
<td>readonly</td>
<td>Specifies that an input field should be read-only (for type=&quot;text&quot; or type=&quot;password&quot;)</td>
</tr>
<tr>
<td>size</td>
<td>number</td>
<td>Specifies the width of an input field</td>
</tr>
<tr>
<td>src</td>
<td>URL</td>
<td>Specifies the URL to an image to display as a submit button</td>
</tr>
<tr>
<td>type</td>
<td></td>
<td>Specifies the type of an input element</td>
</tr>
<tr>
<td>value</td>
<td>value</td>
<td>Specifies the value of an input element</td>
</tr>
</tbody>
</table>