SIENA*college*
Computer Science

Computer Science 180
Web Design
Siena College
Fall 2011

Topic Notes: HTML Form Elements

Our next topic involves our pages to accept user input and to be more interactive. There are a number of tools for this, and we will focus on the most common: JavaScript.

Before we do this, we take a look at a group of HTML elements we have ignored so far, the *form*-related elements.

Forms in HTML are what allow us to have the familiar buttons and menus and input fields we see on so many pages.

The name comes from the idea that these elements are used to fill out a "form" much like you'd fill out a form on a sheet of paper. Once the form is filled out, it is submitted for processing. The same idea applies here, but we will put aside the idea of "processing" the form for the moment, and look at how to add these elements to our pages, then how to interact with them using JavaScript.

When used as part of a form that will be submitted, the tags should be inside a `<form>` element. Even though we will not be submitting our forms initially, it is a good idea to place our form-related elements within a `<form>` ... `</form>` block:

```
<form action="">
 ...
</form>
```

The `action` attribute is required; it is expected to be the URI of a script that will be used to process the form data. But for now, since we will not be submitting form data, we can provide an empty value.

The `<form>` element is just a container – to do anything useful, we need to add *form controls* to it.

Push Buttons

The `<button>` element, unsurprisingly, creates a push button. By default, the button contains whatever text is inside the element:

```
<button type="button">Push Me!</button>
```

Would create a push button labeled "Push Me!". The `type="button"` creates a "clickable" button but with no default action (i.e., we need to do more if we want it to do anything besides sit there looking nice). Other types are `submit`, which is a button used to submit a form, and `reset`, which clears the form data. We'll see these later.

Input Controls

A number of common controls can be defined using the `<input>` element. The `type` attribute determines the type of control desired.

The most common types:

- `checkbox` – defines a box that can be "checked" (selected) or "unchecked" (unselected), where the `value` attribute specifies a label on the checkbox

```
<input type="checkbox" name="major" value="CS" />I am a CS Major<br />
```

- `radio` – defines a radio button, one of a group of buttons among which only one can be selected at a time (like an old car radio's presets)

```
<input type="radio" name="year" value="freshman" />Freshman<br />
<input type="radio" name="year" value="sophomore" />Sophomore<br />
<input type="radio" name="year" value="junior" />Junior<br />
<input type="radio" name="year" value="senior" />Senior<br />
```

  Among all radio buttons with the same `name` attribute, only one may be selected at a time.

  For both `checkbox` and `radio` buttons, specifying the attribute `checked="checked"` will initialize that button to be checked when the page loads.

  Note that the text when we create `checkbox` or `radio` type `<input>` elements is not clickable to select the element. We would often want to be able to do so. The `<label>` element allows this:

```
<label for="csmajor">I am a CS Major</label>
<input type="checkbox" name="major" value="CS" id="csmajor" /><br />
```

  The `id=` attribute of the `<input>` element must match the `for=` attribute of the corresponding `<label>`.

- `file` – defines a file selector, usually consisting of a "Browse..." button and a field where a file path can be specified

```
<input type="file" />
```

- `text` – defines a one-line text field where input can be typed

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

By default, a text field is drawn with room for 20 characters. This can be changed with the `size` attribute. The maximum length of the input allowed is unlimited, but can be limited with the `maxlength` attribute.

- `password` – defines a one-line text field where input can be typed, but the characters typed are hidden

An `<input>` element is also capable of making push buttons, much like those we've seen using the `<button>` element. These values of `type` create buttons:

- `button` – defines a standard clickable button

```
<input type="button" value="Push Me!" />
```

- `submit` – defines a button intended to submit a form for processing
- `reset` – defines a button to clear or reset a form to its original values
- `image` – define a button using an image to be a submit button

Usually, `<input>` elements contain a `name` attribute and/or an `id` attribute to allow us to access the values/settings of the control.

Select Lists

Select lists, or drop-down lists, are created using the `<select>` element. The items in the list are added with `<option>` elements within.

```
<select>
  <option value="yankees">Yankees</option>
  <option value="redsox">Red Sox</option>
  <option value="rays">Rays</option>
  <option value="bluejays">Blue Jays</option>
  <option value="orioles">Orioles</option>
</select>
```

An additional element, the `<optgroup>` can be used to put the options into labeled groups within the list.

```
<select>
  <optgroup label="Playoff Teams">
    <option value="yankees">Yankees</option>
    <option value="rays">Rays</option>
```

```
  </optgroup>
  <optgroup label="Others">
    <option value="redsox">Red Sox</option>
    <option value="bluejays">Blue Jays</option>
    <option value="orioles">Orioles</option>
  </optgroup>
</select>
```

Text Areas

An `<input>` element with `type="text"` is used to input text into a form, but that control is limited to a single line of text. Sometimes, we want to have a similar control, but allowing multiple lines of input. This is where the `<textarea>` element comes in.

```
<textarea rows="4" cols="40">
This text should be in a 4x40 text area!
</textarea>
```

The required attributes `rows` and `cols` specify the visible size of the `<textarea>`. If more text is specified than fits in the `<textarea>`'s visible area, browsers should provide a scroll bar.

Field Sets

When a page contains a number of controls, it is often helpful to group and label them. The `<fieldset>` element, along with the optional `<legend>` element allow us to do this.

The default behavior of this tag is to draw a box around the controls contained within. If there is a `<legend>` defined, its contents are placed on the line.