Intro Forms notes (and exercise)

The information for these notes came from Session 7.1, in MindTap Reader 7-1 to 7-8; book pp502 -524, (Carey 6). The information in the bottom half of the last page came from Session 7.3, MindTap Reader 7-21 to 7-21b; book p559-562.

Control elements – A form is made up of control elements, which correspond to the fields of the form. Some control elements are:

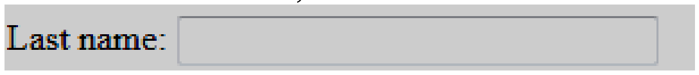
- textboxes (also called input boxes)
- radio buttons,
- checkboxes
- selection lists
- submit and reset buttons
- textarea

<form> element: The tags <form></form> go around the entire form. For example:

```
<form name="PerelForm" id="PerelForm" action="mailto:perel@ecc.edu" method="post">
        form elements go here
</form>
```

Attributes of the form element are name, id, action, and method. The name attribute is the name of the form, a meaningful name that you make up. The id attribute can take the same value as name's value. The action attribute describes how the form is processed, (the action performed on the form data). The method attribute describes the method used to send the form data. (See separate notes called Notes form action method.)

<input> element: Many of the control elements are created using the input element <input> tag. It is used to create **textboxes**, **radio buttons**, **checkboxes**, and the **submit** and **reset button**. (**textboxes** are also called **input boxes**, but I prefer the former.) The <input> tag has attributes type, name, value, and for textboxes, size and maxlength. The type attribute identifies which specific form element it is. For example, for a textbox, type="text" . See the picture and corresponding code below that renders a textbox for a last name field. Notice the text string (label) in front of the box that reads "Last Name",



the code is:

```
Last Name: <input type="text" name="LName" id ="LName" />,
```

Some values of the type attributes:

- for textboxes, **type="text"**
- for radio buttons, **type="radio"** (also called option buttons)
- for checkboxes, **type="checkbox"**
- for submit button, it is **type="submit"**
- for reset button, it is **type="reset".**

And some newer HTML5 data types for input boxes (textboxes) are:

- type="email"
- type="tel"
- type="url"

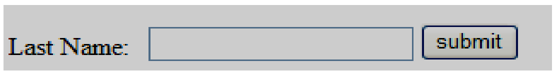
- type="date"
- type="time"

A textbox with type="email" requires the user to enter an email address. A textbox with type="tel" requires the user to enter a phone number, and so on. If the user does not enter what is expected, he might get an pop-up message. The pop-up messages vary by browser. See the chart in MindTap Reader 7-5a; book p511 (Carey 6) for many other values of the type attribute for the input element.

A "first example" of a form… with just a Last Name textbox, and submit button:

```
<form name="PerelForm" id="PerelForm" action="mailto:perel@ecc.edu" method="post">
  Last Name: <input type="text" name="LName" id="LName" size="20" /> <!-- line 1 -->
  <input type="submit" value="submit" />
</form>
```

Renders as **Diagram 1**:

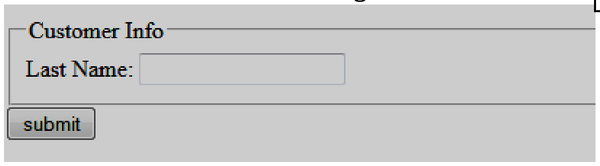


Some other elements. (MindTap Reader 7-4 & 7-6; book p507-509, & 514-516) Form elements, also called control elements are the fields of the form. Forms may contain elements such as <p> or <h1>, etc. Furthermore, there are form-related elements: **`fieldset`**, **`legend`** and **`label`**. For example, if you insert the three tags, <fieldset>, <legend> and <label> as in the code below, it renders as in the **Diagram 2**.

```
<form name="PerelForm" id="PerelForm" action="mailto:perel@ecc.edu" method="post">
 <fieldset>
     <legend>Customer Info</legend>
        <label>
        Last Name: <input type="text" name="LName" id="LName" size="20" />  <!-- line 1 -->
        </label>
   </fieldset>
   <input type="submit" value="submit" />
</form>
```

The label element surrounds the last name textbox code and its preceding text label to associate them with one another. (implicit label method)

The code above renders this **Diagram 2**:



`fieldset` renders as a border around the control elements of the form. A fieldset is used to group similar fields together. For example, in Tutorial 7 all Customer Information fields are placed together in one fieldset, and "Share your experience" fields are placed in a separate fieldset.

`legend` –places a title at the top of the fieldset. The <legend> element code goes just below the opening <fieldset> tag.

And **`label`** *associates* text label: Last Name: with its textbox:

[Aside:Whether the <label> tag is present or not, the rendered page looks the same. But first of all, the idea is that one can then reference **`label`** in a CSS style rule to apply styles to it. And second of all, one can click anywhere in the area of the field to select it. That is, if no label, one must click directly on the form element which is a "smaller target" that anywhere in its vicinity. Also this makes the code accessible.]

There are 2 ways to use the **`label`** element, implicitly and explicitly:

-- implicit association between the text label and the control:

```
<label>
Last Name: <input type="text" name="LName" id="LName" size="20" maxlength="30" />
</label>
```

-- explicit association between the text label and the control:

```
<label for="LName">Last Name:</label>
<input type="text" name="LName" id="LName" size="20" />
```

Note: In the explicit label, the **for** attribute is set equal to the id name of the control element with which it is associated.

Other form elements:

- **radio buttons** – also called **option buttons**. Round buttons. User can select one only.
- **checkboxes** – self-explanatory. User can check one, or more or none of the boxes.
- **selection list** - takes the form of a drop-down list but could be an open list. The term *selection list* is appropriate because it uses the <select> tag (It uses <option> tags too.)
- **text area** created with the <textarea> tag. A larger type of textbox for the user comments.

Refer to **Reference chart for form elements** document for pictures of each of these form elements and how to code each one.

(CSS for forms will not be on the test.)

Exercise to try:

1. Use a blank HTML5 document template with Komodo Edit or other editor. Save it as **first-form.htm** to your desired location. Inside of <body></body>, type the code that rendered **Diagram 2** on the previous page. Verify that it renders as in **Diagram 2**. Then try completing and submitting the form to see what happens.

2. Continue with a textbox for phone:

```
<label>
Phone: <input type="tel" name="phone" id="phone" size="12" maxlength="12" />
</label>
```

3. Save and render it to see what it looks like.

4. Add class="blockLabel" to the inside of <label> tag for each of Last name textbox and Phone textbox. That is:

```
<label class="blockLabel">
```

5. Create a new CSS file called **formcss.css file** and save it in the same location as **first-form.htm**. That is, in Komodo Edit, go to File > New > File for Template…, and select CSS and name it. Then link the CSS file to **first_form.htm**. That is, insert the following in between <head> and </head> of **first-form.htm**:

```
    <link href="formcss.css" rel="stylesheet" />
```

6. Add the following to the CSS file and resave the file:

```
    .blockLabel {display: block}
```
[1]

7. Resulting code:

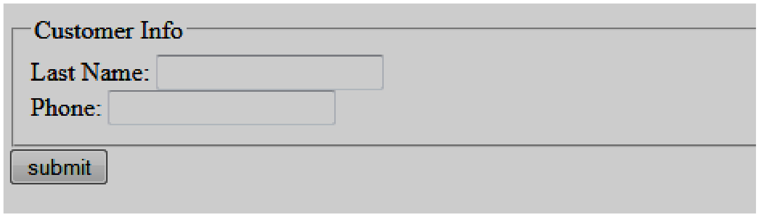
```
  <form name="PerelForm" id="PerelForm" action="mailto:perel@ecc.edu"    method="post">
    <fieldset>
      <legend>Customer Info</legend>

        <label class="blockLabel">
        Last Name:   <input type="text" name="LName" id="LName" size="20" maxlength="30" />
        </label>

        <label class="blockLabel">
        Phone:   <input type="tel" name="phone" id="phone" size="12" maxlength="12" />
        </label>

    </fieldset>
    <input type="submit" value="submit" />
  </form>
```

8. Verify that the form now renders where each of the Last Name and Phone controls is now a "block-level" element. That is, they each start on a new line. Try completing and submitting the form.



HTML5 langauge for form validation:

In CS103, we do not do server-side programming and thus will not write any programs to process the form data. Nonetheless, there are some client-side tools (attributes) we can use to try to check that the data is valid or is what we want before the form gets submitted to a server to be processed.

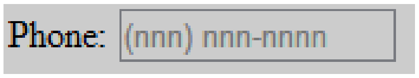
The following attributes can be used to guide the user to input data into the form in a format that the creator of the form prefers. The user may receive a pop-up message if he does not enter the proper format for the data type given. He may be restricted in what he can enter or get a control "widget" such as a calendar pop-up if the type is date. Refer to MindTap Reader 7-8; book p523-524 (Carey 6) for #1 and 2 below.

[1] Or label.blockLabel {display: block}

1. **placeholder attribute** – The placeholder attribute gives the user a hint as to how he should enter the data, for example, a phone number. The value for this attribute should be an expression that shows the user what format to use. The following example shows the code and how it renders:

Phone: <input type="text" name="phone" id="phone" size="14" **placeholder="(nnn) nnn-nnnn"** />

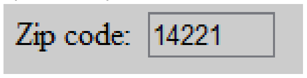
renders as:



2. **value attribute** – A default value for a field is a value that the file is already assinged, It appears in the form before the user enters anything. The user can type over it and give it another value, if necessary, using the value attribute. The following example shows the code and how it renders:

<input name="zip" id="zip" **value="14221"** size="5"/>

renders as:



3. **type attribute** for textboxes (i.e. input boxes) – As mentioned on page 1 of these notes, the textbox (input element) can contain not only type="text", general text, but more specifically it can hold other data types such as the following:

type="*type-value*" where ***type-value*** can be **email, url, date, number, date, time** and others.

When a form control has one of these types and the user incorrectly enters the format needed, he may get alerted with a pop-up message instructing him on how to enter the data correctly. See an example of this with type="email" in the picture in Session 7.3 MindTap Reader 7-21b; book p562 (Carey 6e).

4. **required attribute** – If a form element contains attribute **required="*required*",** (or just **required**) (MindTap Reader 7-21a, book p559-560, Carey 6e), the user will be prompted with a pop-up message if he submits the form without completing the required field.

5. **maxlength attribute** – The maxlength attribute can be used to restrict the number of characters the user is allowed to type into textboxes and so is helpful for data validation too. Examples:

State: <input type="text" name="state" id="state" maxlength="2" />
State: <input type="text" name="zipcode" id="zipcode" maxlength="5" />