CIS 4004: Web Based Information Technology Fall 2012

Inside HTML5 – Part 4 - Forms

Instructor :   Dr. Mark Llewellyn
                    markl@cs.ucf.edu
                    HEC 236, 407-823-2790
            http://www.cs.ucf.edu/courses/cis4004/fall2012

Department of Electrical Engineering and Computer Science
University of Central Florida

Inside HTML5 – Part 4 - Forms

- One of the biggest problems with XHTML and HTML 4 forms was that they were just dumb fields. Validation was required not only on the server-side, but also you had to duplicate the validation in the user's browser using JavaScript if you wanted to give the user the seamless experience they deserve.

- Given that almost every web page has some kind of form – search, comments, sign-up, and so on, HTML5 has come to the rescue and provides built-in validation for many of the most common types of data entered via web forms.

Inside HTML5 – Part 4 - Forms

- HTML5 makes developing forms quicker. There are some nice goodies like the addition of two new HTTP types for form action (`update` and `delete`) to go along with the current `get` and `post` actions.

- The most useful new features and the new form input types which provide special user interfaces and built-in error reporting.

- Eventually, you won't need JavaScript validation at all for these fundamental data types. Until everyone is using an HTML5 browser, some JavaScript validation will still be required.

Inside HTML5 – Part 4 - Forms

- It was actually the new form fields that were at the basis of the specification that ultimately became HTML5. It is also in this area where you'll see the principle of specifying backwards-compatible extensions to HTML5 in action.

- The extensions are largely new values of the `type` attribute of the `input` element.

- XHTML and HTML4 specified that browsers should assume `<input type=text>` if you don't specify a `type` attribute, or you use an unknown type. Therefore, legacy browsers that don't understand the new extensions will fall back to the default and will allow the user to enter data in a plain text field. This fallback can be detected in JavaScript and handled so that old browsers can mimic the new behaviors.

Inside HTML5 – Part 4 - Forms

- The HTML5 specification makes no requirements on how browsers should present the new input types to the user or report errors (handle validation). Different browsers and different devices will present different user interfaces, as we will see.

- The manner in which the browser reports errors is also undefined by the HTML5 specification. Error messages are part of the browser and are thus automatically localized. This means much less work for the developer and a more usable experience for the consumer.

Inside HTML5 – Part 4 - Forms

- The HTML5 specification makes no requirements on how browsers should present the new input types to the user or report errors (handle validation). Different browsers and different devices will present different user interfaces, as we will see.

- The manner in which the browser reports errors is also undefined by the HTML5 specification. Error messages are part of the browser and are thus automatically localized. This means much less work for the developer and a more usable experience for the consumer.

Inside HTML5 – Part 4 - Forms

- There are two basic parts to any form: the collection of fields, labels, and buttons that the visitor sees on the page and hopefully fills out; and the processing script that takes that information and converts it into a format that you can read or tally, or process in some manner perhaps on a database.

- Constructing a form's fields and buttons is straightforward and similar to creating any other part of the web page. Some of these form field types include text boxes, special password boxes, radio buttons, checkboxes, drop-down menus, large text areas, and even clickable images.

- Each element has a name that will serve as a label to identify the data once it is processed. CSS is used to style the form.

Inside HTML5 – Part 4 - Forms

- Using forms often requires using a server-side scripting language to receive the submitted information. It requires code on the web server that listens for form responses and processes the information in the response by storing information in a database, sending it in an email, or redirecting the user to new information.

- PHP is often the server-side scripting language that is utilized for this purpose, but there are many other server-side scripting languages. We'll cover PHP in more depth later in the course.

- Before we look at some of the new HTML5 form elements, let's examine creating basic forms and processing them.

Inside HTML5 – Part 4 - Forms

- Every form has 3 basic parts:

  - The `form` element.

  - The actual form elements where the visitor enters information.

  - The submit button that sends the collected information to the server.

- The form element includes the URL of the script that will process the form and its method (get, post, etc.).

- The example on the following page is a snippet of markup from a much larger form that we'll examine later. This example shows only a very simple form.

File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   Plugins   Window   ?   X

ResultSetTableModel.java | Ch08_LargeCo_SQL.txt | mini form 1.html | form.css | form.html

The form element

C:\Courses\CIS 4004 - Web-Based Info Tech\Fall 2012\code\Inside HTML5 - Part 4 - Forms\mini form 1.html

```html
 9  <div id="wrapper">
10      <h1>Create a New Account</h1>
11      <form method="post" action="showform.php">
12          <!-- START ACCOUNT FIELDS -->
13          <fieldset>
14              <h2 class="account">Account</h2>
15              <ul>
16                  <li>
17                      <label for="first_name">First Name:</label>
18                      <input type="text" id="first_name" name="first_name" class="large" required
19                  </li>
20                  <li>
21                      <label for="last_name">Last Name:</label>
22                      <input type="text" id="last_name" name="last_name" class="large" />
23                  </li>
24                  <li>
25                      <label for="email">Email:</label>
26                      <input type="email" id="email" name="email" class="large" />
27                  </li>
28              </ul>
29          </fieldset>
30          <!-- end account -->
31          <fieldset>
32              <input type="submit" class="create_profile" value="Create Account" />
33          </fieldset>
34      </form>
35  </div>
```

URL of the PHP script to process data

Submit button to send data to script

Hyper Text Markup Language file   length : 1000   lines : 37   Ln : 30   Col : 29   Sel : 0   UNIX   ANSI   INS

Rendering of the form in Safari with no CSS applied

The submit button

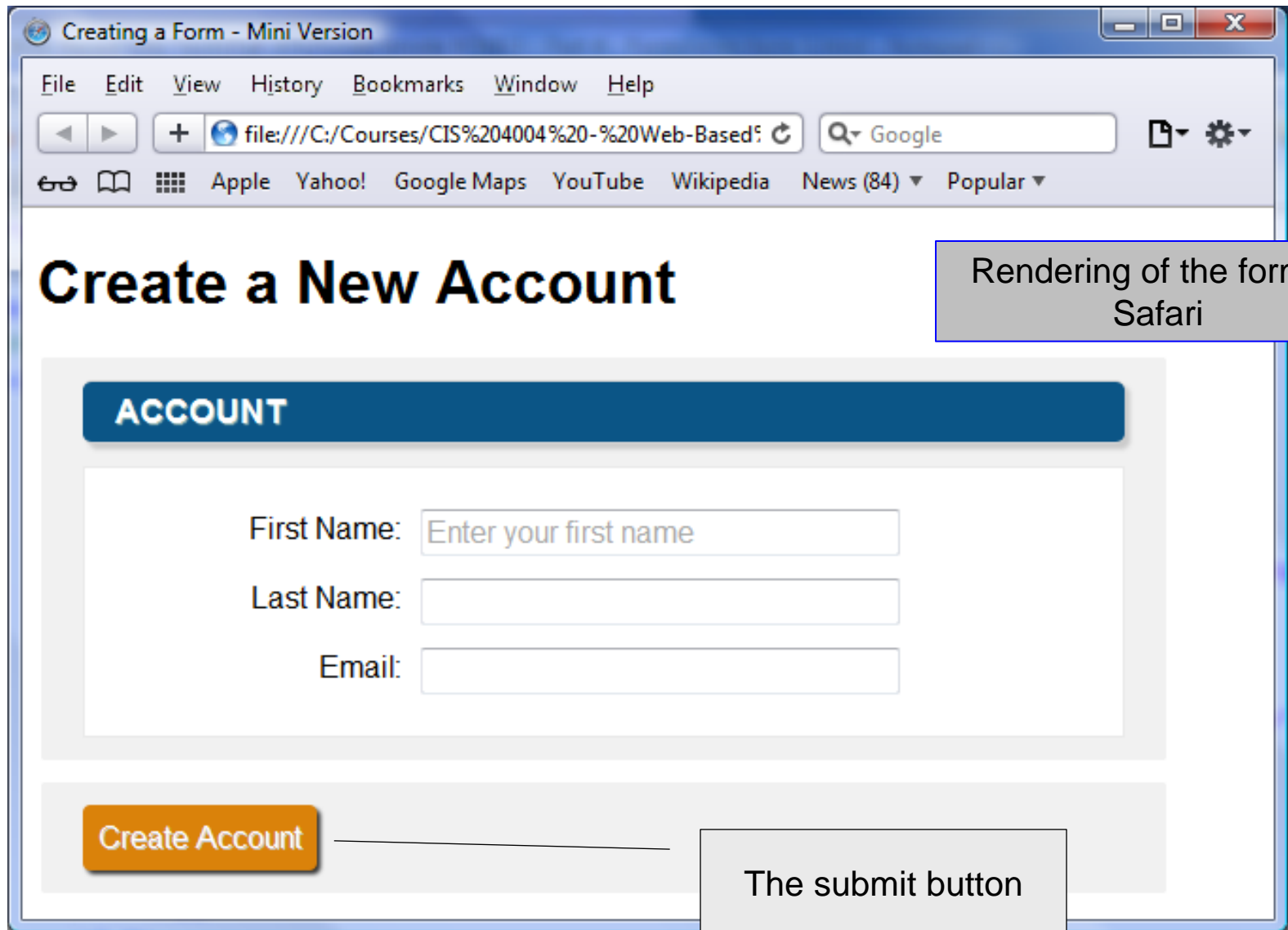File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   Plugins   Window   ?   X

form.html | showformdata.php | email form.html | emailformdata.php | form.css

```css
 1  body {
 2      font-size: 100%;
 3      font-family: Arial, sans-serif;
 4  }
 5
 6  #wrapper {
 7      width: 600px;
 8  }
 9
10  h2 {
11      background-color: #dedede;
12      border-bottom: 1px solid #d4d4d4;
13      border-top: 1px solid #d4d4d4;
14      border-radius: 5px;
15      box-shadow: 3px 3px 3px #ccc;
16      color: #fff;
17      font-size: 1.1em;
18      margin: 12px;
19      padding: 0.3em 1em;
20      text-shadow: #9FBEB9 1px 1px 1px;
21      text-transform: uppercase;
22  }
23
24  h2.account            { background-color: #0B5586; }
25  h2.address            { background-color: #4494C9; }
26  h2.public-profile     { background-color: #377D87; }
27  h2.emails             { background-color: #717F88; }
28
```

Part of the CSS to style
the form example

Cascade Styl length : 2105   lines : 147   Ln : 1   Col : 1   Sel : 0   UNIX

Java Update Available
A new version of Java is ready to be in

Rendering of the form in Safari

Create a New Account

ACCOUNT

First Name: [Enter your first name]

Last Name: [ ]

Email: [ ]

Create Account

The submit button

Output from the `showformdata.php` script.

Fieldsets and Legends

- Let's look at some of the basic form elements in the previous example and see how they can be styled and grouped to create a form that is both visually appealing as well as useful for your web page.

- If you have a lot of information to fill out on a form, you can use a `fieldset` element to group related elements and make the form easier to follow. The easier it is for your visitors to understand the form, the more likely they are to fill it out correctly.

- You can also use the `legend` element to give each `fieldset` a caption that describes the purpose of each grouping.   However, there is currently an almost universal lack of browser support for visual control over the `legend` element, so I recommend using regular heading elements.

Fieldsets and Legends

- The legend element limits the power of CSS to position it. It is this reason that its styling is limited in most browsers. I recommend that you use an appropriately styled `<p>` element or `<h1>-<h6>` element to recreate its effect. Notice that this is the approach I took in the previous example.

- Text boxes can contain one line of freeform text – that is, anything that the visitor wants to type – and they are typically used for names, addresses, and so on.

- There are many ways to separate your form elements from each other. In this example, I an using unordered lists, but it is also common to see `<div>`, `<p>` or `<br>` elements used in this context.

Textboxes

- Personally, I feel that the unordered list is more semantically correct for this as I would view a form as a list of elements to be completed by the visitor before they click the submit button.

- Text boxes can be different sizes to accommodate different types of fields. In the previous example, I used a CSS style sheet to set the width using classes. However, you can also set the width with the `size = "n"` attribute on the HTML5 element itself. I would normally take this approach as it is easier to make changes to a form in general via CSS than it is to go through the markup looking for size attributes in the various form elements.

- While every form element must have the `name` attribute set, the `value` attribute is optional and only needs to be specified if the form element is to have a default value.

Textboxes

- The `name` attribute must be specified for each element of your form. The reason is that this is how you access these values on the server to which the form data is sent for processing.

- Older versions of IE can get the `name` and `id` attributes confused when the script uses the `getElementByID` method. The way this is normally handled is to use the same unique value for both the `name` and `id` for each field in the form. This enhances the accessibility of your form. Notice that this is the approach I took in the previous example.

HTML5 Form Attribute Enhancements

- Forms are one area of HTML that many developers find painstaking, because it often requires extra effort with CSS and JavaScript to make them function well.

- HTML5 has made some significant improvements to forms by adding many form enhancements that were not available in HTML 4 or XHTML.

- These enhancements include attributes `autofocus`, `required`, `placeholder`, `maxlength`, and `pattern`. I'll introduce some of these as we go along with additional examples, but if you look at the complete code for the previous example (see course webpage for complete markup and next page for quick view), you'll see that both `required` and `placeholder` were used in text box for the first name.

```
</head>
<body>
<div id="wrapper">
        <h1>Create a New Account</h1>
        <form method="post" action="http://www.cs.ucf.edu/courses/cis4004/fall2012/
showformdata.php">
                <!-- START ACCOUNT FIELDS -->
                <fieldset>
                        <h2 class="account">Account</h2>
                        <ul>
                                <li>
                                        <label for="first_name">First Name:</label>
                                        <input type="text" id="first_name"
name="first_name" class="large" required="required" placeholder="Enter your first name" />
                                </li>
                                <li>
                                        <label for="last_name">Last Name:</label>
                                        <input type="text" id="last_name" name="last_name"
class="large" />
                                </li>
                                <li>
                                        <label for="email">Email:</label>
                                        <input type="email" id="email" name="email"
class="large" />
                                </li>
                        </ul>
                </fieldset>
                <!-- end account -->
                <fieldset>
                        <input type="submit" class="create_profile" value="Create
Account" />
                </fieldset>
        </form>
</div>
</body>
```

Placeholders are a great way to give visitors a hint or extra instructions for filling out the form. The placeholder attribute will put text in a light grey color inside your text box. When the user begins to input text in the file, the light grey text will disappear, and it will come back if the user leaves the field without entering any information.

Create a New Accou

ACCOUNT

First Name: Enter your first name

Last Name:

Email:

Create Account

HTML5 Form Attribute Enhancements

- The `required` attribute can be applied to any number of fields in a form. Any field that is designated as required must have a value entered in the field for the form to submit. If the user clicks the submit button and a required field has no data entered and error will be generated and the form data will not be submitted to the server.

- When desired, an element can have the `autofocus` attribute set to have a value of `autofocus`. If it is the first element to have this attribute, the input element will by default have focus when the page loads. Notice in the previous example, that no element had `autofocus` set, so the visitor was required to click on the first (or some) element to begin filling out the form. Notice the difference in the two renderings on the next page.

No autofocus on first element of form

Autofocus on first element of form

Password Boxes

- The only difference between a password box and a regular text box is that whatever is typed in the password box is hidden by bullets or asterisks. The information is not encrypted when it is sent to the server.

- A password box only keeps onlookers from seeing a user's password as it is typed. To really protect the password, a secure (https) server connection must be utilized.

Labeling Form Parts

- HTML5 provides a semantic method for marking up labels so that they are formally linked to the associated element. This makes them useable for scripting and other purposes.

- Notice in the example markup (see page 10) that each form element is inside a `label` element with a `for` attribute.

- For example, you might want to have "First Name" before the text field where the visitor should type their first name.

CAUTION ! ! !

Placeholders are sometimes incorrectly used as a replacement for the label. Be sure to use placeholders as hints to the visitor only.

Labeling Form Parts

- Using field labels in this semantically correct way gives you an easy way to identify them in a CSS style sheet.

- If you use the `for` attribute, you must also add the `id` attribute to the associated form element's start tag in order to mark it with a label. The document will not validate if you miss this step and use a label but do not include a matching id.

- Using the `for` attribute in the `label` requires that the label match the `id` attribute of the form element.

- Labels for radio buttons and checkboxes (more later) allow the user to click the label as well as the form element to modify the state. This is sometimes easier for the visitor to do than to place the cursor on a small dot or checkbox.

Radio Buttons

- Radio buttons are used when you want to provide the visitor, typically a limited number of options for some field, but the visitor is limited to being able to select only one of the options.

- Radio buttons are named after the type of radios that many old cars had in which selecting a station was done by depressing a selector button. Only one selector button could be pushed at a time.

- The `name` attribute serves a dual purpose for radio buttons: it links the radio buttons in a given set together, and it identifies the `value` when it is sent to the script for processing.

- The `value` attribute is crucial for a radio button, since the visitor has no way of typing a value for a radio button.

- Part of the large form example that illustrates radio buttons is shown on the next page.

```
                                  form.html — Edited
        <label for="bio">Bio:</label>
        <textarea id="bio" name="bio" rows="8" cols="50"></textarea>
</li>
<li>

        <label>Gender:</label>
        <fieldset class="radios">
        <ul>
                <li>

                        <input type="radio"    id="gender_male" name="gender" value="male" />
                        <label for="gender_male">Male</label>
                </li>
                <li>

                        <input type="radio" id="gender_female" name="gender" value="female" />
                        <label for="gender_female">Female</label>
                </li>
        </ul>
        </fieldset>
</li>


profile -->
```

CSS is used to style the radio button list elements to display inline rather than vertical. Also, the radio button labels have a right margin of 25 pixels to separate the radio button label pair.

PUBLIC PROFILE

Picture: Choose File no file selected

Maximum size of 700k. JPG, GIF, PNG.

Screen Name:

Web:

Have a homepage or a blog? Put the address here.

Bio:

Gender: ○ Male ○ Female

Radio Buttons

- If desired, the attribute `checked = "checked"` can be set to make the radio button active by default when the page is opened. You can do this to only one radio button in the set.

- If you do not set the `value` attribute, the word "on" is sent to the processing script. Its not particularly useful, since you cannot tell which button in the set was selected.

- For radio buttons only, the `name` attribute must be the same for every radio button in the set. The `id` attribute for each element on the page must be unique.

Selection Boxes

- Selection boxes are a convenient way to offer your visitors a choice from a given set of options. They are most often rendered as drop-down lists.

- You can give the user the option of choosing only one option or several options from the list.

- Selection boxes render as a box of items with a scroll bar.

- Selection boxes are made up of two HTML5 elements: select and option. You set the common name attribute in the select element and you set the value attribute in each of the option elements.

- The part of the big example markup that displays the selection box is shown on the next page.

```
</li>
<li>

        <label for="city">City:</label>
        <input type="text" id="city" name="city" class="large" />
</li>
<li

        <label for="state">State:</label>
        <select id="state" name="state">
                <option value="AL">Alabama</option>
                <option value="AK">Alaska</option>
                <option value="CA">California</option>
                <option value="FL">Florida</option>
                <option value="IN">Indiana</option>
                <option value="TX">Texas</option>
        </select>
</li>
<li>

        <label for="zip_code">ZIP Code:</label>
        <input type="text" id="zip_code" name="zip_code" class="small" />
</li>
```

Exploded view of the selection box activated by the visitor who will be selecting the Florida option. When the user lets up on the click – the form element will change to mirror their selection.

Email:

Password:

Re-enter Password:

ADDRESS

Street Address:

City:

State: Alabama

ZIP Code:

Alabama
Alaska
California
✓ Florida
Indiana
Texas

PUBLIC PROFILE

Selection Boxes

- A visitor will not be able to not make a selection in a menu unless you set the `size` attribute.

- If you have a particularly large menu of items, you may want to group the options into categories. Before the first `option` element in the first group that you want to place together enter an `optgroup` element and enclose all of the subgroup menu items inside the `optgroup` element.   Repeat this for as many subgroupings as you'd like to create.

- Browsers generally do not create true submenus, but rather group the items into a single menu with subgroups.

- The example on the next page illustrates this concept.

```
                </fieldset>
                <!-- end emails -->


<fieldset>
        <label for="referral"> Where did you find out about us?</label>
        <select id="referral" name="referral">
         <optgroup label="On-line">
           <option value="social_network">Social Network</option>
           <option value="search_engine">Search Engine</option>
         </optgroup>
         <optgroup label="Off-Line">
           <option value="postcard">Postcard</option>
           <option value="word_of_mouth">A person</option>
         </optgroup>
</fieldset>


                <fieldset>
                        <input type="submit" class="create_profile" value="Create
Account" />

                </fieldset>
```
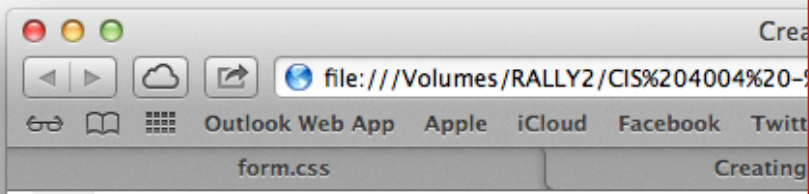
Exploded view of the subgrouping of the selection box.

Selection Boxes

- The attribute `selected="selected"` is used to specify that an option in the selection box is to be selected by default.

- If desired, the attribute `size="n"`, where `n` represents the height (in lines) of the selection box can be used to define how many lines are visible at one time by the visitor.

- If you use the `size` attribute, the selection box appears more like a list, and there is no automatically selected option (unless you use `selected`).

- If `size` is bigger than the number of options, visitors can deselect all values by clicking in the empty space.

Check Boxes

- Radio buttons can accept only one answer per set (grouping of radio buttons). Check boxes allow the user to select as many options in a set as they like.

- Similar to radio buttons, check boxes are linked together in a set or group via the value of the `name` attribute.

- The visitor can select as many checkboxes as appropriate. Each corresponding value will be sent to the script, along with the name of the checkbox set.

- As you can see in the example on the next page, the label for a checkbox comes after the input element. This means that you will often need to style the label for a checkbox separately.

File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   X

showformdata.php   email form.html   emailformdata.php   notes.html   form.

```
110    eldset>
111    d public profile -->
112
113    ART EMAILS FIELDS -->
114  □ ldset>
115    <h2 class="emails">Emails</h2>
116  □ <ul class="checkboxes">
117  □    <li>
118          <input type="checkbox" id="email_ok_msg_from_users" name="email_signup[]" value="user_emails" />
119          <label for="email_ok_msg_from_users">It is okay to email me with messages from other users.</label>
120       </li>
121  □    <li>
122          <input type="checkbox" id="email_ok_occasional_updates" name="email_signup[]" value="occasional_updates'
123          <label for="email_ok_occasional_updates">It is okay to email me with occasional promotions about our otl
124       </li>
125    </ul>
126    eldset>
127    d emails -->
128
129  □ ldset>
130    <input type="submit" class="create_profile" value="Create Account" />
131    eldset>
```

Notice that the label for the checkbox comes after the input element. Also notice that the label text does not need to match the `value` attribute. This is because the label text identifies the checkboxes to the visitor in the browser, whereas the value identifies the data to the script.

Hyper Text Markup Language file   length : 4146   lines : 135   Ln : 1   Col : 1   Sel : 0   UNIX   ANSI   INS

Text Areas

- Text areas are used when you want to provide your visitor room to write questions or comments.

- There is no `value` attribute with text areas. The value instead is the text that appears between the start and end `textarea` tags.

- Visitors can enter up to 32,700 characters in a text area (the maximum number of characters allowed). Scroll bars will appear automatically when needed, depending on the size of the text area and the number of characters the visitor has entered.

- The attributes `rows` and `cols` set the height of the `textarea` in rows and the width of the `textarea` in columns respectively. The attribute `maxlength` (new in HTML5) sets the upper limit on the number of characters that can be entered in the `textarea`.

File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   Plugins   Window   ?   X

| showformdata.php | email form.html | emailformdata.php | notes.html | form... |

```
83              <input type="text" id="screen_
84          </li>
85          <li>
86              <label for="web_site">Web:</la
87              <input type="url" id="web_site
88              <p class="instructions">Have a homepage or a blog? Put the address here.</p>
89          </li>
90          <li>
91              <label for="bio">Bio:</label>
92              <textarea id="bio" name="bio" rows="8" cols="50"></textarea>
93          </li>
94          <li>
95              <label>Gender:</label>
96              <fieldset class="radios">
97                  <ul>
98                      <li>
99                          <input type="radio"    id="gender_male" name="gender" value="male" />
100                         <label for="gender_male">Male</label>
101                     </li>
102                     <li>
103                         <input type="radio" id="gender_female" name="gender" value="female" />
104                         <label for="gender_female">Female</label>
105                     </li>
```

In this case, the height of the `textarea` will be 8 rows and the width will be 50 characters. If the user enters more text than 8 rows, vertical scrollbars will appear at that point in time.

Hyper Text Markup Language file   length : 4146   lines : 135   Ln : 87   Col : 85   Sel : 0   UNIX   ANSI   INS

K:\CIS 4004 - Fall 2012\code\Inside HTML5 - Part 4 - Forms\form.html - Notepad++

File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?

showformdata.php | email form.html | emailformdata.php | notes.html | **form.html** | form.css

```
91              <label for="bio">Bio:</label>
92              <textarea id="bio" name="bio" rows="8" cols="50" maxlength="20"></textarea>
93          </li>
94          <li>
```

Creating a Form

File Edit View History Bookmarks Window Help

file:///K:/CIS%204004%20-%20Fall%202012/code

Google

Apple Yahoo! Google Maps YouTube Wikipedia News (106) ▼ Popular ▼

Web: [   ]

Have a homepage or a blog? Put the address here.

Bio: [Four score and seven   ]

Gender: ○ Male   ○ Female

EMAILS

☐ It is okay to email me with messages from other users.

In this case the `maxlength` attribute is set to 20. As soon as the visitor has entered 20 characters, no more text can be entered.

Calendar And Time Controls

- A common issue web developers have to deal with is how to create calendar widgets that allow the users to pick a date from a calendar so they do not have to enter the date themselves.

- In the past, creation of such a widget always required JavaScript, but with HTML5 the need to use JavaScript is disappearing.

- HTML5 has several new input types that create different calendar and time controls. Currently, not all browsers support these new types. Opera has the best and most elaborate support followed by Safari and Chrome. For the examples that follow, I've used Opera 12.02 to demonstrate these new input types.

Date and Datetime

- The new HTML5 input type `date` allows the user to select a calendar date.

- In Opera, a calendar with a date picker is provided. In Safari a scrollable element is provided that defaults to the current date.

- The HTML5 input type `datetime` generates a calendar along with a time selector with the time being UTC.

- The HTML5 input type `datetime-local` is identical to `datetime` except that the time is based on your local time (whatever is considered local time by your computer).

- These are all illustrated on the next few pages.

File   Edit   Search   View   Encod

ResultSetTableModel.java   Ch

A portion of the markup for the following examples. Notice in particular the new `input` element types. Also notice that these forms are for display purposes only and are not being processed so no `method` or `action` attributes are specified.

```html
14   <div>
15       <form>
16           <fieldset>
17               <legend>Select Your Birthday</legend>
18               <ul>
19                   <li>
20                       <label for="birthday">When is your birthday?</label>
21                       <input type="date" id="birthday" name="birthday" required="required"/>
22                   </li>
23               </ul>
24           </fieldset>
25       </form>
26   </div>
27   <br />
28   <div>
29       <form>
30           <fieldset>
31               <legend>Select The Next Day and Time of Your Next Class (UTC)</legend>
32               <ul>
33                   <li>
34                       <label for="nextclass">What day and time is your next class?</label>
35                       <input type="datetime" id="nextclass" name="nextclass" required="required"/>
36                   </li>
37               </ul>
38           </fieldset>
39       </form>
40   </div>
```

Hyper Text Markup Language file   length : 1991   lines : 97   Ln : 76   Col : 5   Sel : 0   UNIX   ANSI   INS

File Edit View Bookmarks Tools Help

Local localhost/K:/

Notice the differences in the input boxes for the three different input types. Date provides only a date, while datetime provides a date and time area, but is designated UTC, and datetime-local drops the UTC designation. Next pages illustrate them in action.

Select Your Birthday

- When is your birthday? [▼]

Select The Next Day and Time of Your Next Class (UTC)

- What day and time is your next class? [▼] [:] 🔽 UTC

Select The Next Day and Time of Your Next Class (Local)

- What day and time is your next class? [▼] [:] 🔽

The screenshot shows an Opera browser window titled "Creating a Form - Text Version - Opera" displaying a form with a calendar date widget.

date

Select Your Birthday

- When is your birthday?

October ◄ ► 2012

Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

Today

Select The Next Day and Time

- What day and time is yo... UTC

Select The Next Day and Time

- What day and time is your next class?

Calendar widget is displayed allowing the user to select a date. Note scrollable year and month. Plus a "today" button to select today's date.

File   Edit   View   Bookmarks   Tools   Help

datetime-local

Local   localhost/K:/CIS%204004%20-%20Fall%202012/code/Inside

Identical in operation to `datetime` except that UTC designation is removed.

Select The Next Day and Time of Your Next Class (Local)

- What day and time is your next class? 2012-10-10 ▼ 12:07

◀	October	▶	2012

Mon	Tue	Wed	Thu	Fri	Sat	Sun
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31	1	2	3	4
5	6	7	8	9	10	11

Today

What Time Is It? (Only Local)

- What time is it now? `:`

What Month Is It?

- What month is it? ▼

Date and Datetime

- There are two new attributes, `min` and `max` that can be used to restrict the values for dates and times of the widget.

- For a date, if you wanted to make sure the user could not pick a date to far in the past, the `min` attribute would be set. Similarly, to make sure that they cannot enter a date too far in the future, the `max` attribute would be set. The values would have the format: `YYYY-MM-DD`.

- For times, similar restrictions are allowed but the format for a time is `HH:MM`.

Number Selectors

- The new `number` input type is used to allow the user to enter a number, again, without them being required to type the number.

- It accepts only numbers, otherwise, a validation error is returned.

- It allows the `min` and `max` attributes to be specified to limit the range of values that can be selected.

- It also allows for another new attribute, `step`, which allows you to specify the increment values that can be entered.

- An example is shown on the next page.

File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   Plugins   Window   ?

Ch08_LargeCo_SQL.txt | mini form 2.html | form.html | mini form 3.html

```
14  <div>
15      <form>
16          <fieldset>
17              <legend>Select A Number</legend>
18              <ul>
19                  <li>
20                      <label for="number">What is your favorite number?</label>
21                      <input type="number" id="numbergetter" name="numbergetter" required="required"/>
22                  </li>
23              </ul>
24          </fieldset>
25      </form>
26  </div>
27  <br />
28  <div>
29      <form>
30          <fieldset>
31              <legend>Select A Number Between 10 and 100 (increment by 5)</legend>
32              <ul>
33                  <li>
34                      <label for="number">What is your favorite number?</label>
35                      <input type="number" id="numberlimiter" name="numberlimiter" min="10" max="100" step="5"
36                  </li>
37              </ul>
38          </fieldset>
39      </form>
40  </div>
```

Hyper Text Markup Language file   length : 3969   lines : 189   Ln : 1   Col : 1   Sel : 0   UNIX   ANSI   INS

The up/down arrow keys allow the user to scroll through the available numbers until their desired number is selected.

Sliding Numeric Selectors

- The new `range` input type generates a slider control. It has no text area for the user to type into, and like the `number` input type, it can use the `min`, `max`, and `step` attributes.

- In the past, with HTML4 or XHTML, including such a form control would have required JavaScript to be able to generate and control the widget.

- There is limited control over how the slider looks because at the moment the controls are browser-specific. However, you can apply a height and width to the range control. If you specify a height larger than the width, the control renders vertically instead of the default horizontal rendering.

File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   Plugins   Window   ?

Ch08_LargeCo_SQL.txt | mini form 2.html | form.html | mini form 3.html

```
53              </li>
54          </ul>
55      </fieldset>
56  </form>
57  iv>
58  />
59  v>
60  <form>
61      <fieldset>
62          <legend>Select A Number (move the slider to select)</legend>
63          <ul>
64              <li>
65                  <label for="range">On a scale of 1 to 10 how to you like HTML5?</label>
66                  <input type="range" id="range1" name="range1" min="0" max="10" step="1" required="required",
67                  <output onforminput="value=range1.value"></output>
68              </li>
69          </ul>
70      </fieldset>
71  </form>
72  iv>
73  />
74  />
75  />
```
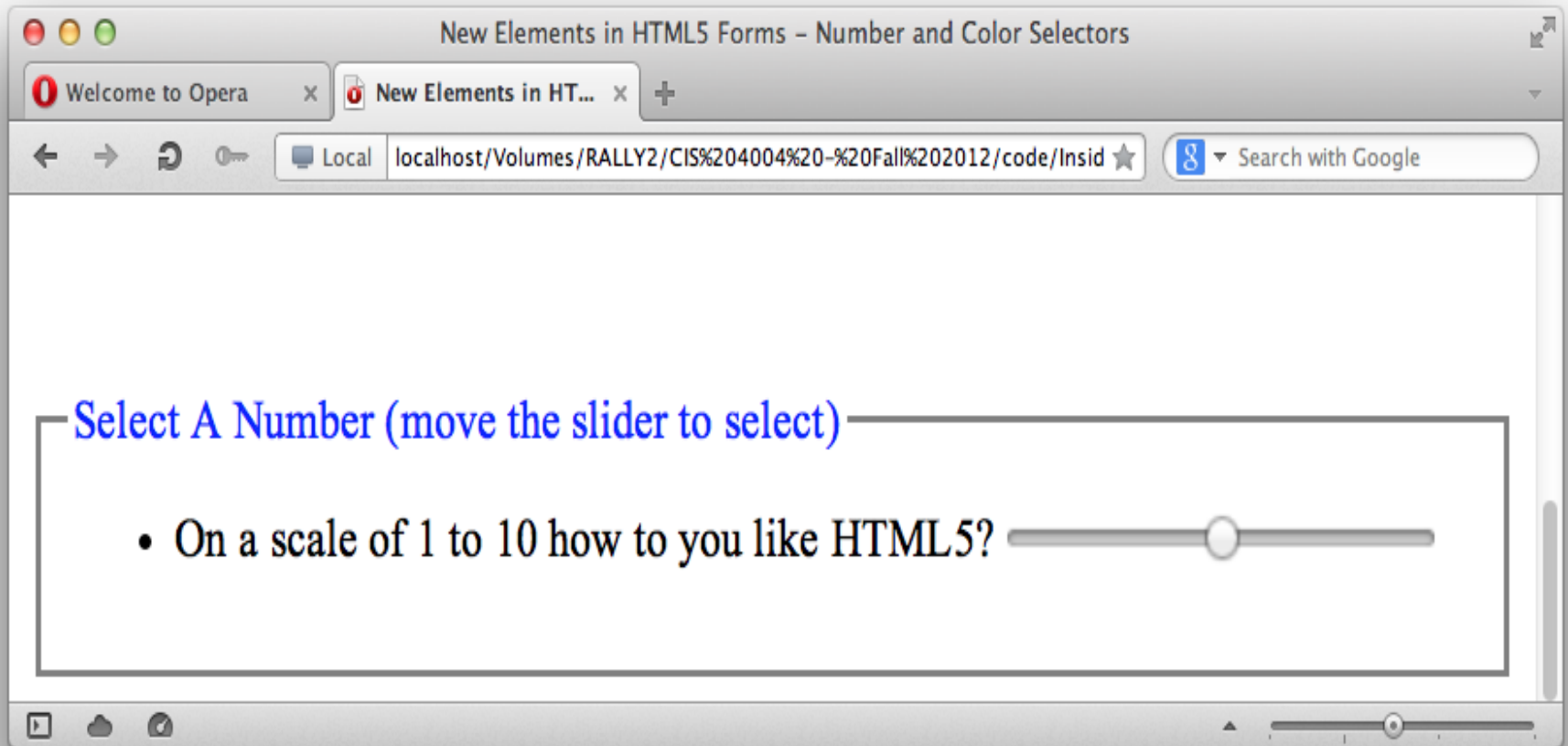
Hyper Text Markup Language file   length : 3974   lines : 189   Ln : 67   Col : 21   Sel : 0   UNIX   ANSI   INS

Color Selectors

- The new `color` input type allows the user a choice of some basic colors with the options of entering a hex value or using a color picker, similar to what is used in many software packages.

- The only desktop browser that currently supports this input type is Opera 11+ (although newer Blackberry browsers also support the color selector).

- The next couple of pages illustrate this input type.

File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?

Ch08_LargeCo_SQL.txt | mini form 2.html | form.html | mini form 3.html

```
73        />
74        />
75        />
76    v>
77    <form>
78       <fieldset>
79          <legend>Select A Color</legend>
80          <ul>
81             <li>
82                <label for="color">Pick your favorite color</label>
83                <input type="color" id="color" name="color" required="required"/>
84             </li>
85          </ul>
86       </fieldset>
87    </form>
88   iv>
89     />
90     />
91     />
92     />
93     />
94     />
95   v>
```

Hyper Text Markup Language file   length : 3974   lines : 189   Ln : 67   Col : 21   Sel : 0   UNIX   ANSI   INS
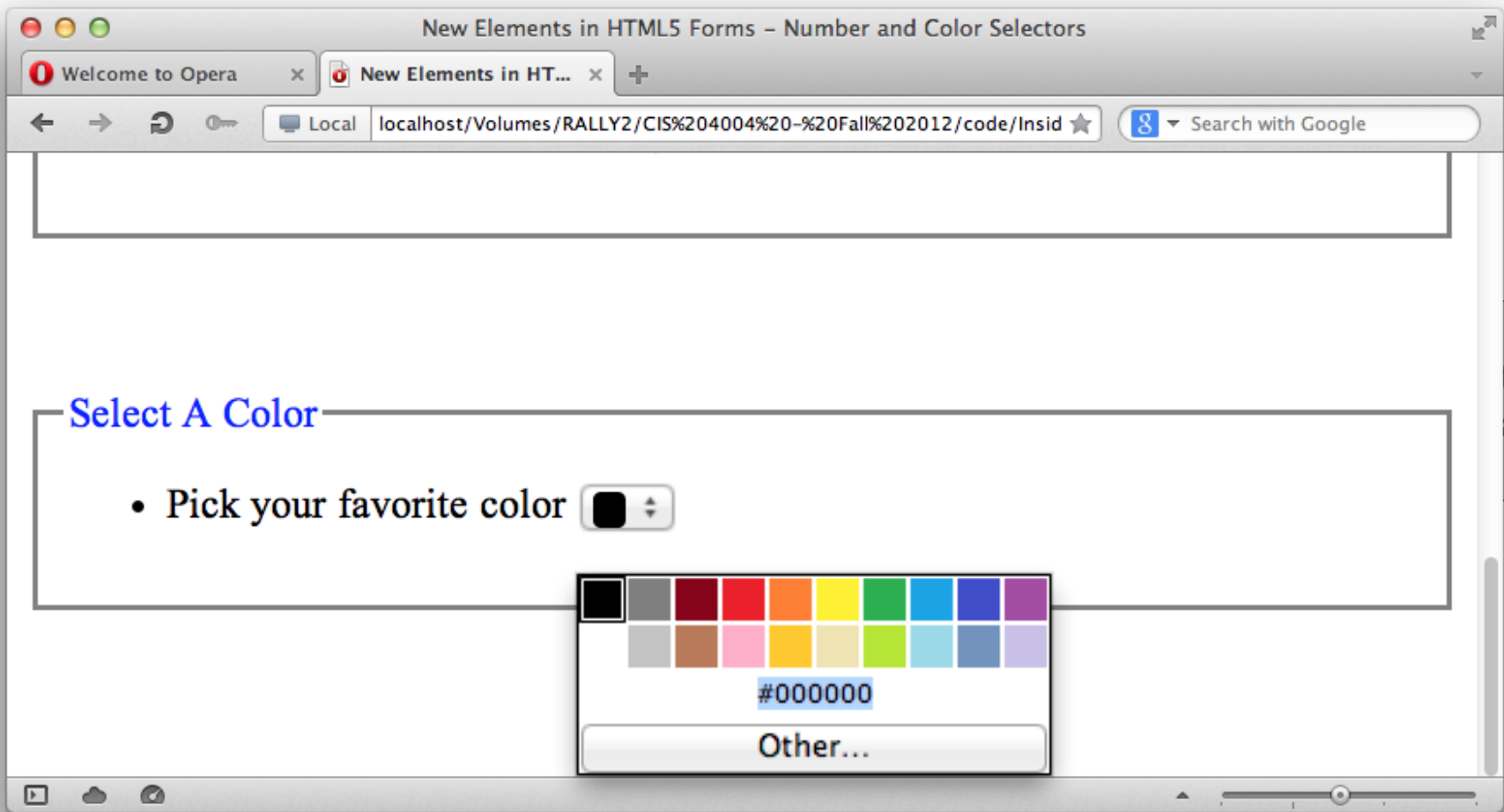
Displaying Results With The `output` Element

- The `output` element uses JavaScript to display results, usually from a calculation or from a script. It however, can also be used to add a little extra functionality to various input types like the `range` type we just saw.

- To have the value that the range slider is currently set at as the slider is being dragged you would use the output element.

- The default value is blank, but when the user moves the slider, the output value is changed and displayed to the user in real time.

- The example on the next two pages illustrates this technique. Don't worry about the JavaScript at this point, we'll see it later.

File Edit Search View Encoding Language Settings Macro Run TextFX Plugins Window ?

Ch08_LargeCo_SQL.txt | mini form 2.html | form.html | mini form 3.html

```
55              </fieldset>
56          </form>
57        iv>
58         />
59      v>
60        <form>
61          <fieldset>
62              <legend>Select A Number (move the slider to select)</legend>
63              <ul>
64                  <li>
65                      <label for="range">On a scale of 1 to 10 how to you like HTML5?</label>
66                      <input type="range" id="range1" name="range1" min="0" max="10" step="1" required="required",
67                      <output onforminput="value=range1.value"></output>
68                  </li>
69              </ul>
70          </fieldset>
71        </form>
72        iv>
73         />
74         />
75         />
76      v>
77        <form>
```

Hyper Text Markup Language file   length : 3974   lines : 189   Ln : 67   Col : 21   Sel : 0   UNIX   ANSI   INS
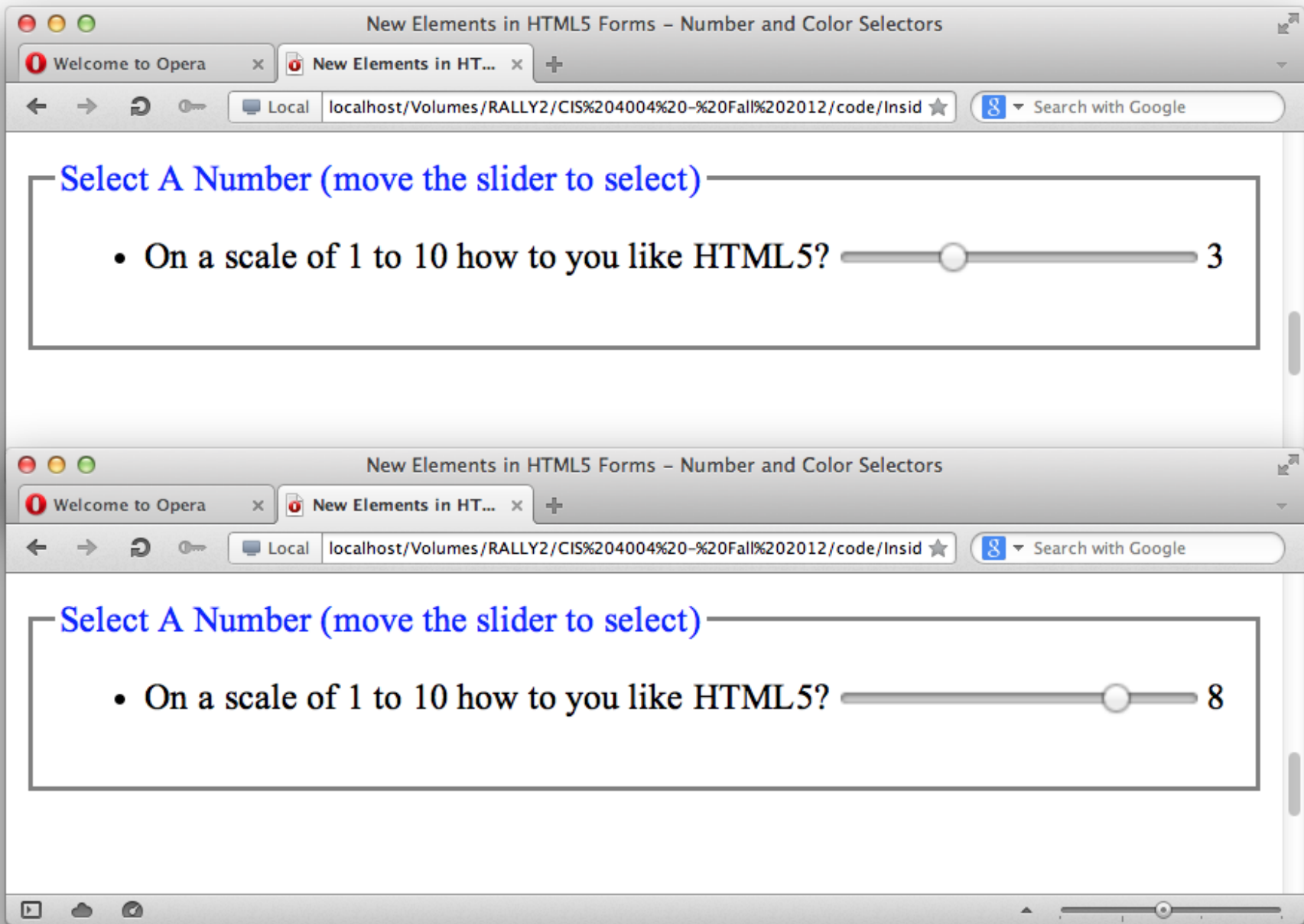
Creating An Autocomplete Feature With `list` and `datalist`

- The `datalist` element is new in HTML5. Combined with the `list` attribute, it is used to provide a predefined list of options, making the process of creating a list seem like an autocomplete form.

- User's don't necessarily have to choose from the predefined options, they can type their own answers if they wish.

- The `datalist` is similar to a `select` element. However, with a `select` element, the user cannot type their own option if they need to do so.

- The next two pages illustrate this new element.

File   Edit   Search   View   Encoding   Language   Settings   Macro   Run   TextFX   Plugins   Window   ?   X

Ch08_LargeCo_SQL.txt   mini form 2.html   form.html   mini form 3.html

```
110        <br />
111        <br />
112     <div>
113        <form>
114           <fieldset>
115              <legend>Select Your Favorite Professional Cyclist</legend>
116              <ul>
117                 <li>
118                    <label for="favcyclist">Who is your favorite cyclist?</label>
119                    <input list="cyclistlist" type="text" id="favcyclist" name="favcyclist" required="requir
120                    <datalist id="cyclistlist" width="300px">
121                       <option value="Fabian Cancellara" />
122                       <option value="Tom Boonen" />
123                       <option value="Peter Sagan" />
124                       <option value="Juan Antonio Flecha" />
125                    </datalist>
126                 </li>
127              </ul>
128           </fieldset>
129        </form>
130     </div>
131     <br/><br />
132     <br />
```

Hyper Text Markup Language file   length : 3992   lines : 189   Ln : 130   Col : 7   Sel : 0   UNIX   ANSI   INS
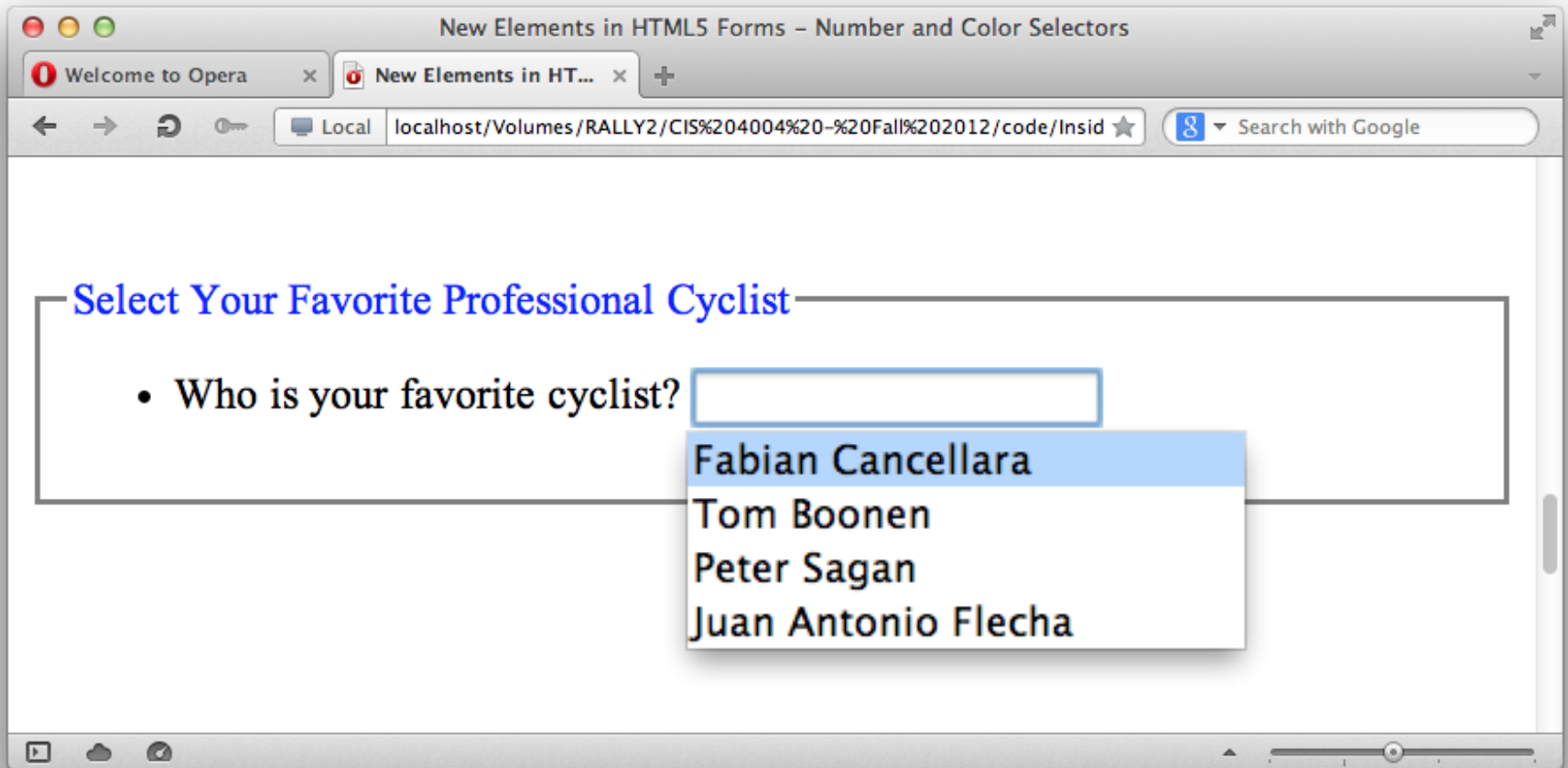
Rendering of the `datalist` element as user is seeing the list. Notice option for them to type in their own value in the text box.

This section not complete

Browser window titled "Processing Form Data" showing output from the showform.php script.

Field Name	Value(s)
first_name	Mark
last_name	Llewellyn
email	markl@cs.ucf.edu
password	passwrod
password2	password
street_address	4000 Central Florida Blvd., Bdlg 116
city	Orlando
state	FL
zip_code	32816
screen_name	mark54321
web_site	http://www.cs.ucf.edu/faculty/markl.html
bio	Happy to be here!
gender	male
user_emails	on
occasional_updates	on
picture	Desert Landscape.jpg

Output from `showform.php` script (i.e., the processed data from the form)