Fundamentals of Web Development

Third Edition by Randy Connolly and Ricardo Hoar

Chapter 5

HTML 2: Tables and Forms

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

In this chapter you will learn . . .

- What HTML tables are and how to create them
- How to use CSS to style tables
- What forms are and how they work
- What the different form controls are and how to use them
- How to improve the accessibility of your websites

Pearson

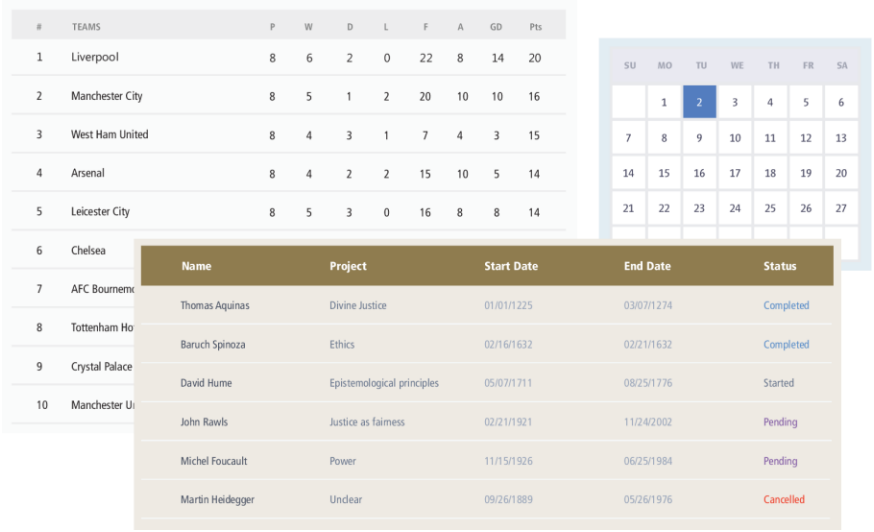
Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

HTML Tables

A **table** in HTML is created using the <table> element and can be used to represent information that exists in a two-dimensional grid.

Just like a real-world table, an HTML table can contain any type of data: not just numbers, but text, images, forms, even other tables

<table> contains any number of rows (<tr>); each row contains any number of table data cells (<td>)



#	TEAMS	P	W	D	L	F	A	GD	Pts
1	Liverpool	8	6	2	0	22	8	14	20
2	Manchester City	8	5	1	2	20	10	10	16
3	West Ham United	8	4	3	1	7	4	3	15
4	Arsenal	8	4	2	2	15	10	5	14
5	Leicester City	8	5	3	0	16	8	8	14
6	Chelsea								
7	AFC Bourneme								
8	Tottenham Ho								
9	Crystal Palace								
10	Manchester U								

SU	MO	TU	WE	TH	FR	SA
	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

Name	Project	Start Date	End Date	Status
Thomas Aquinas	Divine Justice	01/01/1225	03/07/1274	Completed
Baruch Spinoza	Ethics	02/16/1632	02/21/1632	Completed
David Hume	Epistemological principles	05/07/1711	08/25/1776	Started
John Rawls	Justice as fairness	02/21/1921	11/24/2002	Pending
Michel Foucault	Power	11/15/1926	06/25/1984	Pending
Martin Heidegger	Unclear	09/26/1889	05/26/1976	Cancelled

Pearson

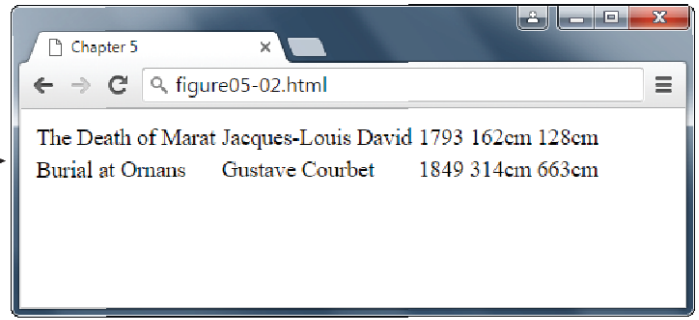
Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Basic table structure

<table>

<tr>					
	The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
<tr>	Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

```
<table>
   <tr>
      <td>The Death of Marat</td>
      <td>Jacques-Louis David</td>
      <td>1793</td>
      <td>162cm</td>
      <td>128cm</td>
   </tr>
   <tr>
      <td>Burial at Ornans</td>
      <td>Gustave Courbet</td>
      <td>1849</td>
      <td>314cm</td>
      <td>663cm</td>
   </tr>
</table>
```



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Spanning Rows and Columns

- All content must appear within the <td> or <th> container.
- Each row must have the same number of <td> or <th> containers.
- If you want a given cell to cover several columns or rows, then you can do so by using the **colspan** or **rowspan** attributes

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

FIGURE 5.4 Spanning columns

<table>

<tr>	Title <th>	Artist <th>	Year <th>	Size (width x height) <th colspan=2>	
<tr>	The Death of Marat <td>	Jacques-Louis David <td>	1793 <td>	162cm <td>	128cm <td>
<tr>	Burial at Ornans <td>	Gustave Courbet <td>	1849 <td>	314cm <td>	663cm <td>

Notice that this row now only has four cell elements.

```
<table>
   <tr>
      <th>Title</th>
      <th>Artist</th>
      <th>Year</th>
      <th colspan="2">Size (width x height)</th>
   </tr>
   <tr>
      <td>The Death of Marat</td>
      <td>Jacques-Louis David</td>
      <td>1793</td>
      <td>162cm</td>
      <td>128cm</td>
   </tr>
   ...
</table>
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Additional table elements

A title for the table is good for accessibility.

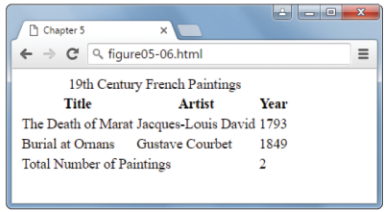
These describe our columns and can be used to aid in styling.

Table header could potentially also include other <tr> elements.

Yes, the table footer comes *before* the body.

Potentially, with styling, the browser can scroll this information while keeping the header and footer fixed in place.

```
<table>
   <caption>19th Century French Paintings</caption>
   <col class="artistName" />
   <colgroup id="paintingColumns">
      <col />
      <col />
   </colgroup>

   <thead>
      <tr>
         <th>Title</th>
         <th>Artist</th>
         <th>Year</th>
      </tr>
   </thead>

   <tfoot>
      <tr>
         <td colspan="2">Total Number of Paintings</td>
         <td>2</td>
      </tr>
   </tfoot>

   <tbody>
      <tr>
         <td>The Death of Marat</td>
         <td>Jacques-Louis David</td>
         <td>1793</td>
      </tr>
      <tr>
         <td>Burial at Ornans</td>
         <td>Gustave Courbet</td>
         <td>1849</td>
      </tr>
   </tbody>

</table>
```



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Using Tables for Layout

Prior to the broad support for CSS in browsers, HTML tables were frequently used to create page layouts. Unfortunately, this practice of using tables for layout had some problems:

1. This approach tended to increase the size of the HTML document
2. The resulting markup is not semantic. Using <table> simply to get two block elements side by side is an example of using markup simply for presentation reasons
3. Using tables for layout results in a page that is not accessible

The next chapter will examine how to use CSS for layout purposes.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Styling Tables (Borders)

Borders can be assigned to both the **<table>** and the **<td>** element. Interestingly, borders cannot be assigned to the <tr>, <thead>, <tfoot>, and <tbody> elements.

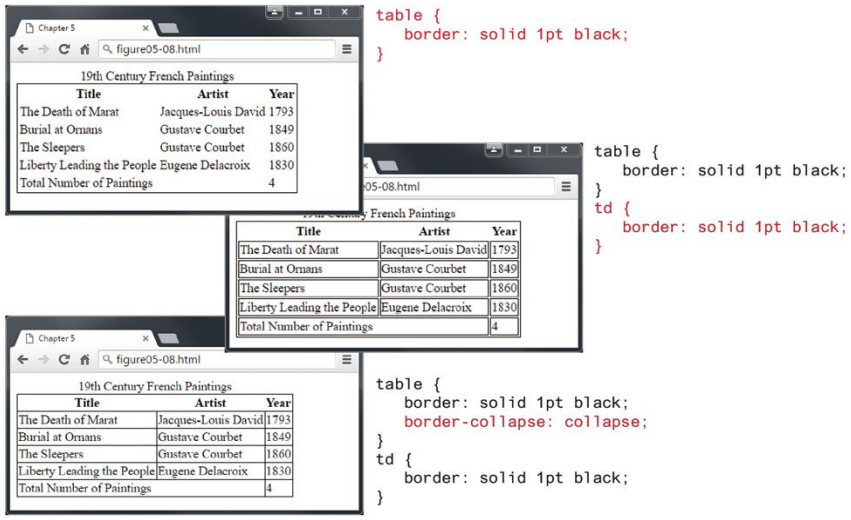
Notice as well the **border-collapse** property. This property selects the table's border model.

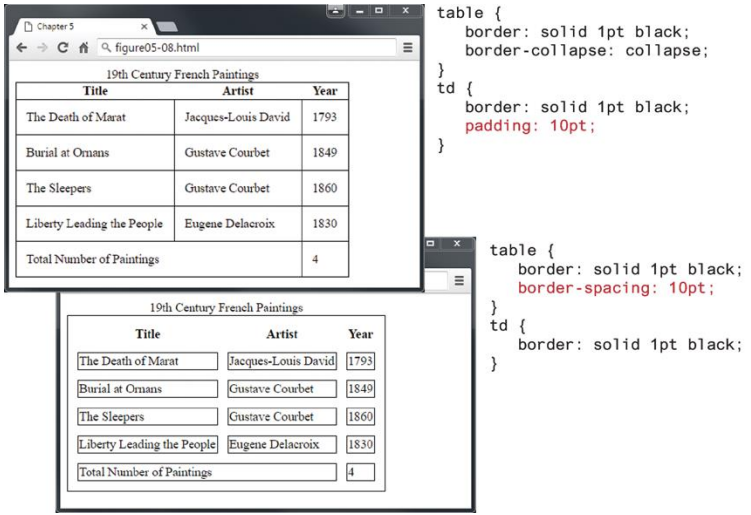
- The default is the **separated** model or value (each cell has its own unique borders)
- The **collapsed** border model makes adjacent cells share a single border.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Styling table borders



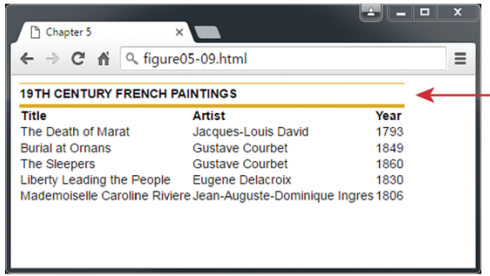


Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Boxes and Zebras

- While there is almost no end to the different ways one can style a table, there are a number of common approaches. We will look at two of them here.
- The first of these is a box format, in which we simply apply background colors and borders in various ways




```
caption {
    font-weight: bold;
    padding: 0.25em 0 0.25em 0;
    text-align: left;
    text-transform: uppercase;
    border-top: 1px solid #DCA806;
}
table {
    font-size: 0.8em;
    font-family: Arial, sans-serif;
    border-collapse: collapse;
    border-top: 4px solid #DCA806;
    border-bottom: 1px solid white;
    text-align: left;
}
```

Pearson

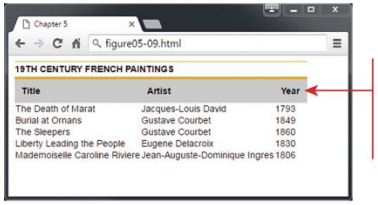
Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Boxes and Zebras (ii)

While there is almost no end to the different ways one can style a table, there are a number of common approaches. We will look at two of them here.

- The first of these is a box format, in which we simply apply background colors and borders in various ways
- See how the pseudo-element nth-child (covered in Chapter 4) can be used to alternate the format of every second row

```
caption {
   font-weight: bold;
   padding: 0.25em 0 0.25em 0;
   text-align: left;
   text-transform: uppercase;
   border-top: 1px solid #DCA806;
}
table {
   font-size: 0.8em;
   font-family: Arial, sans-serif;
   border-collapse: collapse;
   border-top: 4px solid #DCA806;
   border-bottom: 1px solid white;
   text-align: left;
}
```




```
thead tr {
   background-color: #CACACA;
}
th {
   padding: 0.75em;
}
```

```
tbody tr {
   background-color: #F1F1F1;
   border-bottom: 1px solid white;
   color: #6E6E6E;
}
tbody td {
   padding: 0.75em;
}
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Boxes and Zebras (iii)

We can then

- add special styling to the :hover pseudo-class of the <tr> element to highlight a row when the mouse cursor hovers over
- use the pseudo-element nth-child (covered in Chapter 4) to alternate the format of every second row.

```
tbody tr:hover {
   background-color: #9e9e9e;
   color: black;
}
```

```
tbody tr:nth-child(even) {
   background-color: lightgray;
}
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Introducing Forms

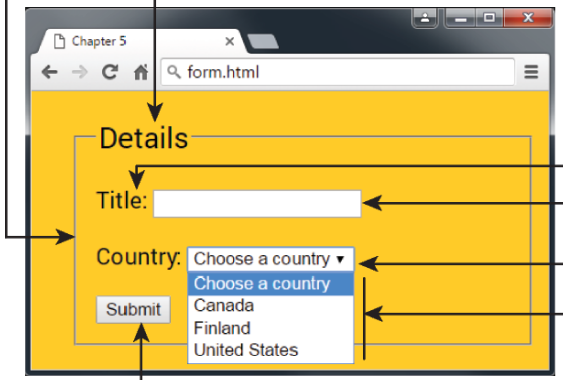
Forms provide the user with an alternative way to interact with a web server.

- Up to now, clicking hyperlinks was the only mechanism available to the user
- Using a form, the user can enter text, choose items from lists, and click buttons
- Typically, programs running on the server will take the input from HTML forms and do something with it,
  - such as save it in a database,
  - interact with an external web service, or
  - customize subsequent HTML based on that input.
- HTML5 has added a number of new controls and more customization options

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Form Structure

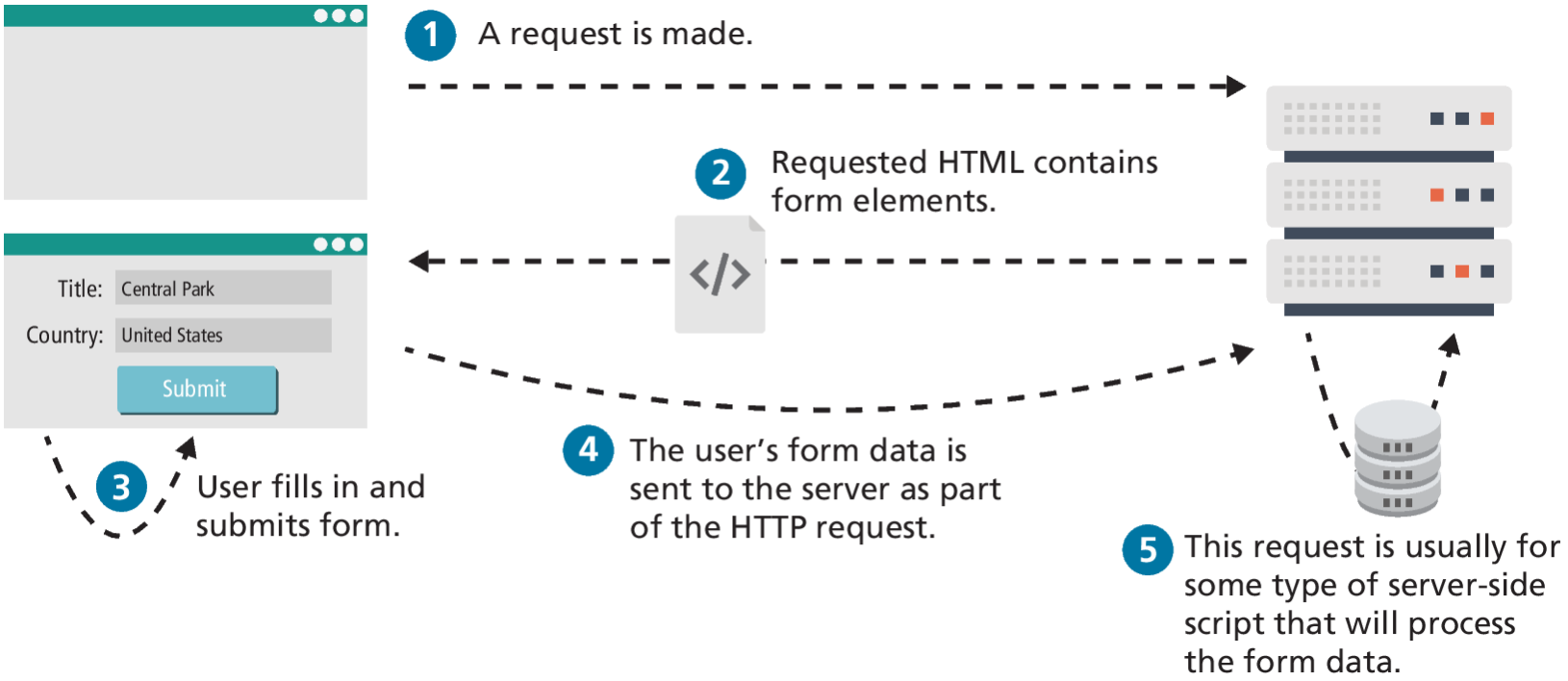



```
<form method="post" action="process.php">
  <fieldset>
    <legend>Details</legend>
    <p>
      <label>Title: </label>
      <input type="text" name="title" />
    </p>
    <p>
      <label>Country: </label>
      <select name="where">
        <option>Choose a country</option>
        <option>Canada</option>
        <option>Finland</option>
        <option>United States</option>
      </select>
    </p>
    <input type="submit" />
  </fieldset>
</form>
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

How Forms Work



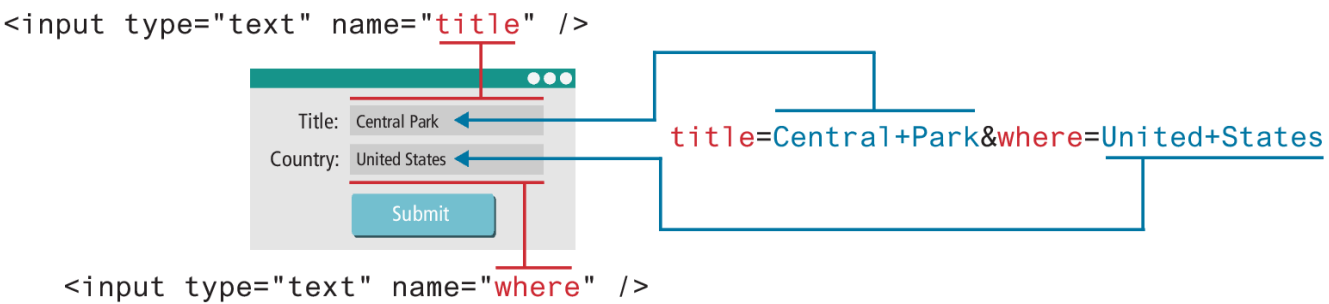
Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Query Strings

The browser “sends” the data to the server via an HTTP request using a query string.

A **query string** is a series of **name=value** pairs separated by ampersands (the & character). Special symbols must be **URL encoded**



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

The <form> Element

Two important attributes that are essential features of any <form> are the **action** and the **method** attributes.

- The **action** attribute specifies the URL of the server-side resource that will process the form data.
- The **method** attribute specifies how the query string data will be transmitted from the browser to the server. There are two possibilities:
  - GET and
  - POST.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

GET

GET

- Data can be clearly seen in the address bar. This may be an advantage during development but a disadvantage in production.
- Data remains in browser history and cache. Again this may be beneficial to some users, but it is a security risk on public computers.
- Data can be bookmarked (also an advantage and a disadvantage).
- There is a limit on the number of characters in the returned form data.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

POST

POST

- Data can contain binary data.
- Data is hidden from user.
- Submitted data is not stored in cache, history, or bookmarks.

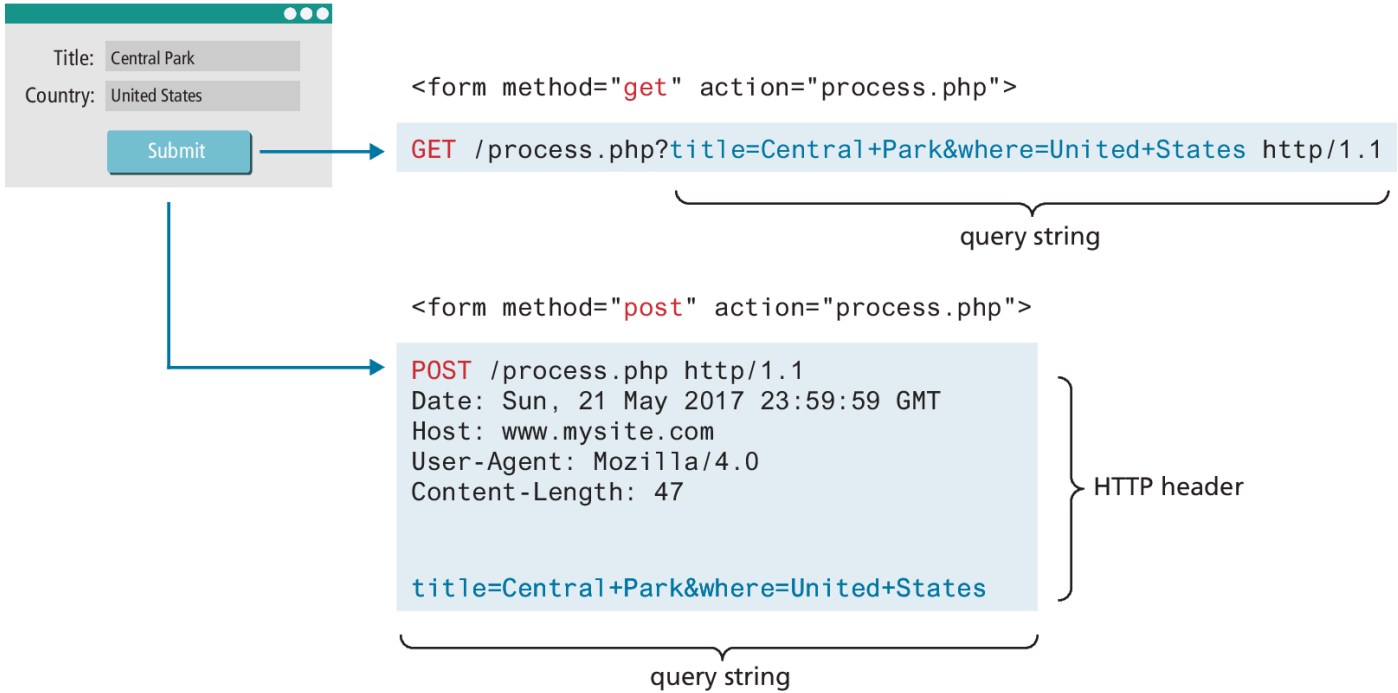
NOTE

It should be noted that while the POST method “hides” form data, any user could easily inspect the HTTP header. As a result, the POST method is NOT sufficient from a security standpoint.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

GET vs POST (example)



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Form Control Elements

<button> Defines a clickable button.

<datalist> An HTML5 element that defines lists of pre-defined values to use with input fields.

<fieldset> Groups related elements in a form together.

<form> Defines the form container.

<input> Defines an input field. HTML5 defines over 20 different types of input.

<label> Defines a label for a form input element.

<legend> Defines the label for a fieldset group.

<optgroup> Defines a group of related options in a multi-item list.

<option> Defines an option in a multi-item list.

<output> Defines the result of a calculation.

<select> Defines a multi-item list.

<textarea> Defines a multiline text entry box.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Text Input Controls

Most forms need to gather text information from the user. Whether it is a search box or a login form or a user registration form, some type of text input is usually necessary.

```
<input type="text" ... />
```

Text:

```
<textarea>
  enter some text
</textarea>
```

```
<textarea placeholder="enter some text">
</textarea>
```

TextArea: enter some text

TextArea: Enter some text

```
<input type="password" ... />
```

Password:

Password: ••••

```
<input type="search" placeholder="enter search text" ... />
```

Search: enter search text

Search: HTML

```
<input type="email" ... />
```

Email: fdsdfs

Please enter a valid email address

In Opera

Email: sdasdas

Please enter an email address.

In Chrome

```
<input type="url" ... />
```

url: sdsdfdf

Please enter a URL.

```
<input type="tel" ... />
```

Tel:

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Choice Controls

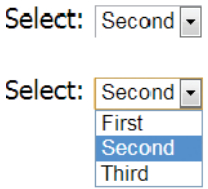
Forms often need the user to select an option from a group of choices. HTML provides several ways to do this.

- Select Lists
- Radio Buttons
- Checkboxes

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Select Lists

- The **<select>** element is used to create a multiline box for selecting one or more items. The options (defined using the **<option>** element) can be hidden in a dropdown list or multiple rows of the list can be visible.
- The selected attribute in the **<option>** makes it a default value.
- The value attribute is optional; if it is not specified, then the text within the container is sent instead



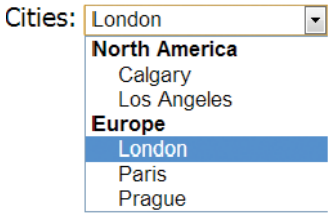

```
<select name="choices">
  <option>First</option>
  <option selected>Second</option>
  <option>Third</option>
</select>
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Select Lists (ii)

- Option items can be grouped together via the **<optgroup>** element.




```
<select ... >
 <optgroup label="North America">
   <option>Calgary</option>
   <option>Los Angeles</option>
 </optgroup>
 <optgroup label="Europe">
   <option>London</option>
   <option>Paris</option>
   <option>Prague</option>
 </optgroup>
</select>
```

Pearson

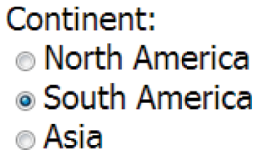
Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Radio Buttons

Radio buttons are useful when you want the user to select a single item from a small list of choices and you want all the choices to be visible.

radio buttons are added via the **<input type="radio">** element

The checked attribute is used to indicate the default choice, while the value attribute works in the same manner as with the **<option>** element.




```
<input type="radio" name="where" value="1">North America<br/>
<input type="radio" name="where" value="2" checked>South America<br/>
<input type="radio" name="where" value="3">Asia
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Checkboxes

A **checkbox** is used for obtaining a yes/no or on/off response from the user.

Checkboxes are added via the **<input type="checkbox">**

The **checked** attribute can be used to set the default value of a checkbox.

Each checked checkbox will have its value sent to the server.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Button Controls

<input type="submit"> Creates a button that submits the form data to the server.

<input type="reset"> Creates a button that clears any of the user's already entered form data.

<input type="button"> Creates a custom button. This button may require JavaScript for it to actually perform any action.

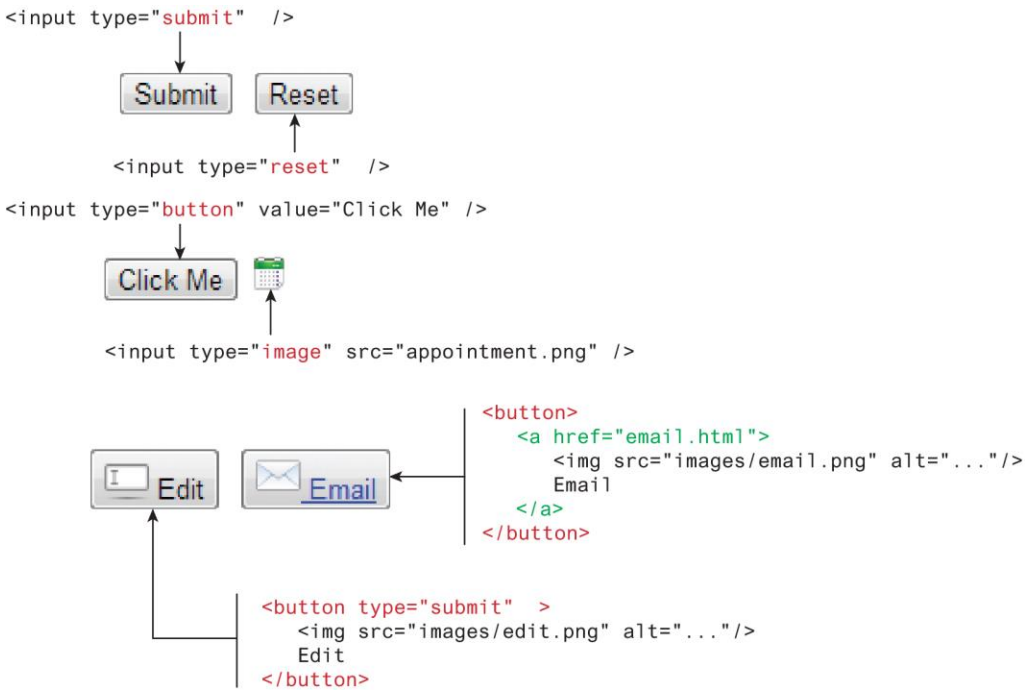
<input type="image"> Creates a custom submit button that uses an image for its display.

<button> Creates a custom button.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Example button elements



Pearson

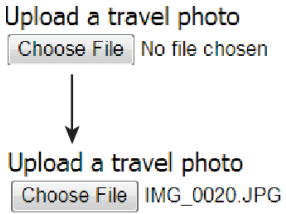
Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Specialized Controls

<input type="hidden"> element will be covered in more detail in Chapter 15 on State Management

<input type="file"> element, which is used to upload a file from the client to the server

Notice that the **<form>** element must use the **post** method and must include the **enctype="multipart/form-data"** attribute as well.




```
<form method="post" enctype="multipart/form-data" ... >
   ...
   <label>Upload a travel photo</label>
   <input type="file" name="photo" />
   ...
</form>
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Number and Range

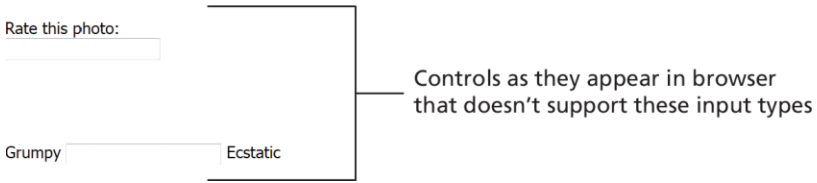
- HTML5 introduced the number and range controls provide a way to input numeric values that eliminates the need for client-side numeric validation (for security reasons you would still check the numbers for validity on the server)

Rate this photo:
2

```
<label>Rate this photo: <br/>
<input type="number" min="1" max="5" name="rate" />
```

Grumpy Ecstatic

```
Grumpy
<input type="range" min="0" max="10" step="1" name="happiness" />
Ecstatic
```



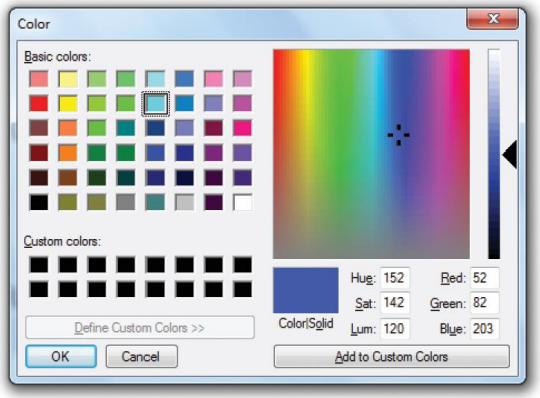
Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Color

When it is necessary, the HTML5 color control provides a convenient interface for the user

Background Color:




```
<label>Background Color: <br/>
<input type="color" name="back" />
```

Background Color:

Control as it appears in browser that doesn't support this input type

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Date and Time Controls

date Creates a general date input control. The format for the date is “yyyy-mm-dd.”

time Creates a time input control. The format for the time is “HH:MM:SS,” for hours:minutes:seconds.

datetime Creates a control in which the user can enter a date and time.

datetime-local Creates a control in which the user can enter a date and time without specifying a time zone.

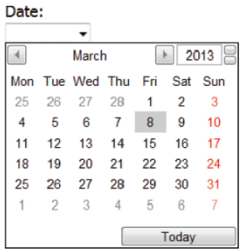
month Creates a control in which the user can enter a month in a year. The format is “yyyy-mm.”

week Creates a control in which the user can specify a week in a year. The format is “yyyy-W##.”

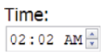
Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

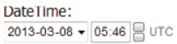
Date and time controls Figure



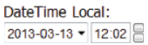

```
<label>Date: <br/>
<input type="date" ... />
```



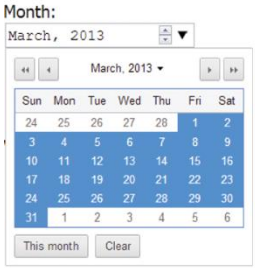

```
<input type="time" ... />
```



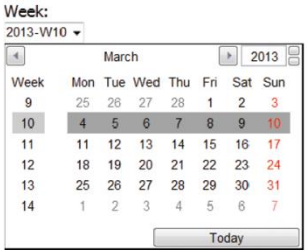

```
<input type="datetime" ... />
```




```
<input type="datetime-local" ... />
```




```
<input type="month" ... />
```




```
<input type="week" ... />
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Table and Form Accessibility

The W3C created the **Web Accessibility Initiative (WAI)** in 1997 to improve the accessibility of websites. Perhaps the most important guidelines in that document are:

- *Provide text alternatives for any nontext content so that it can be changed into other forms people need, such as large print, braille, speech, symbols, or simpler language*.
- *Create content that can be presented in different ways (for example, simpler layout) without losing information or structure*.
- *Make all functionality available from a keyboard*.
- *Provide ways to help users navigate, find content, and determine where they are*.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Accessible Tables

- Describe the table's content using the <caption> element
- Connect the cells with a textual description in the header using use the **scope** attribute.

```
<table>
  <caption>Famous Paintings</caption>
  <tr>
    <th scope="col">Title</th>
    <th scope="col">Artist</th>
    <th scope="col">Year</th>
    <th scope="col">Width</th>
    <th scope="col">Height</th>
  </tr>
  <tr>
    <td>The Death of Marat</td>
    <td>Jacques-Louis David</td>
    <td>1793</td>
    <td>162cm</td>
    <td>128cm</td>
  </tr>
…
```

LISTING 5.1 (edited) Connecting cells with headers

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Accessible Forms

We already made use of the <fieldset>, <legend>, and <label> elements.

Their main purpose is to logically group related form input elements together with the <legend> providing a type of caption for those elements.

Each <label> element should be associated with a single input element.

You can make this association explicit by using the for attribute so means that if the user clicks on or taps the <label> text the control will receive the focus

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Associating labels and input elements

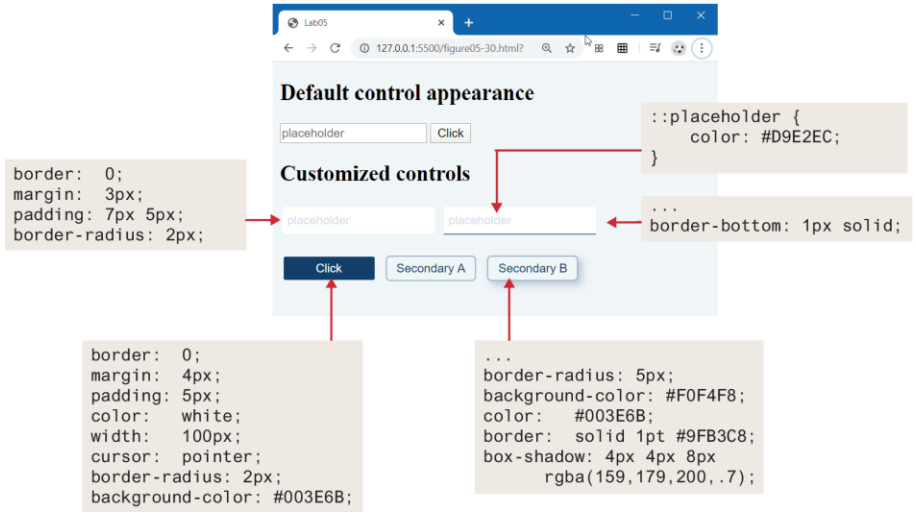
```
<label for="f-title">Title: </label>

<input type="text" name="title" id="f-title"/>


<label for="f-country">Country: </label>

<select name="where" id="f-country">
    <option>Choose a country</option>
    <option>Canada</option>
    <option>Finland</option>
    <option>United States</option>
</select>
```

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Styling Form Elements

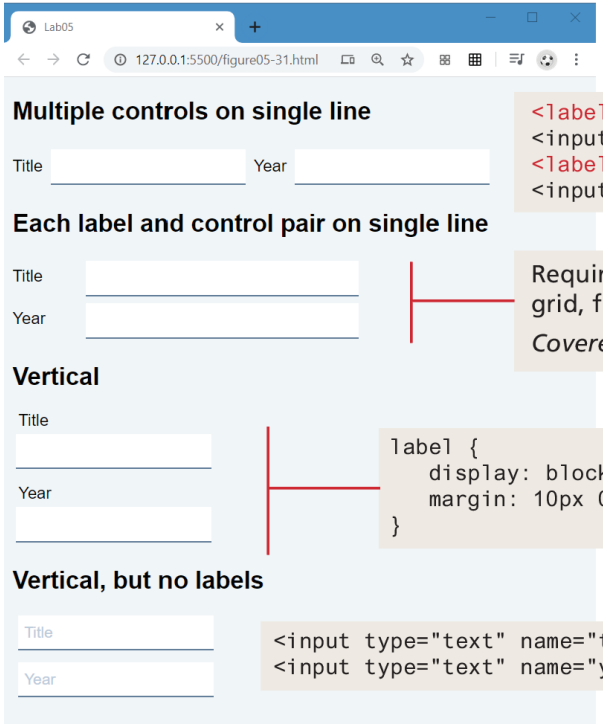
Let's begin with the common text and button controls. A common styling change is to eliminate the borders and add in rounded corners and padding.



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Working with labels




```
<label for="title">Title</label>
<input type="text" name="title" id="title"/>
<label for="year">Year</label>
<input type="text" name="year" id="year" />
```

Requires CSS layout techniques, such as grid, flex, floats, or positioning.

Covered in Chapter 7.

```
label {
   display: block;
   margin: 10px 0 0 5px;
}
```

```
<input type="text" name="title" placeholder="Title" />
<input type="text" name="year" placeholder="Year" />
```

Pearson

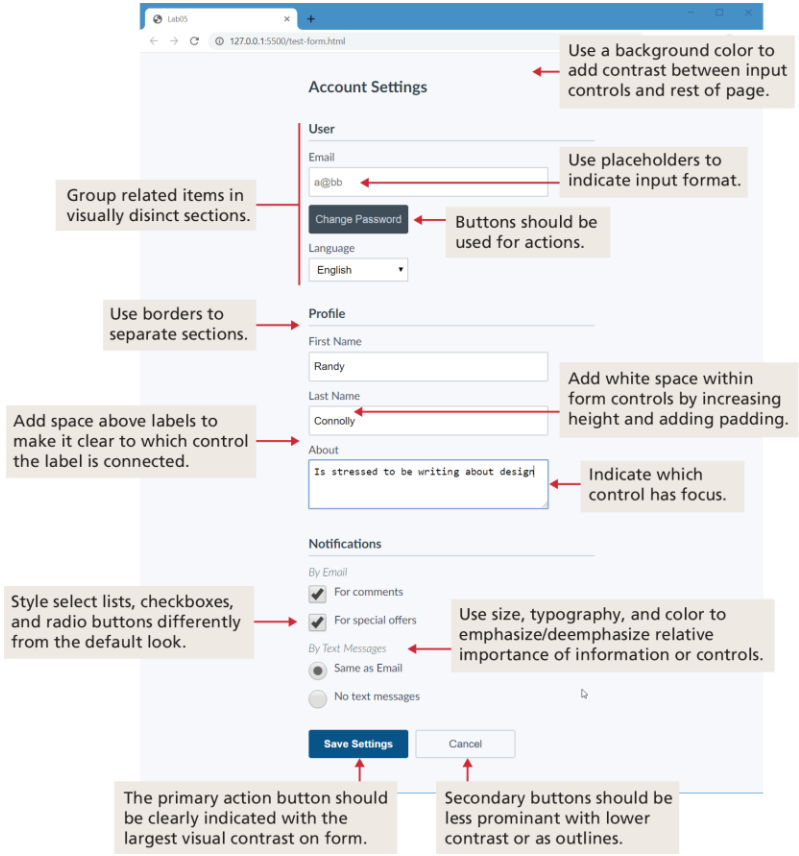
Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Form Design

A well-designed form communicates to a user that the site values their time and data.

Perhaps the first and most important rule is to style your form elements so they look different from the default settings.

Figure 5.33 describes and illustrates a small set of straightforward additional precepts



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Validating User Input

- User input must never be trusted.
- It could be missing.
- It might be in the wrong format.
- It might even contain JavaScript or SQL as a means to causing some type of havoc.
- Thus, almost always user input must be tested for validity.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Types of Input Validation

- **Required information.** Some data fields just cannot be left empty.
- **Correct data type.** Some fields such as numbers or dates, must follow the rules for its data type in order to be considered valid.
- **Correct format.** Some information, such as postal codes, credit card numbers, and social security numbers have to follow certain pattern rules.
- **Comparison.** Some user-entered fields are considered correct or not in relation to an already inputted value (password confirm)
- **Range check.**
- **Custom.**

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

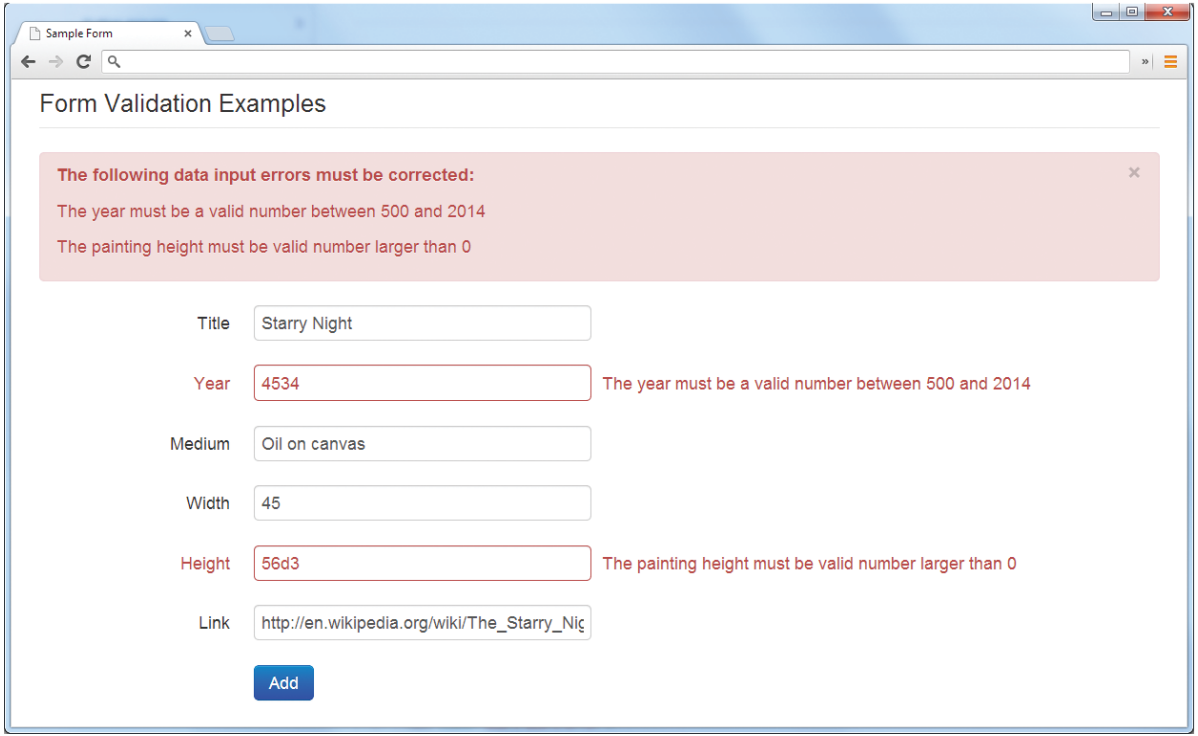
Notifying the User

- **What is the problem?** Users do not want to read lengthy messages to determine what needs to be changed.
- **Where is the problem?** Some type of error indication should be located near the field that generated the problem.
- **If appropriate, how do I fix it?** For instance, don't just tell the user that a date is in the wrong format; tell him or her what format you are expecting

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Notifying the User (Figures)



Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

How to Reduce Validation Errors

Provide textual hints to the user on the form itself (last slide)

Using tool tips or pop-overs to display context-sensitive help about the expected input

Another technique for helping the user understand the correct format for an input field is to provide a JavaScript-based mask

Providing sensible default values for text fields can reduce validation errors

Finally, many user input errors can be eliminated by choosing a better data entry type than the standard <input type="text">.

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Where to Perform Validation

Validation can be performed at three different levels.

- With HTML5, the browser can perform **basic validation**.
- **JavaScript validation** dramatically improves the user experience of data-entry forms, and is an essential feature of any real-world web site that uses forms. Unfortunately, JavaScript validation cannot be relied on.
- **server-side validation** is arguably the most important since it is the only validation that is guaranteed to run.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Key Terms

- accessibility
- branch
- CAPTCHA
- checkbox
- forking
- form
- GET
- Git
- GitHub
- input validation
- local repository
- POST
- query string
- radio buttons
- remote repository
- spam bots
- table
- URL encoded
- version control
- Web Accessibility Initiative (WAI)

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved

Copyright

This work is protected by United States copyright laws and is provided solely for the use of instructors in teaching their courses and assessing student learning. Dissemination or sale of any part of this work (including on the World Wide Web) will destroy the integrity of the work and is not permitted. The work and materials from it should never be made available to students except by instructors using the accompanying text in their classes. All recipients of this work are expected to abide by these restrictions and to honor the intended pedagogical purposes and the needs of other instructors who rely on these materials.

Pearson

Copyright © 2021, 2018, 2015 Pearson Education, Inc. All Rights Reserved