

Module II

HTML Tables and Forms & Advanced CSS

2.1 Introduction to 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.
- A table is a matrix of cells. The cells in the top row often contain column labels, those in the leftmost column often contain row labels, and most of the rest of the cells contain the data of the table.
- The content of a cell can be almost any document element, including text, a heading, a horizontal rule, an image, and a nested table.

Basic Table Tags

A table is specified as the content of the block tag `<table>`.

There are two kinds of lines in tables:

- the line around the whole table is called the border;
 - the lines that separate the cells from each other are called rules.
-
- A table that does not include the border attribute will be a matrix of cells with neither a border nor rules.
 - The browser has default widths for table borders and rules, which are used if the border attribute is assigned the value "border." Otherwise, a number can be given as border's value, which specifies the border width in pixels. For example, border = "3" specifies a border 3 pixels wide.
 - A border value of "0" specifies no border and no rules. The rules are set at 1 pixel when any nonzero border value is specified. The border attribute is the most common attribute for the `<table>` tag.
 - In most cases, a displayed table is preceded by a title, given as the content of a `<caption>` tag, which can immediately follow the opening `<table>` tag.
 - The cells of a table are specified one row at a time. Each row of a table is specified with a row tag, `<tr>`.
 - Within each row, the row label is specified by the table heading tag, `<th>`. Each data cell of a row is specified with a table data tag, `<td>`

The first row of a table usually has the table's column labels. For example, if a table has three data columns and their column labels are, Apple, Orange, and Screwdriver respectively, the first row can be specified by the following:

```
<tr>
<th> Apple </th>
<th> Orange </th>
<th> Screwdriver </th>
</tr>
```

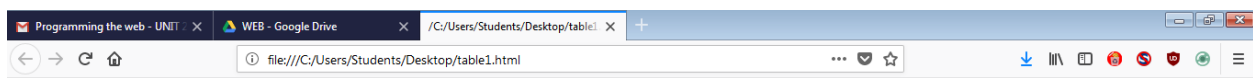
Each data row of a table is specified with a heading tag and one data tag for each data column. For example, the first data row for our work-in-progress table might be as follows:

```
<tr>
<th> Breakfast </th>
<td> 0 </td>
<td> 1 </td>
<td> 0 </td>
</tr>
```

The following document describes the whole table:

Eg:

```
<html>
<body>
<br />
<table border="1">
<caption> Diet </caption>
<tr>
    <th>Breakfast</th>
    <th>Lunch</th>
    <th>Dinner</th>
</tr>
<tr>
    <td>Apple</td>
    <td>Rice</td>
    <td>Cucumber</td>
</tr>
<tr>
    <td>Watermelon</td>
    <td>Rice</td>
    <td>Papaya</td>
</tr>
</table>
</body>
</html>
```

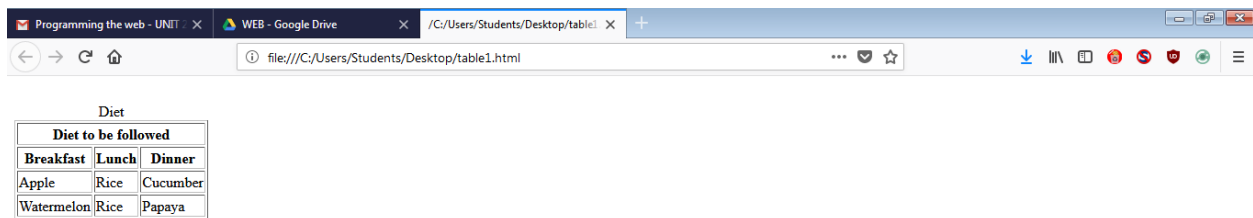


Breakfast	Lunch	Dinner
Apple	Rice	Cucumber
Watermelon	Rice	Papaya

The rowspan and colspan Attributes

In many cases, tables have multiple levels of row or column labels in which one label covers two or more secondary labels. For example, consider the display of a partial table shown in Figure below. In this table, the upper-level label ‘Diet to be followed’ spans the three lower-level label cells. Multiple-level labels can be specified with the rowspan and colspan attributes.

```
<tr>
<th colspan="3">Diet to be followed</th>
</tr>
```



Diet to be followed		
Breakfast	Lunch	Dinner
Apple	Rice	Cucumber
Watermelon	Rice	Papaya

The colspan attribute specification in a table header or table data tag tells the browser to make the cell as wide as the specified number of rows.

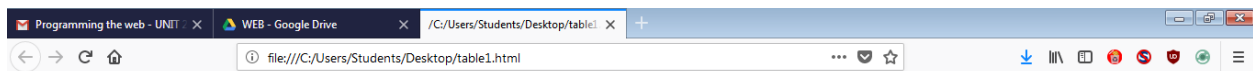
Eg:

```
<html>
<body>
  <table border="1">
    <caption> Diet </caption>
    <tr>
      <th rowspan="2"></th>
      <th colspan="3">Fruit Juice Drinks</th>
    </tr>
    <tr>
      <th>Apple</th>
      <th>Orange</th>
      <th>Strawberry</th>
    </tr>
    <tr>
      <th>Breakfast</th>
      <td>0</td>
      <td>1</td>
      <td>0</td>
    </tr>
    <tr>
      <th> Lunch</th>
      <td>1</td>
      <td>1</td>
```

```

        <td>1</td>
    </tr>
    <tr>
        <th>Dinner</th>
        <td>0</td>
        <td>1</td>
        <td>0</td>
    </tr>
</table>
</body>
</html>

```



Diet			
	Fruit Juice Drinks		
	Apple	Orange	Strawberry
Breakfast	0	1	0
Lunch	1	1	1
Dinner	0	1	0

Additional Table Elements

- The <thead>, <tfoot>, and <tbody> elements can also be used in table.
- The headings of the table are put in <thead> element, the content of rows in <tbody> element and if any summaries in <tfoot> element.
- These elements divide the table into different section. CSS can be applied on these sections separately.
- The <col> and <colgroup> elements are also mainly used to aid in the styling of the table. Number of columns can be grouped and similar style is applied to the whole group. The possible properties that can be set are borders, backgrounds, width, and visibility.
- HTML tables were frequently used to create page layouts, which divide the window into many frames. Images, link tag <a> and text can be put in different cells of the table. Some of the problems occurred due to this approach are –
 - Tend to dramatically increase the size of the HTML document
 - Large number of extra tags are required for <table> elements
 - These files take longer to download and are difficult to maintain because of the extra markup.
 - It is not semantic, as tables are meant to indicate tabular data

```

<table>
<tr>
    <td>
        
    </td>
    <td>
        <h2>Castle</h2>
        <p>Lewes, UK</p>
        <p>Photo by: Michele Brooks</p>
    </td>
</tr>
</table>

```

<p>Built in 1069, the castle has a tremendous view of the town of Lewes and the surrounding countryside.

</p>

<h3>Other Images by Michele Brooks</h3>

<table>

<tr>

<td></td>

<td></td>

</tr>

<tr>

<td></td>

<td></td>

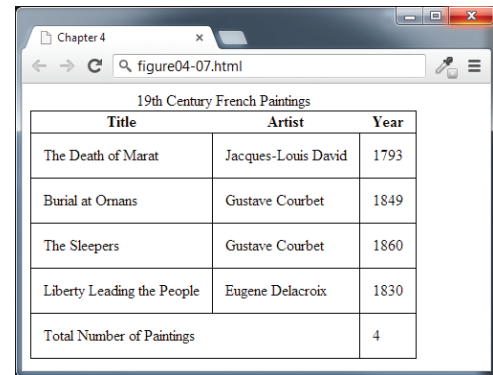
</tr>

</table>

</td>

</tr>

</table>



Title	Artist	Year
The Death of Marat	Jacques-Louis David	1793
Burial at Ornans	Gustave Courbet	1849
The Sleepers	Gustave Courbet	1860
Liberty Leading the People	Eugene Delacroix	1830
Total Number of Paintings		4

2.2 Styling Tables

All the CSS properties can be applied on tables, the other styling properties for only table are-

- Table Borders
- Boxes and Zebras

Table Borders

Table borders can be assigned to both the <table> and the <td> element (or <th>). Borders cannot be assigned to the <tr>, <thead>, <tfoot>, and <tbody> elements.

The border-collapse property selects the table's border model. By default, each cell has its own unique borders. The space between the adjacent borders can be changed by using the border-spacing property.

Boxes and Zebras

By using the CSS style, it is possible to change the background colors and borders, change the appearance of a row when mouse moves over it and also change the format of nth child




Title	Artist	Year
The Death of Marat	Jacques-Louis David	1793
Burial at Ornans	Gustave Courbet	1849
The Sleepers	Gustave Courbet	1860
Liberty Leading the People	Eugene Delacroix	1830
Mademoiselle Caroline Riviere	Jean-Auguste-Dominique Ingres	1806

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

19TH CENTURY FRENCH PAINTINGS		
Title	Artist	Year
The Death of Marat	Jacques-Louis David	1793
Burial at Ornans	Gustave Courbet	1849
The Sleepers	Gustave Courbet	1860
Liberty Leading the People	Eugene Delacroix	1830
Mademoiselle Caroline Riviere	Jean-Auguste-Dominique Ingres	1806

```
tbody tr:nth-child(odd) {
background-color: white;
}
```

2.3 Introducing Forms

- **Forms** provide the user with an alternative way to interact with a web server. Another way is by using hyperlinks.
- Using a form, the user can enter text, choose items from lists, and click buttons. Programs running on the server will take the input from HTML forms and processes it, or save it.
- A form is defined by a `<form>` element, which is a container for other elements that represent the various input elements within the form as well as plain text and almost any other HTML element.

How Forms Work

While forms are constructed with HTML elements, a form also requires some type of server-side resource that processes the user's form input.

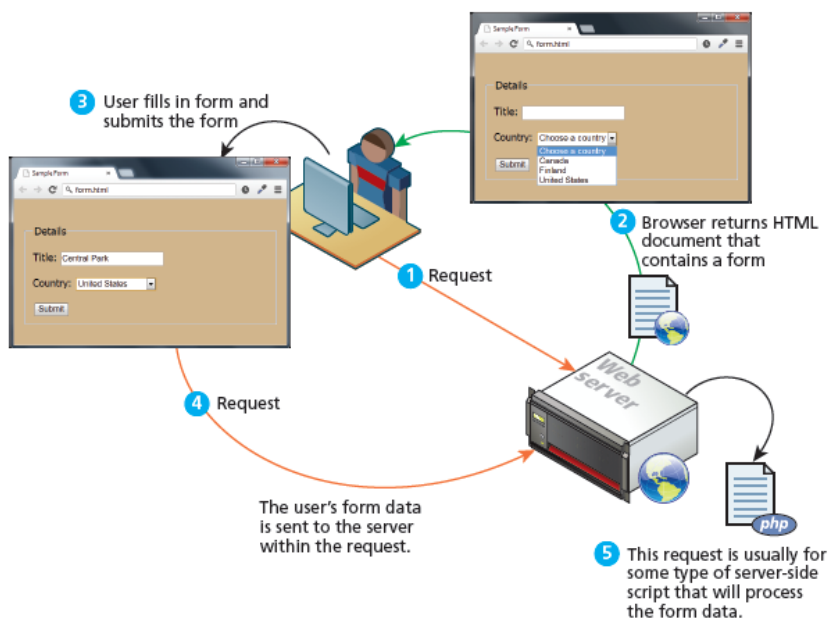
Sending of data to the server (query string):

- The browser packages the user's data input into something called a query string.
- A **query string** is a series of name=value pairs separated by ampersands (the **&** character). The names in the query string are defined in the HTML form; each form element contains a name attribute, which is used to define the name for the form data in the query string.
- The values in the query string are the data entered by the user.
- Query strings have certain rules defined by the HTTP protocol.
- Certain characters such as spaces, punctuation symbols, and foreign characters cannot be part of a query string.

```
<input type="text" name="title" />
```

```
title=Central+Park&where=United+States
```

```
<select name="where">
```

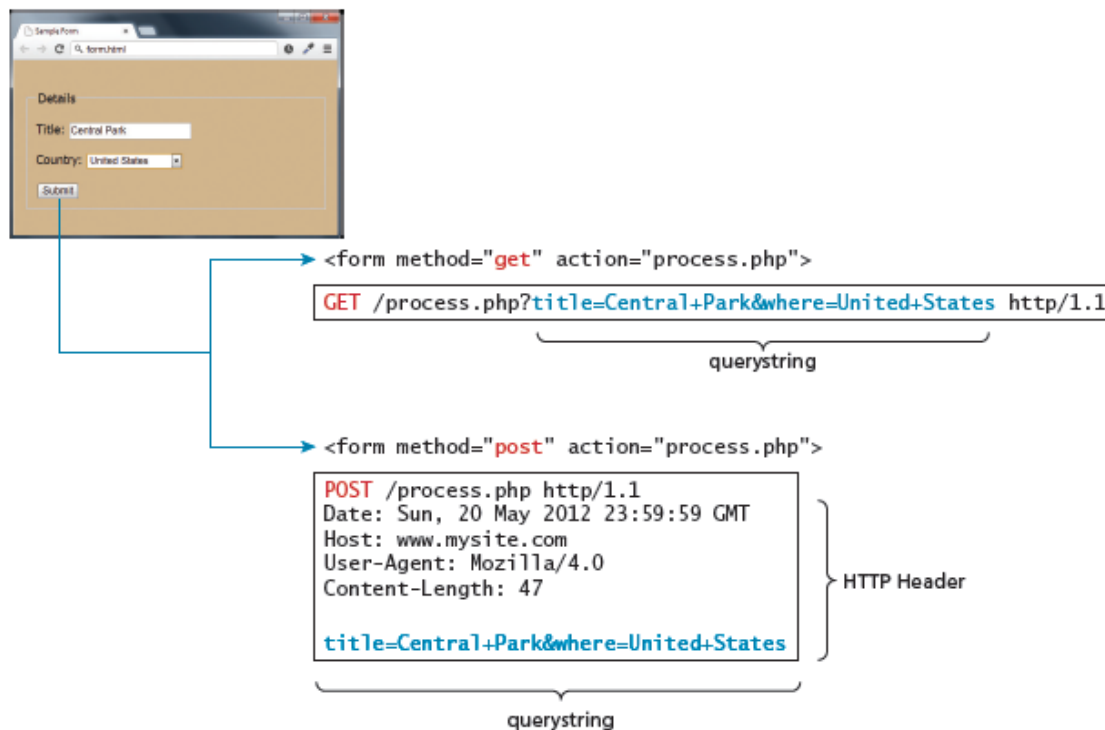


Example: Suppose the user a request for an HTML page from the server (1) that contains some type of form on it. This could be something as complex as a user registration form or as simple as a search box. After the user fills out the form, there needs to be some mechanism for submitting the form data back to the server. This is typically achieved via a **submit** button. As interaction between the browser and the web server is governed by the HTTP protocol, the form data must be sent to the server via a standard HTTP request. This request is typically some type of server-side program that will process the form data in some way; this could include checking it for validity, storing it in a database, or sending it in an email.

The <form> Element

- The HTML form contains two important attributes that are essential features of any form, namely the **action** and the **method** attributes.
- Action** attribute specifies the **URL of the server-side resource** that will process the form data. This could be a resource on the same server as the form or a completely different server.
- 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.

The use of GET or POST method decides where the browser locates the user input in the HTTP request. Using **GET**, the browser locates the data in the URL of the request; with **POST**, the form data is located in the HTTP header after the HTTP variables.



	GET Method	POST Method
1	Clearly seen in the address bar	Data is hidden from the user
2	Usually used during development, for testing	Used once the product is ready
3	Data remains in browser history and cache.	Submitted data is not stored in browser history or cache.
4	Limits on the number of characters in the form.	There is no limit on number of characters entered in form.

Generally, form data is sent using the POST method. However, the GET method is useful when you are testing or developing a system, since you can examine the query string directly in the browser's address bar.

2.4 Form Control Elements

Some of the form related HTML elements are –

Type	Description
<code><button></code>	Defines a clickable button.
<code><datalist></code>	An HTML5 element that defines lists of pre-defined values to use with input fields.
<code><fieldset></code>	Groups related elements in a form together.
<code><form></code>	Defines the form container.
<code><input></code>	Defines an input field. HTML5 defines over 20 different types of input.
<code><label></code>	Defines a label for a form input element.
<code><legend></code>	Defines the label for a fieldset group.
<code><option></code>	Defines an option in a multi-item list.
<code><optgroup></code>	Defines a group of related options in a multi-item list.
<code><select></code>	Defines a multi-item list.
<code><textarea></code>	Defines a multiline text entry box.

Text Input Controls

Text Input Controls – used to get text information from the user. It is used in search box to enter search element, in a login form to enter the name, in user registration form to enter name, parents name, caste etc. Different text input controls are –

Type	Description
text	Creates a single-line text entry box. <code><input type="text" name="title" /></code>
textarea	Creates a multiline text entry box. <code><textarea rows="3" cols="50" /></code>
password	Creates a single-line text entry box for a password (which masks the user entry as bullets or some other character) <code><input type="password" ... /></code>
email	Creates a single-line text entry box suitable for entering an email address. This is an HTML5 element. Some browsers will perform validation when form is submitted. <code><input type="email" ... /></code>
tel	Creates a single-line text entry box suitable for entering a telephone. This is an HTML5 element. Since telephone numbers have different formats in different parts of the world, current browsers do not perform any special

	formatting or validation. Some devices may, however, provide a specialized keyboard for this element. <input type="tel" ... />
url	Creates a single-line text entry box suitable for entering a URL. This is an HTML5 element. Browsers perform validation on submission. <input type="url" ... />


Choice Controls


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

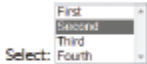
Select Lists

- 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.
- Option items can be grouped together via the <optgroup> element. The selected attribute in the <option> makes it a default value.
- The value attribute of the <option> element is used to specify what value will be sent back to the server in the query string when that option is selected.
- The value attribute is optional; if it is not specified, then the text within the container is sent.

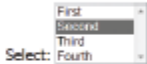
The new <datalist> element is a new addition to HTML5. This element allows to define a list of values that can appear in a drop-down autocomplete style list for a text element. This can be helpful for situations in which the user must have the ability to enter anything, but are often entering one of a handful of common elements.








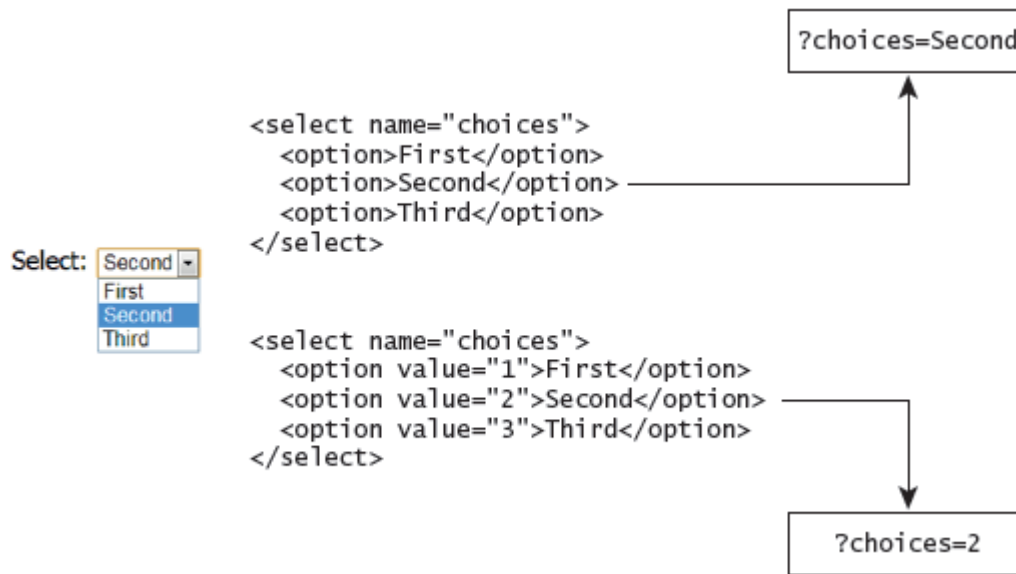
```
<select name="choices">
  <option>First</option>
  <option selected>Second</option>
  <option>Third</option>
</select>
```



```
<select size="3" ... >
```



```
<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>
```



```

<input type="text" name="city" list="cities" />
<datalist id="cities">
  <option>Calcutta</option>
  <option>Calgary</option>
  <option>London</option>
  <option>Los Angeles</option>
  <option>Paris</option>
  <option>Prague</option>
</datalist>

```

Radio Buttons

Radio buttons are used when the user has to select a single item from a small list of choices and all the choices have to be visible. They are added by using the `<input type="radio">` element. The buttons are made mutually exclusive (i.e., only one can be chosen) by sharing the same name attribute. The checked attribute is used to indicate the default choice, while the value entered in value attribute is sent to the server when submit button is clicked.

Eg:

```

<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

```

Continent:

- ☐ North America
- ☒ South America
- ☐ Asia

Checkboxes

Checkboxes are used for getting yes/no or on/off responses from the user. Checkboxes are added by using the `<input type="checkbox">` element. You can also group checkboxes together by sharing the same name attribute. Each checked checkbox will have its value sent to the server.

I accept the software license ☒ `<input type="checkbox" name="accept" >`

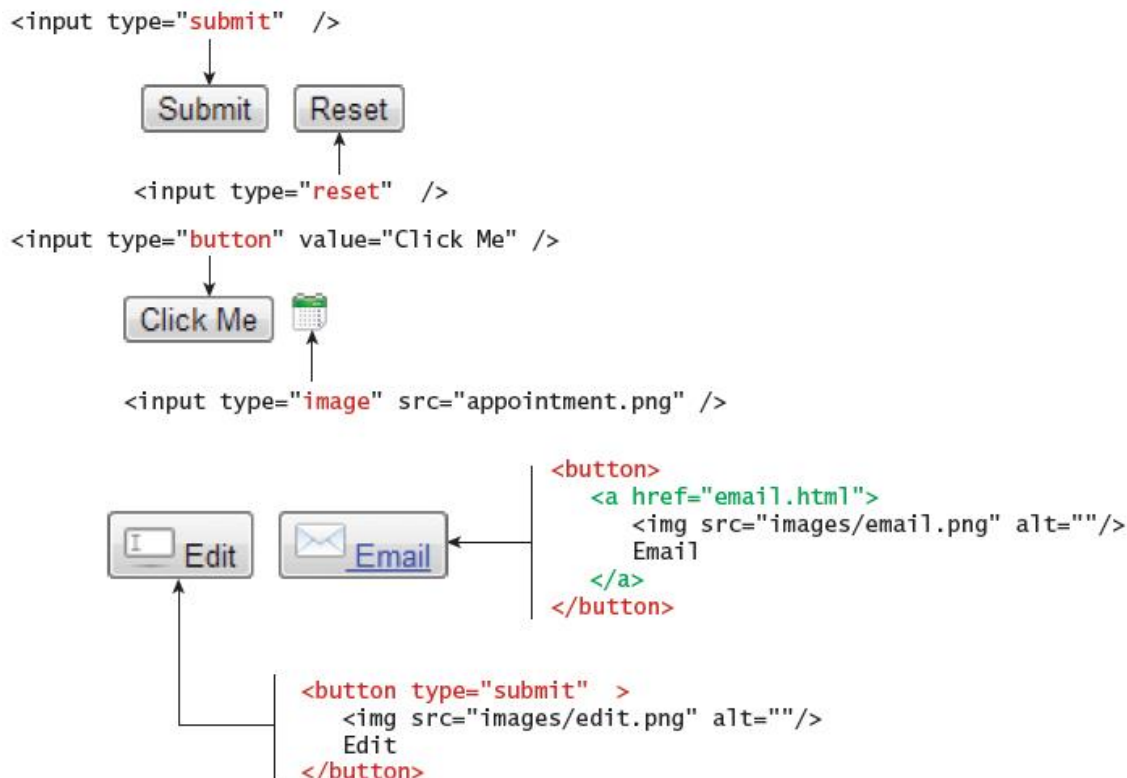
Where would you like to visit? `<label>Where would you like to visit? </label>
`
☒ Canada `<input type="checkbox" name="visit" value="canada">Canada
`
☐ France `<input type="checkbox" name="visit" value="france">France
`
☒ Germany `<input type="checkbox" name="visit" value="germany">Germany`

`?accept=on&visit=canada&visit=germany`

Button Controls

HTML defines several different types of buttons.

Type	Description
<code><input type="submit"></code>	Creates a button that submits the form data to the server
<code><input type="reset"></code>	Creates a button that clears the user's already entered form data.
<code><input type="button"></code>	Creates a custom button. This button may requires a script for it to actually perform any action
<code><input type="image"></code>	Creates a custom submit button that uses an image for its display
<code><button></code>	Creates a custom button. The <code><button></code> element differs from <code><input type="button"></code> in that you can completely customize what appears in the button. It can be used to include both images and text.



Date and Time Controls

The user entering of date or time may need to be validated, as it may cause error when manually entered. The new HTML date and time controls make it easier for users to input these tricky date and time values. The format output of all date & time controls vary depending on the browser.

Type	Description
date	Creates a general date input control. The format for the date is “mm-dd-yyyy”.
time	Creates a time input control. The format for the time is “HH:MM AM/PM,” for hours:minutes
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 “mm- yyyy.”
week	Creates a control in which the user can specify a week in a year. The format is “W##- yyyy.”

Date:

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

Time:

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

DateTime:

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

DateTime Local:

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

Month:

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

Week:

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

2.5 Table and Form Accessibility

Users with sight disabilities, for instance, experience the web using voice reading software. Color blind users might have trouble differentiating certain colors in proximity; users with muscle control problems may have difficulty using a mouse, while older users may have trouble with small text and image sizes.

The term **web accessibility** refers to the assistive technologies, various features of HTML that work with those technologies, and different coding and design practices that can make a site more usable for people with visual, mobility, auditory, and cognitive disabilities.

In order to improve the accessibility of websites, the W3C created the **Web Accessibility Initiative (WAI)** in 1997. The WAI produces guidelines and recommendations, as well as organizing different working groups on different accessibility issues.

Perhaps the most important guidelines for web accessibility 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.*
- *Provide ways to help users navigate, find content, and determine where they are.*

The guidelines provide detailed recommendations on how to achieve this advice.

Accessible Tables

HTML tables can be quite frustrating, for people with visual disability. One important way to improve the accessibility is to only use tables for tabular data, not for layout. Using the following accessibility features for tables in HTML can improve the using of tables for those users:

1. Describe the table's content using the <caption> element. This provides the user with the ability to discover what the table is about before having to listen to the content of each and every cell in the table.
2. Connect the cells with a textual description in the header. It is quite revealing to listen to reader software recite the contents of a table that has not made these connections. It sounds like this: "row 3, cell 4: 45.56; row 3, cell 5: Canada; row 3, cell 6: 25,000; etc." However, if these connections have been made, it sounds instead like this: "row 3, Average: 45.56; row 3, Country: Canada; row 3, City Count: 25,000; etc.,".

Accessible Forms

HTML forms are also potentially problematic with respect to accessibility. The use of the <fieldset>, <legend>, and <label> elements, provide a connection between the input elements in the form.

The main purpose of <fieldset> and <legend> 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, by using the 'for' attribute. So that if the user clicks on or taps the <label> text, that control will receive the form's focus (i.e., it becomes the current input element and any keyboard input will affect that control).

Associating label and the input tag -

```
<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>
```

2.6 Microformats

The web has millions of pages and there are many similar information from site to site. Most sites have Contact Us page, in which addresses and other information are displayed; calendar of upcoming events or information about products or news. These types of common information can be tagged in a similar way, and automated tools can be used to gather and transform the information.

A **microformat** is a small pattern of HTML markup and to represent common blocks of information such as people, events, and news stories so that the information in them can be extracted and indexed by software agents.