

MODULE-2

HTML Tables and Forms

INTRODUCING TABLES

HTML Tables

- A table in HTML is created using the **<table>** element Tables can be used to display:
 - Many types of content
 - Calendars, financial data, list, etc...
 - Any type of data
 - Images
 - Text
 - Links
 - Other tables

Table Basics

- an HTML **<table>** contains any number of rows (**<tr>**)
- each row contains any number of table data cells (**<td>**)
- Content goes inside of **<td></td>** tags

<table>

<tr>

<td>The Death of Marat</td>

</tr>

</table>

A basic Example

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

```

<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>

```



Title	Artist	Year	Width	Height
The Death of Marat	Jacques-Louis David	1793	162cm	128cm
Burial at Ornans	Gustave Courbet	1849	314cm	663cm

```

<table>
  <tr>
    <th>Title</th>
    <th>Artist</th>
    <th>Year</th>
    <th>Width</th>
    <th>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>
  <tr>
    <td>Burial at Ornans</td>
    <td>Gustave Courbet</td>
    <td>1849</td>
    <td>314cm</td>
    <td>663cm</td>
  </tr>
</table>

```



Spanning Rows and Columns

Each row must have the same number of `<td>` or `<th>` containers. If you want a given cell to cover several columns or rows, use the **colspan** or **rowspan** attributes

Title	Artist	Year	Size (width x height)	
The Death of Marat	Jacques-Louis David	1793	162cm	128cm
Burial at Ornans	Gustave Courbet	1849	314cm	663cm

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>

```

Artist	Title	Year
Jacques-Louis David	The Death of Marat	1793
	The Intervention of the Sabine Women	1799
	Napoleon Crossing the Alps	1800

Notice that these two rows now only have two cell elements.

```

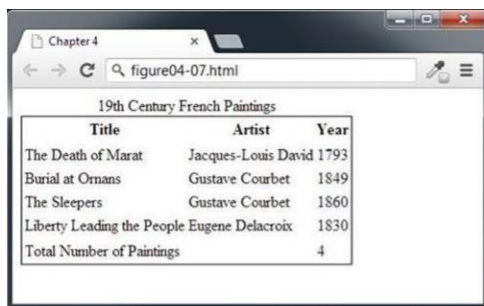
<table>
  <tr>
    <th>Artist</th>
    <th>Title</th>
    <th>Year</th>
  </tr>
  <tr>
    <td rowspan="3">Jacques-Louis David</td>
    <td>The Death of Marat</td>
    <td>1793</td>
  </tr>
  <tr>
    <td>The Intervention of the Sabine Women</td>
    <td>1799</td>
  </tr>
  <tr>
    <td>Napoleon Crossing the Alps</td>
    <td>1800</td>
  </tr>
  ...
</table>

```

STYLING TABLES

In HTML5 it is left to CSS, However legacy support for deprecated HTML attributes still exist

- **width, height**—for setting the width and height of cells
- **cell spacing**—for adding space between every cell in the table
- **cell padding**—for adding space between the content of the cell and its border
- **bgcolor**—for changing the background color of any table element



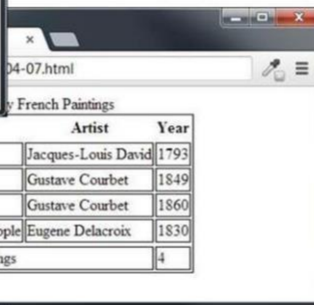
Chapter 4

figure04-07.html

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
Total Number of Paintings	4	

```
table {
  border: solid 1pt black;
}
```



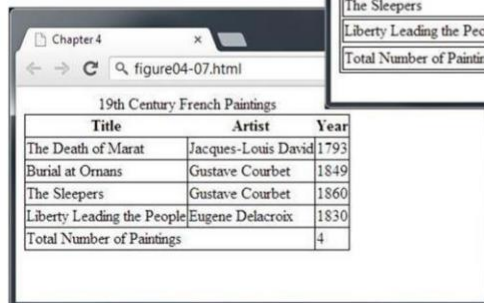
Chapter 4

figure04-07.html

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
Total Number of Paintings	4	

```
table {
  border: solid 1pt black;
}
td {
  border: solid 1pt black;
}
```



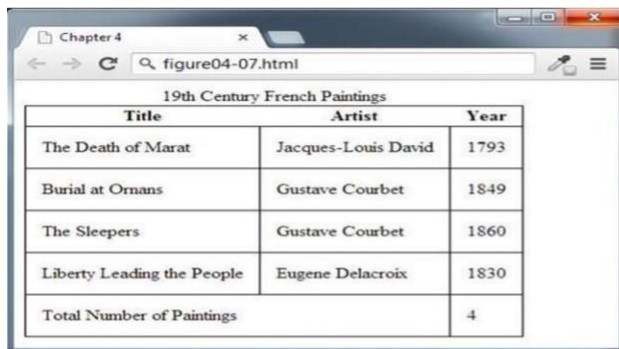
Chapter 4

figure04-07.html

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
Total Number of Paintings	4	

```
table {
  border: solid 1pt black;
  border-collapse: collapse;
}
td {
  border: solid 1pt black;
}
```



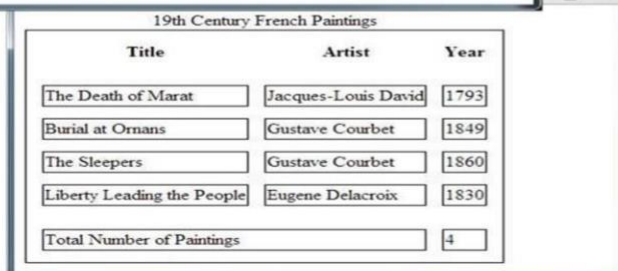
Chapter 4

figure04-07.html

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
Total Number of Paintings	4	

```
table {
  border: solid 1pt black;
  border-collapse: collapse;
}
td {
  border: solid 1pt black;
  padding: 10pt;
}
```



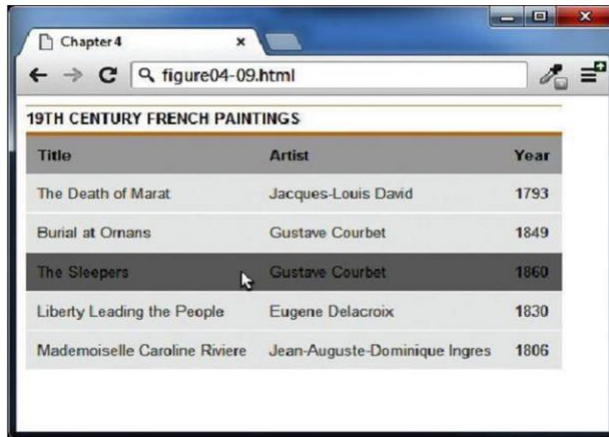
Chapter 4

figure04-07.html

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
Total Number of Paintings	4	

```
table {
  border: solid 1pt black;
  border-spacing: 10pt;
}
td {
  border: solid 1pt black;
}
```



Chapter 4 x

figure04-09.html

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



Chapter 4 x

figure04-09.html

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;
}
```

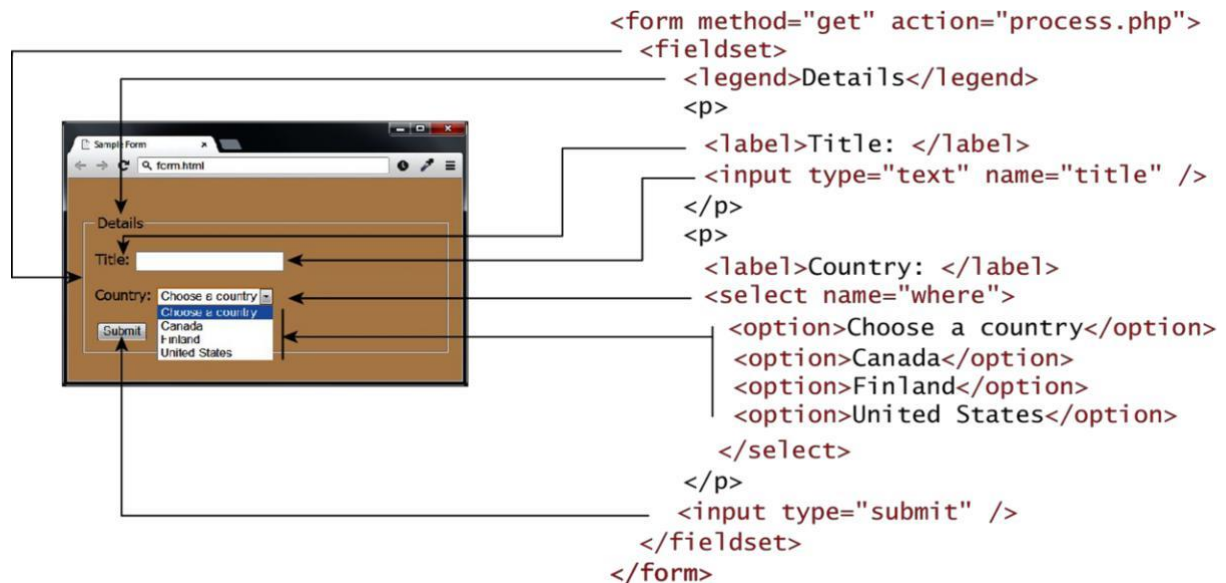
VTUPulse.com

HTML Forms

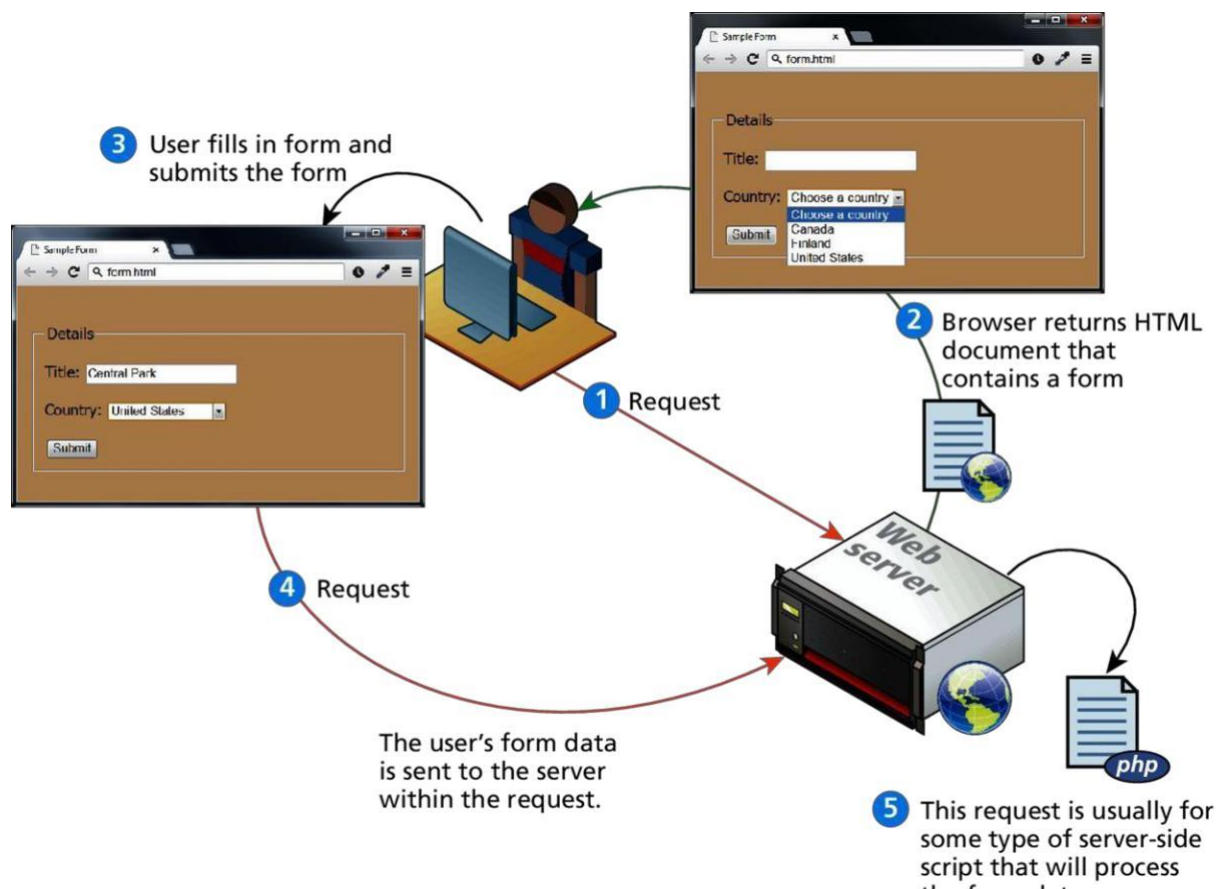
Forms provide the user with an alternative way to interact with a web server.

- Forms provide rich mechanisms like:
 - Text input
 - Password input
 - Options Lists
 - Radio and check boxes

Form Structure



How forms interact with servers



<form> element

Two essential features of any form, namely the **action** and the **method** attribute.

- 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.
 - GET
 - POST

GET vs POST



Advantages and Disadvantages

- Data can be clearly seen in the address bar.
- Data remains in browser history and cache.
- Data can be bookmarked
- Limit on the number of characters in the form data returned.

POST

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

FORMS CONTROL ELEMENTS

Type	Description
<button>	Defines a clickable button.
<datalist>	An HTML5 element form defines lists to be used with other form elements.
<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.
<option>	Defines an option in a multi-item list.
<optgroup>	Defines a group of related options in a multi-item list.
<select>	Defines a multi-item list.
<textarea>	Defines a multiline text entry box.

Text Input Controls

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" ... /></code>
password	Creates a single line text entry box for a password <code><input type="password" ... /></code>
search	Creates a single-line text entry box suitable for a search string. This is an HTML5 element. <code><input type="search" ... /></code>
email	Creates a single-line text entry box suitable for entering an email address. This is an HTML5 element. <code><input type="email" ... /></code>
tel	Creates a single-line text entry box suitable for entering a telephone. This is an HTML5 element. <code><input type="tel" ... /></code>
url	Creates a single-line text entry box suitable for entering a URL. This is an HTML5 element. <code><input type="url" ... /></code>

Select List

- `<select>` element is used to create a multi line box for selecting one or more items
 - The options are defined using the `<option>` element
 - can be hidden in a dropdown or multiple rows of the list can be visible
- Option items can be grouped together `<optgroup>` element.

Select:

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

Select:
First
Second
Third

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

Select:
First
Second
Third
Fourth

```
<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>
```

Cities:
North America
Calgary
Los Angeles
Europe
London
Paris
Prague

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 buttons are 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
- the value attribute works in the same manner as with the `<option>` element

Continent:

☐ North America
☒ South America
☐ Asia

```
<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
```

Checkboxes

Checkboxes are used for getting yes/no or on/off responses from the user.

- checkboxes are added via the `<input type="checkbox">`

Element

- You can also group checkboxes together by having them share the same name attribute
- Each checked checkbox will have its value sent to the server
- Like with radio buttons, the checked attribute can be used to set the default value of a checkbox

I accept the software license ☒

Where would you like to visit?

☒ Canada

☐ France

☒ Germany

```

<label>I accept the software license</label>
<input type="checkbox" name="accept" >

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

```

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

Button

Controls 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 any of the user's already entered form data.
<code><input type="button"></code>	Creates a custom button. This button may require Javascript for it to actually perform any action.
<code><input type="image"></code>	Creates a custom submit button that uses an image for its display.

<button>

Creates a custom button. The <button> element differs from <input type="button"> in that you can completely customize what appears in the button; using it, you can, for instance, include both images and text, or skip

server-side processing entirely by using hyperlinks.

You can turn the button into a submit button by using the type="submit" attribute.

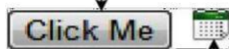
```
<input type="submit" />
```



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



```
<input type="button" value="Click Me" />
```

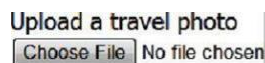


```
<input type="image" src="appointment.png" />
```



```
<button type="submit" >
  
  Edit
</button>
```

```
<button>
  <a href="email.html">
    
    Email
  </a>
</button>
```

**Specialized Controls****<input type="hidden">****<input type="file">**

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

Date and Time Controls

From a user's perspective, entering dates can be tricky as well: you probably have wondered at some point in time when entering a date into a web form, what format to enter it in, whether the day comes before the month, whether the month should be entered as an abbreviation or a number, and so on.

Date:

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

Time:

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

DateTime:

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

DateTime Local:

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

TABLE AND FORM ACCESSIBILITY

Web Accessibility

- ❖ Not all web users are able to view the content on web pages in the same manner.
- ❖ 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)**
 - [Web Content Accessibility Guidelines](#)

Accessible Tables

1. Describe the table's content using the <caption> element
2. Connect the cells with a textual description in the header

```
<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>
```

MICROFORMATS

- ❖ The web has millions of pages in it. Yet despite the incredible variety, there is a surprising amount of similar information from site to site.
- ❖ Most sites have some type of Contact us pages.
- ❖ Similarly, many sites contain calendar of upcoming events or information about products or news.
- ❖ The idea behind microformats is that if this type of common information were tagged in similar ways, then automated tools would be able to gather and transform it.
- ❖ A **microformat** is a small pattern of HTML markup and attributes 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.
- ❖ One of the most common microformat is hcard.

Advanced CSS: Layout

NORMAL FLOW

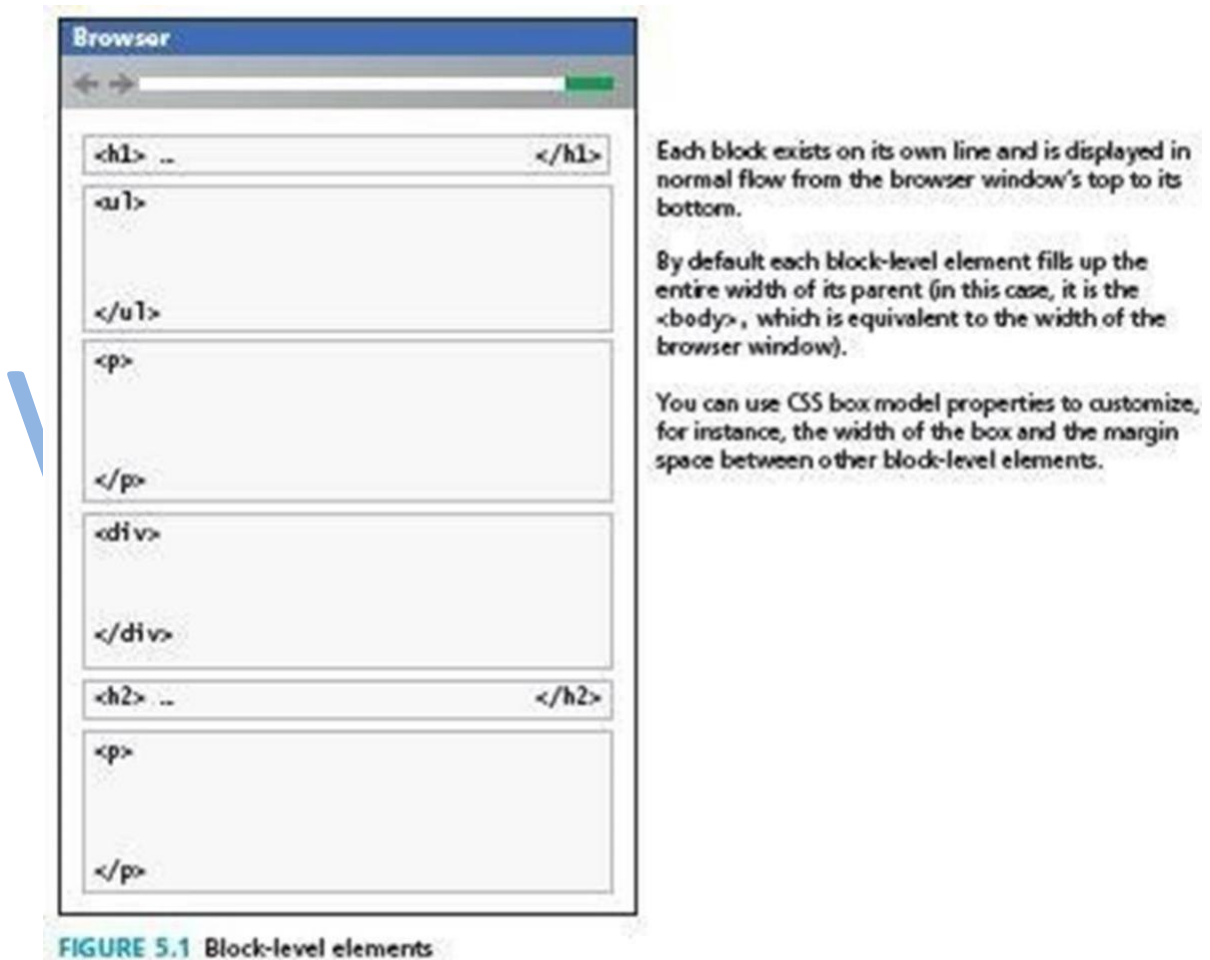
Normal flow, which refers here to how the browser will normally display block

Level elements and inline elements

Block Level elements such as p, div, table, ul and table are each contained on their own line.

They begin with a line break.

2 block level elements can't exist on the same line as in fig



- Inline elements do not form their own blocks but instead are displayed within lines.
- Normal text in an html document is in line, as are elements such as `em`, `a`, `tag`, and `span`.
- Inline elements line up next to one another horizontally from left to right on the same line.

- When there is no space left on the line the content moves to new line as in figure
- Inline elements are actually two types.
 - Replaced Inline elements.
 - Content and appearance is defined by some external resource,
Ex – tag and some form elements
 - Nonreplaced Inline elements.
 - Content defined within the document, which includes all other inline elements.
- In a Document with normal flow, block level elements and inline elements work together as in figure.
 - Block level will flow from top to bottom
 - Inline level will flow from left to right

VTUPulse.com

POSITIONING ELEMENTS

The position property is used to specify the type of positioning
and The possible values are

Value	Description
absolute	The element is removed from normal flow and positioned in relation to its nearest positioned ancestor.
fixed	The element is fixed in a specific position in the window even when the document is scrolled.
relative	The element is moved relative to where it would be in the normal flow.
static	The element is positioned according to the normal flow. This is the default.

TABLE 5.1 Position Values

Relative positioning

- Element is displaced out of its normal flow position and moved relative to where it would have been placed.
- When an element is positioned relatively, it is displaced out of its normal flow position and moved relative to where it would have been placed.
- The other content around the relatively positioned element —remembers the element old position in the flow, thus the space the element would have occupied as in fig



```
<p>A wonderful serenity has taken possession of my ...
<figure>
  
  <figcaption>British Museum</figcaption>
</figure>
<p>When, while the lovely valley ...
```



```
figure {
  border: 1pt solid #A5A5A5;
  background-color: #EDEDDE;
  padding: 5px;
  width: 150px;
  position: relative;
  top: 150px;
  left: 200px;
}
```

FIGURE 5.4 Relative positioning

Absolute positioning

- When an element is positioned absolutely, it is removed completely from normal flow.
- Thus, unlike with relative positioning, space is not left for the moved element, as it no longer in the normal flow.
- Its position is moved in relation to its container block

Z – Index

- Looking at above fig, you may wonder what would have happened if the <figcaption> had been moved so that it overlapped the <figure>.
- Each positioned element has a stacking order defined by the z – index property.
- Items closest to the viewer have a larger z-index.
- Working with z- index can be tricky.
- First, only positioned elements will make use of their z-index.
- Second, as in below fig, simply setting the z-index value of elements will not necessarily move them on top or behind other items.

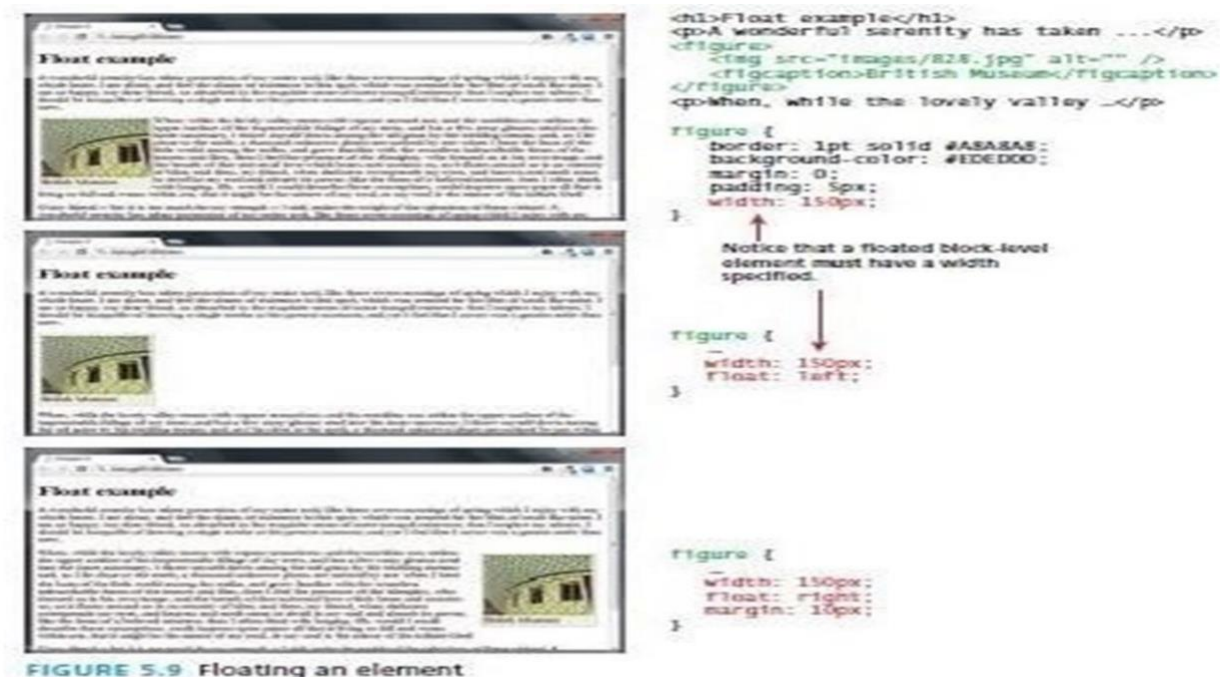
Fixed

- The fixed position value is used relatively infrequently.
- It's a type of absolute positioning, except that the positioning values are in relation to the viewport.

Elements with fixed positioning do not move when the user scrolls up or down the page

FLOATING ELEMENTS.

- It is possible to displace an element out of its position in the normal flow via the css float property.
- It means to far left or far right of its containing block and rest of the content is —re-flowed around the floated element, as in below fig



Overlaying and Hiding elements

One of the most common design tasks with CSS is to place two elements on top of each other or to selectively hide and display elements. Positioning is important for both tasks.

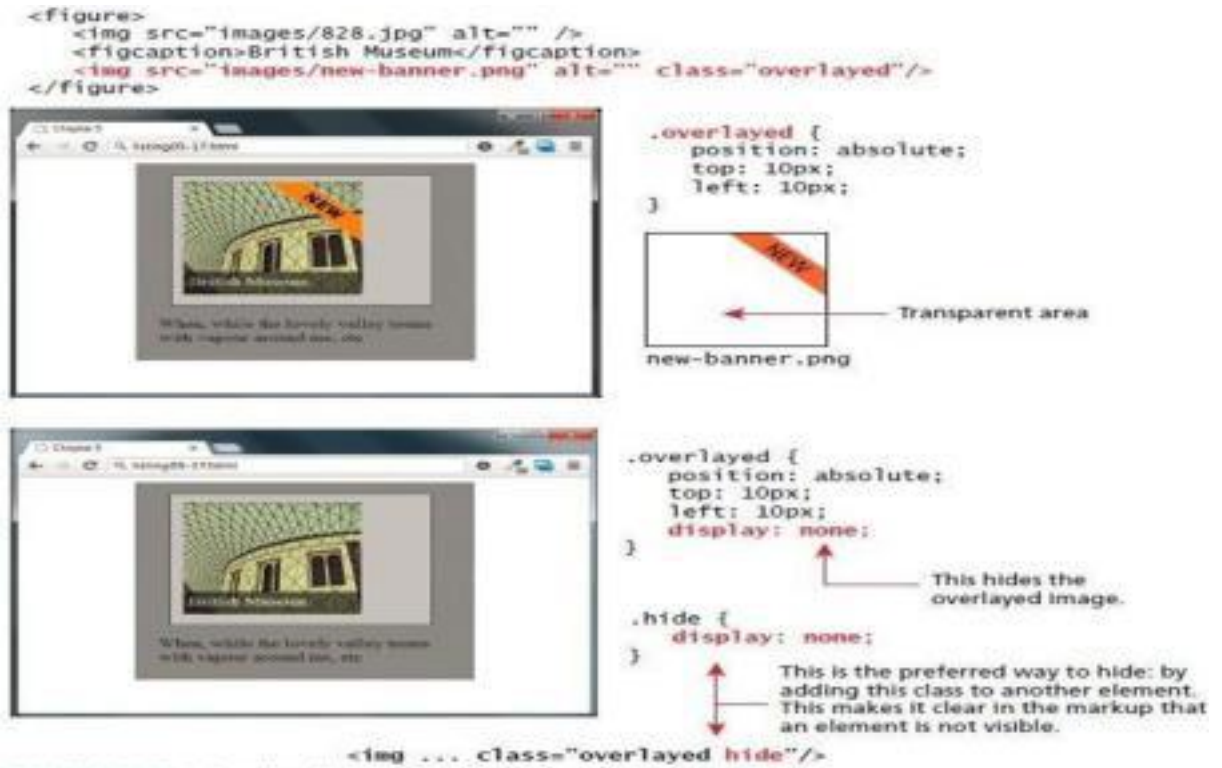


FIGURE 5.17 Using the display property

There are two different ways to hide elements in css:

- Display property
 - It takes an item out of the flow – element no longer exists
- Visibility property
 - Hides the element, but the space for that element remains

CONSTRUCTING MULTICOLUMN LAYOUTS

The previous sections showed two different ways to move items out of the normal top-down flow, namely, by using positioning and by using floats



FIGURE 5.20 Creating two-column layout, step one

APPROACHES TO CSS LAYOUT

One of the main problems faced by web designers is that the size of the screen used to view the page can vary quite a bit.

- 21-inch wide screen monitor that can display 1920 x 1080 pixels
- Older iPhone with a 3.5 screen and a resolution of 320x480 px

Satisfying both users can be difficult; the approach to take for one type of site content might not work as well with another site with different content.

Most designers take one of two basic approaches to dealing with the problems of screen size.

- Fixed Layout
- Liquid Layout
- Hybrid Layout.

Fixed Layout

In a fixed layout, the basic width of the design is set by the designer, typically corresponding to an ideal width based on a typical monitor resolution

The advantage of a fixed layout is that it is easier to produce and generally has a predictable visual result.

Fixed layouts have drawbacks.

- For larger screens, there may be an excessive amount of blank space to the left and/or right of the content.
- It is also optimized for typical desktop monitors; however, as more and more user visits are happening via smaller mobile devices

Liquid Layout

In a liquid layout (also called a fluid layout) widths are not specified using pixels, but percentage values.

Advantage:

- adapts to different browser sizes,

Disadvantages:

- Liquid layouts can be more difficult to create because some elements, such as images, have fixed pixel sizes
- The line length (which is an important contributing factor to readability) may become too long or too short.

Hybrid Layout

A hybrid layout combines pixel and percentage measurements.

- Fixed pixel measurements might make sense for a sidebar column containing mainly graphic advertising images that must always be displayed and which always are the same width.
- percentages would make more sense for the main content or navigation areas, with perhaps min and max size limits in pixels set for the navigation areas

Responsive design

- In the past several years, a lot of attention has been given to so call Responsive layout designs.
- In a responsive design, the page “responds” to changes in the browser size that go beyond the width scaling of a liquid layout.
- We had problems with liquid layout for images.
- In a responsive design layout, images will be scaled down and navigation elements will be replaced as the browser shrinks.

There are 4 important key components to make responsive design work

1. Liquid layouts.
2. Scaling images to viewport size.
3. Setting viewports via the <meta> tag
4. Customizing the css for different viewports using media queries.

Setting Viewport

The web page will tell the mobile browser the viewport size to use via the viewport <meta> element, as below

```
<html>
```

```
<head>
```

```
<meta name="viewport" content="width=device-width"/>
```

(Setting the viewport)

By above, the page is telling the browser there is no need of any scaling and it makes the viewport as many pixels wide as device screen width.

Ex- if device has a screen that is 320 px wide, then the viewport width will be 320 px, if its 480px wide, then viewport will be 480px.

Media Queries

- The other key component of responsive design is CSS media queries.
- A media query is a way to apply style rules based on the medium that is displaying the file.
- Syntax of media queries.

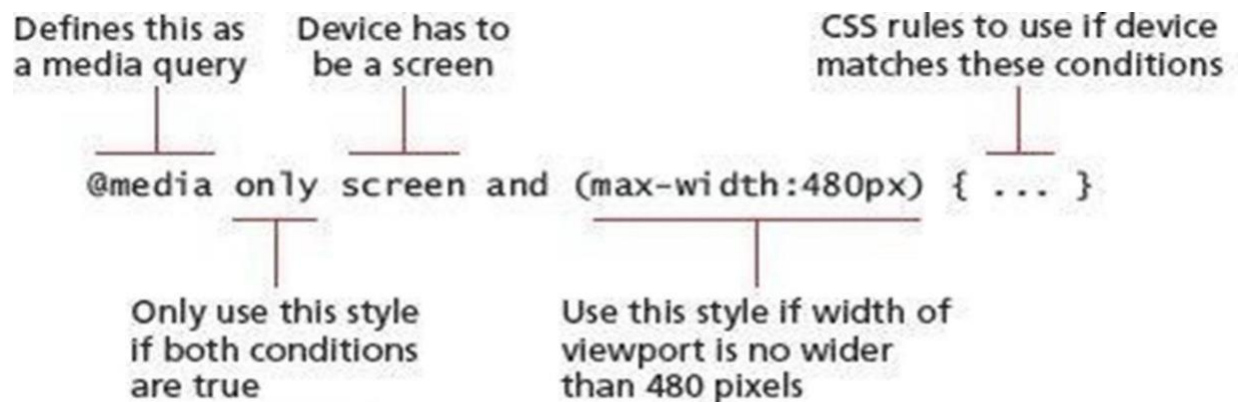


FIGURE 5.33 Sample media query

These queries are Boolean expressions and can be added to your css file or to the link element to conditionally use a different external CSS file based .

CSS FRAMEWORK

A CSS framework is a pre created set of CSS classes or other software tools that make it easier to use and work with CSS.

They are two main types of CSS framework:

1. Grid systems
2. CSS preprocessors.

- Grid systems make it easier to create multicolumn layouts.
- There are many CSS grid systems; some of the most popular are Bootstrap (twitter.github.com/bootstrap), Blueprint (www.blueprintcss.org), and 960 (960.gs).
- The most important of these capabilities is a grid system.
- CSS frameworks provide similar grid features. The 960 framework uses either a 12- or 16-column grid.
- Bootstrap uses a 12-column grid.
- Blueprint uses a 24-column grid.

CSS preprocessor

- CSS preprocessors are tools that allow the developer to write CSS that takes advantage of programming ideas such as variables, inheritance, calculations, and functions.
- A CSS preprocessor is a tool that takes code written in some type of preprocessed language and then converts that code into normal CSS
- The advantage of a CSS preprocessor is that it can provide additional functionalities that are not available in CSS.

VTUPulse.com