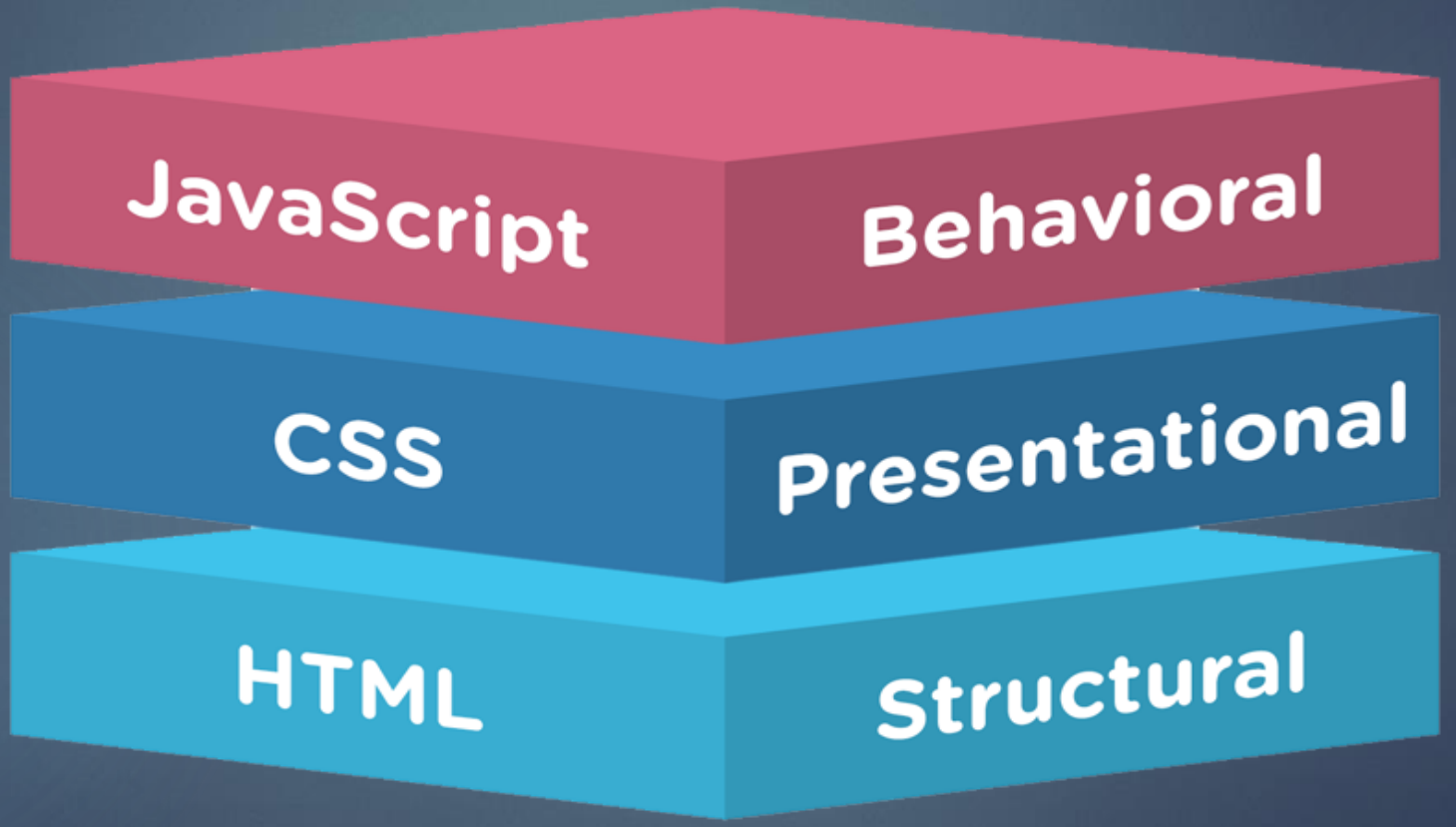
HTML , CSS

Sujata Batra

HTML vs CSS vs JAVASCRIPT

Sujata Batra

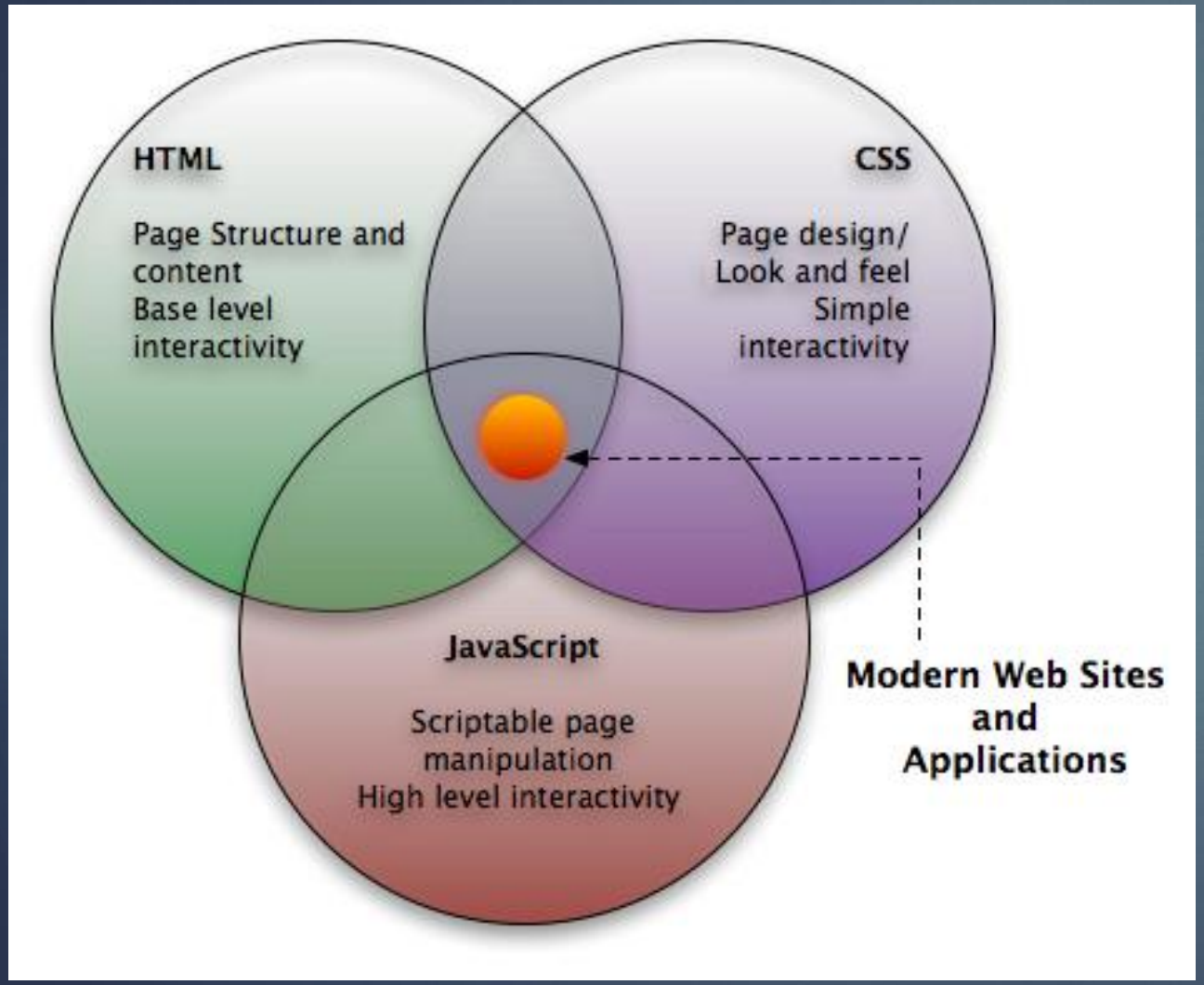
Overview

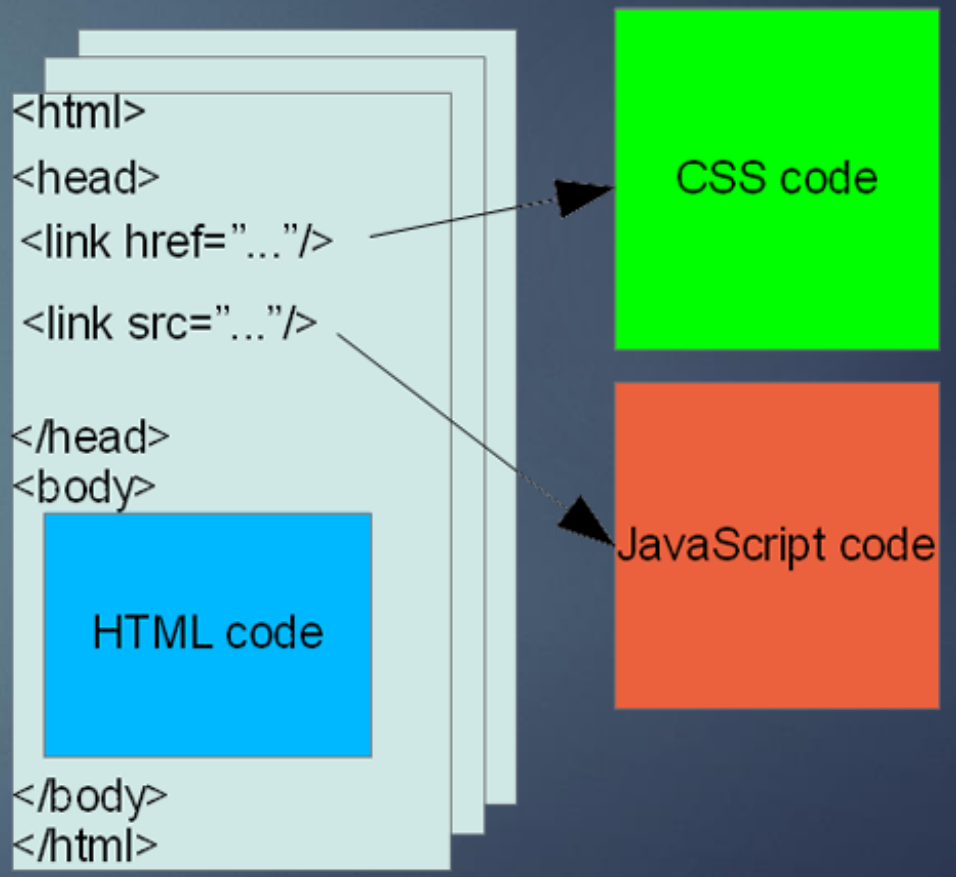


HTML vs CSS vs JAVASCRIPT

Sujata Batra

Overview





What is HTML?

- Tim Berners-Lee was the author of html, with his team at CERN.
- The HTML that Tim invented was strongly based on SGML (Standard Generalized Mark-up Language).
- Hypertext Markup Language (First Version of HTML) was formally published on June 1993.
- Platform independent.
- HTML is a format that tells a computer how to display a web page.
- The documents themselves are plain text files (ASCII) with special "tags" or codes that a web browser knows how to interpret and display on your screen.

Sujata Batra

HTML Versions

- HTML 2.0
  - HTML 2.0 was developed by the Internet Engineering Task Force HTML Working Group in 1996.
- HTML 3.2
- HTML 4.0
- HTML 4.01
  - HTML 4.01 was released as a W3C Recommendation 24. December 1999.
  - HTML 4.01 is a minor update of corrections and bug fixes in HTML 4.0.
  - W3C will not continue to develop HTML. Future W3C work will be focusing on XHTML.

- XHTML 1.0
  - XHTML 1.0 reformulates HTML 4.01 in XML.
  - XHTML 1.0 was released as a W3C Recommendation 20. January 2000.
- HTML5
  - Web Hypertext Application Technology Working Group(WHATWG) and W3C came up with this in 2007
  - HTML 5 is a combination of HTML 4.01 and XHTML 1.0.
  - HTML 5 is backwards compatible.

Sujata Batra

Overview

1. Advance version of HTML.
2. In 2008, the first HTML5 public draft was released
3. HTML5 W3C Final Recommendation was released 28. October 2014.
4. New elements, attributes, and behaviors were introduced.
5. It helps to create more powerful website and interactive web applications.
6. HTML5 comes with XML syntax.
7. HTML5 is to compete with Flash and Silverlight.
8. Empowering Mobile devices.

Technical Advantages Over Previous Version.

1. Audio and Videos are integral part of HTML5 specifications e.g. <audio> and<video> tags.
2. Vector graphics is integral part of HTML5 e.g. SVG and canvas.
3. JS GeoLocation API in HTML5 helps identify location of user browsing any website (provided user allows it).
4. Full duplex communication channels can be established with Server using Web Sockets.
5. Allows JavaScript to run in background. This is possible due to JS Web worker API in HTML5.
6. Application Cache, Web SQL database and Web storage is available as client side storage.
7. Retain Backward Compatibility with previous versions of HTML5.

HTML5 Technology Functions

Semantics: allowing you to describe more precisely what your content is.

Connectivity: allowing you to communicate with the server in new and innovative ways.

Offline & Storage: allowing webpages to store data on the client-side locally and operate offline more efficiently.

Multimedia: making video and audio first-class citizens in the Open Web.

2D/3D Graphics & Effects: allowing a much more diverse range of presentation options.

Performance & Integration: providing greater speed optimization and better usage of computer hardware.

Device Access: allowing for the usage of various input and output devices.

Styling: letting authors write more sophisticated themes.

You can write your HTML code in almost any available text editor, including notepad.

Open source text editor

Brackets http://brackets.io/ Notepad++ https://notepad-plus-plus.org/

or

Eclipse which has built-in HTML Editor.

HTML Document will always be saved in **.html** extension or an **.htm** extension.

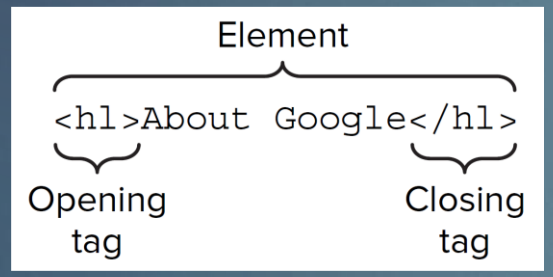
HTML Tags and Elements

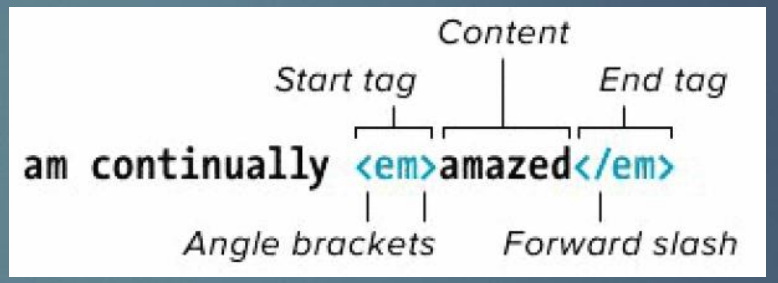
Tags are enclosed in angle brackets < >

For Eg.: <html> Opening Tag, **</html>** Closing Tag.

Element is the combination of (opening & closing Tags and the content between them).

For Eg.:





Part of this text is bold. is a PARAGRAPH element that contains a BOLD element

An HTML document is a collection of elements (text/media with context).

Empty tags vs Container tags

Some elements which does not requires **closing tags**, are known as **Empty Tags** or **Elements**.

For Eg.:

 begining of new line. **BR** stands for **BReak**.

<hr > puts a line across the page. **HR** stands for **H**orizontal **R**ule.

The elements which requires **opening** and **closing tags**, are known as **Container Tags** or **Elements**.

For Eg.: <h1> This is a heading </h1>

 This is a paragraph

HTML Attributes and Values

HTML elements can have attributes which provides additional information about an element.

Always specified in the opening tag and should contained value.

For Eg.:


```
<!DOCTYPE html>
<html>
<head>
<title>Align Attribute  Example</title>
</head>
<body>
<p align="left">This is left aligned</p>
<p align="center">This is center aligned</p>
<p align="right">This is right aligned</p>
</body>
</html>
```

This is left aligned

This is center aligned

This is right aligned

Structural Elements

A standard HTML document has two main structural elements

head contains setup information for the browser & the Web page

For E.g., the title for the browser window, style definitions, JavaScript code, …

body contains the actual content to be displayed in the Web page

```
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>My first HTML document</title>
</head>
  <body>
    <p> Hello world! </p>
  </body>
</html>
```

Comments and doctype

HTML has a mechanism for embedding comments that are not displayed when the page is rendered in a browser.

Eg.: <!-- This is comment text -->

Besides tags, text content, and entities, an HTML document must contain a doctype declaration as the **first line**. **For**

Eg.:

```
<!DOCTYPE html>
<html lang="en">
<head>
  <meta charset="UTF-8" />
  <title>My first HTML document</title>
</head>
  <body>
    <p> Hello world! </p>
  </body>
</html>
```

Current version of HTML is 5 and it makes use of the following declaration: <!DOCTYPE html>

doctype

- DOCTYPE tells the consuming user agent (web browsers, web crawlers, validation tools) what type of document the file is. Using it ensures that the consumer correctly parses the HTML as you intended it.
- HTML 4.01 Strict
  - This DTD contains all HTML elements and attributes, but does NOT INCLUDE presentational or deprecated elements (like font).
    - <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01//EN" "http://www.w3.org/TR/html4/strict.dtd">
- HTML 4.01 Transitional
  - This DTD contains all HTML elements and attributes, INCLUDING presentational and deprecated elements (like font).
    - <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN“ "http://www.w3.org/TR/html4/loose.dtd">

doctype

- HTML 4.01 Frameset
  - This DTD is equal to HTML 4.01 Transitional, but allows the use of frameset content.
    - <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/html4/frameset.dtd">
- HTML 5
  - <!DOCTYPE html>

sujata batra

<head> and <body> Elements

- THE **<HEAD>** ELEMENT IS WHERE YOU INCLUDE A **<TITLE>** ELEMENT (THAT APPEARS IN THE TITLE BAR OF THE BROWSER).
- YOU CAN ALSO INCLUDE LOTS OF OTHER TYPE OF INFORMATION IN THE **<HEAD>** ELEMENT.
  - Cascading Style sheet information, or a link to an external style sheet (or several).
  - "Meta" data, such as who authored the page, the type of content, and clues that search engines may (or may not) use to help categorize your page.
  - JavaScript code.
- THE **<BODY>** ELEMENT CONTAINS THE MAIN BULK OF THE MATERIAL TO BE DISPLAYED ON THE WEBPAGE.
  - Paragraphs.
  - Tables and lists.
  - Images.
  - JavaScript code.
  - PHP code can be included here too (if passed through a PHP parser before being served to the client's browser).
  - Other embedded objects (videos, etc).

<head> Elements

Meta tags

The <meta> tag provides metadata about the HTML document.

Meta elements are typically used to specify page description, keywords, author of the document, last modified, and other metadata.

Some examples –

Example 1 - Define keywords for search engines:

```
<meta name="keywords, description " content="HTML, CSS, XML, XHTML, JavaScript">
```

Example 3 - Define the author of a page:

```
<meta name="author" content="Hege Refsnes">
```

Example 4 - Refresh document every 30 seconds:

```
<meta http-equiv="refresh" content="30">
```

<head> Elements (Cont.)

Title Tag

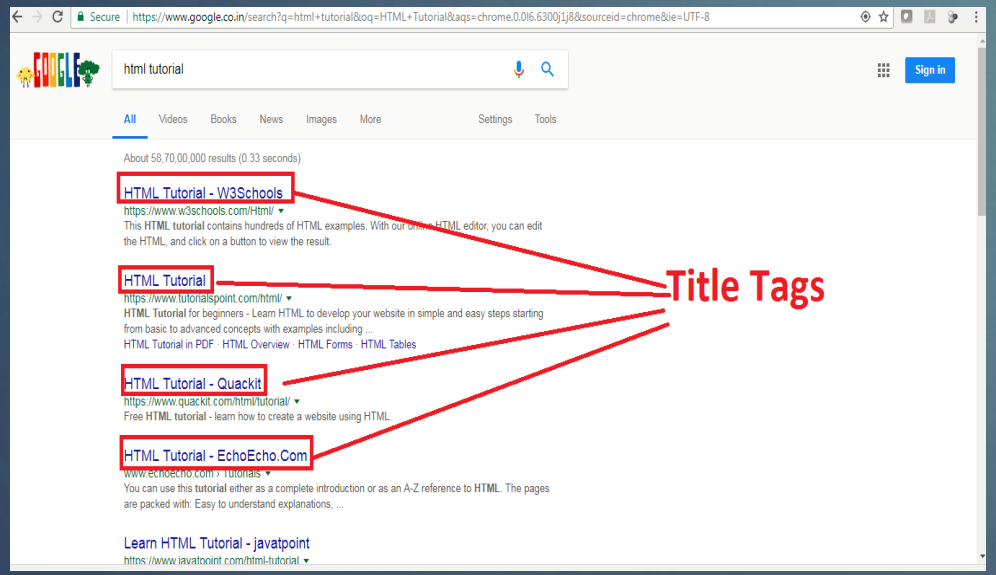
The <title> tag is required in all HTML documents and it defines the title of the document.

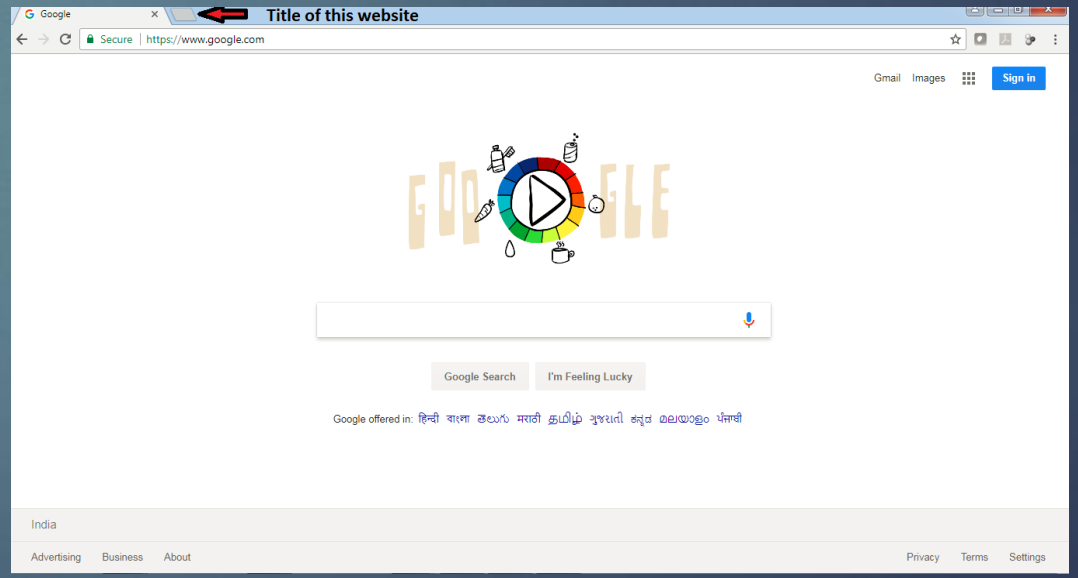
The <title> element: Defines a title in the browser toolbar.

Provides a title for the page when it is added to favorites.

Displays a title for the page in search-engine results.

Eg.:





<head> Elements (Cont.)

Link Tag

The <link> tag defines a link between a document and an external resource.

In HTML the <link> tag has no end tag.

Some Imp. Attributes – charset, - To know browser, which character encoding is used.
href, - hyperlink.
rel, - Relation between linked document.
target. – It specifies where to open the linked document.

Example –

```
<head>
<link rel="stylesheet" type="text/css" href="theme.css">
</head>
```

<head> Elements (Cont.)

Script Tags

The <script> tag is used to define a client-side script, such as a JavaScript.

The <script> element either contains scripting statements, or it points to an external script file through the src attribute.

Example -

```
<html>
<head>
<title>Align Attribute  Example</title>
</head>
<body>

<p id="demo"></p>

<script>document.getElementById("demo").innerHTML
= "Hello JavaScript!"; </script>

</body>
</html>
```

Hello JavaScript!

<head> Elements (Cont.)

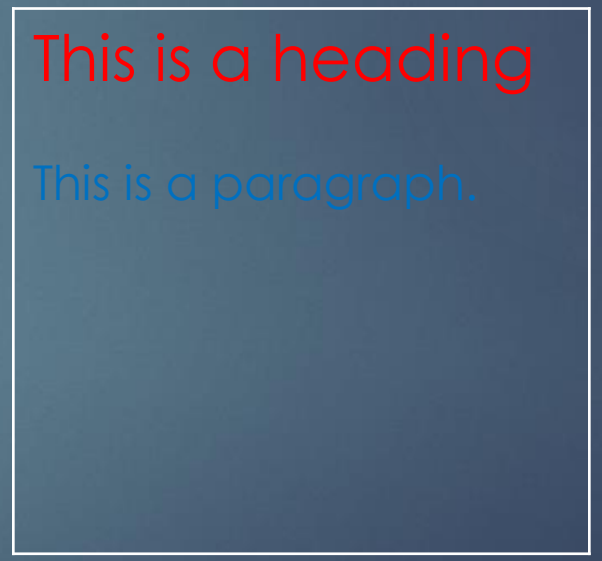
Style Tag

The <style> tag is used to define style information for an HTML document.

Inside the <style> element you specify how HTML elements should render in a browser.

Example-

```
<html>
<head>
h1 {color:red;}
p {color:blue;}
</head>
<body>
<h1>A heading</h1>
<p>A paragraph.</p>
</body>
</html>
```



Elements for the BODY section

Block-level elements

The BODY of a document consists of multiple block elements. If plain text is found inside the body, it is assumed to be inside a paragraph P. See the syntax rules for an explanation of the syntax used in the overview.

Headings
H1 - Level 1 header
H2 - Level 2 header
H3 - Level 3 header
H4 - Level 4 header
H5 - Level 5 header
H6 - Level 6 header

Text containers
P - Paragraph
PRE - Preformatted text
BLOCKQUOTE - Large quotation
ADDRESS - Address information

Text Level Elements
Logical Markups
Physical Markups
Special Markups

Lists
UL - Unordered list
OL - Ordered list
DIR - Directory list
MENU - Menu item list
LI - List item
DL - Definition list
DT - Definition term
DD- Definition

Others
DIV - Logical division
CENTER - Centered division
FORM - Input form
HR - Horizontal rule
TABLE - Tables

Elements for the BODY section

Headings

There are 6 types of heading tags.

Eg.: –

```
<html>
<body>
<h1>This is heading 1</h1>
<h2>This is heading 2</h2>
<h3>This is heading 3</h3>
<h4>This is heading 4</h4>
<h5>This is heading 5</h5>
<h6>This is heading 6</h6>
<p><b>Tip:</b> Use h1 to h6 elements only for headings. Do not use them just to make text bold or big. Use other tags for that.</p>
</body>
</html>
```

This is heading 1

This is heading 2

This is heading 3

This is heading 4

This is heading 5

This is heading 6

Tip: Use h1 to h6 elements only for headings. Do not use them just to make text bold or big. Use other tags for that.

Elements for the BODY section (Cont.)

<p> - Paragraph Tag and <pre> - Preformatted Tag

<p> Tag - Another way to structure your text in paragraph forms.

<Pre> Tag - is used to apply structural exactness.

Eg.:

```
<html>
<body>
<p>This is a paragraph of text.</p>
<p>This is a second paragraph of text.</p>
<pre>This is preformatted text with       exact space,
line and breaks.</pre>
</body>
</html>
```

This is a paragraph of text.

This is a second paragraph of text.

```
This is preformatted text with       exact space,
line and breaks.
```

Elements for the BODY section (Cont.)

<blockquote> Tag and <address> Tag

Blockquote Tag - Indicates that the enclosed text is an extended quotation.

Address Tag - Address Information of the Author/Owner.

Eg.:

```
<html>
<body>
<blockquote cite="http://www.sujatabatra.com/">
  <p>This is a quotation taken from the Sujata Batra.</p>
</blockquote>

<address>
Written by
<a href="mailto:sujatabatra@hotmail.com">Sujata
Batra</a>.<br>
Visit us at:<br>
www.SujataBatra.com<br>
S1, TowerX, Some Business Bay,<br>
Outer Ring Road, Delhi.
</address>
</body>
</html>
```

This is a quotation taken from the Sujata Batra.

Written by Sujata Batra.
Visit us at:
www.SujataBatra.com
S1, TowerX, Some Business Bay,
Outer ring road, Delhi.

Elements for the BODY section (Cont.)

Links and Navigation

Anchor Element-

An anchor can be used to create a link to another document (with the href attribute).

Types –

External : Our Best Friend

Internal : contact

Image Tag-

The syntax for the tag to insert image into the webpage is-

<img src="url" alt="alternate text">

Eg.: <img src="\images\SomeImage.jpg" alt="Some Image">

Elements for the BODY section (Cont.)

Unordered List and Odered Lists

Unordered Lists - tag. Item lists in <li> tag. The list items will be marked with bullets.

Ordered Lists - tag. Item lists in <li> tag. . The list items will be marked with numbers.

Eg.:

```
<html>
<body>
<h2>Unordered List </h2>
<ul>
  <li>Java</li>
  <li>Python</li>
  <li>Ruby</li>
</ul>
<h2>Ordered List </h2>
<ol>
  <li>Java</li>
  <li>Python</li>
  <li>Ruby</li>
</ol>
</body>
</html>
```

Unordered List

- Java
- Python
- Ruby

Ordered List

1. Java
2. Python
3. Ruby

Elements for the BODY section (Cont.)

Div Tag

** tag –** Used to defines a division or a section in an HTML document. And to group block-elements to format them with CSS.

Eg.:

```
<html>
<body>
<div style="color:#00FF00">
  <h2>Sujata Academy</h2>
  <p>Welcome to Java Training</p>
</div>
</body>
</html>
```

Sujata Academy

Welcome to Java Training.

Elements for the BODY section (Cont.)

Table Element

<table> Tag : **<tr>** **T**able **R**ow - Defines a new row,
<td> **T**able **D**ata - Defines a single cell,
<th> **T**able **H**eadings - Defines header cell.

```
<html>
<body>
<style> table, th, td { border: 1px solid black; }
</style>
<table>
 <tr>
  <th>Day</th> <th>Session</th>
 </tr>
 <tr>
  <td>Thursday</td> <td>HTML</td>
 </tr>
 <tr>
  <td>Friday</td> <td>CSS</td>
 </tr>
</table>
</body>
</html>
```

Day	Session
Thursday	HTML
Friday	CSS

Elements removed in HTML5

Element	Use instead
<acronym>	<abbr>
<applet>	<object>
<basefont>	CSS
<big>	CSS
	CSS
<dir>	
<font>	CSS
<frame>	
<frameset>	
<noframes>	
<strike>	CSS
<tt>	CSS

Sujata Batra

HTML5 New Tags and Elements

HTML5 Indroduces 28 New Elements, Some of them are mentioned here.

Navigation:

<article>
<aside>
<header>
<hgroup>
<footer>
<figure>
<figcaption>
<nav>
<section>

Multimedia/Interactivity:

<audio>
<canvas>
<embed>
<source>
<track>
<video>

New <input> types:

color
date
datetime
datetime-local
email
month
number
range
search
tel
time
url
week

Miscellaneous:

<bdi>
<command>
<datalist>
<details>
<mark>
<meter>
<output>
<progress>
<summary>
<rp>
<rt>
<ruby>
<time>
<wbr>

Defining HTML5 Documents

Sujata Batra

Remember the DOCTYPE declaration-

```
<!DOCTYPE html>
```

Again, HTML5 simplifies this line:

```
<html lang="en">
```

The default character encoding (charset) declaration

```
<meta charset="UTF-8">
```

Semantic Elements

A semantic element clearly describes its meaning to both the browser and the developer.

Eg. of non-semantic elements: ** ** and **** - Tells nothing about its content.

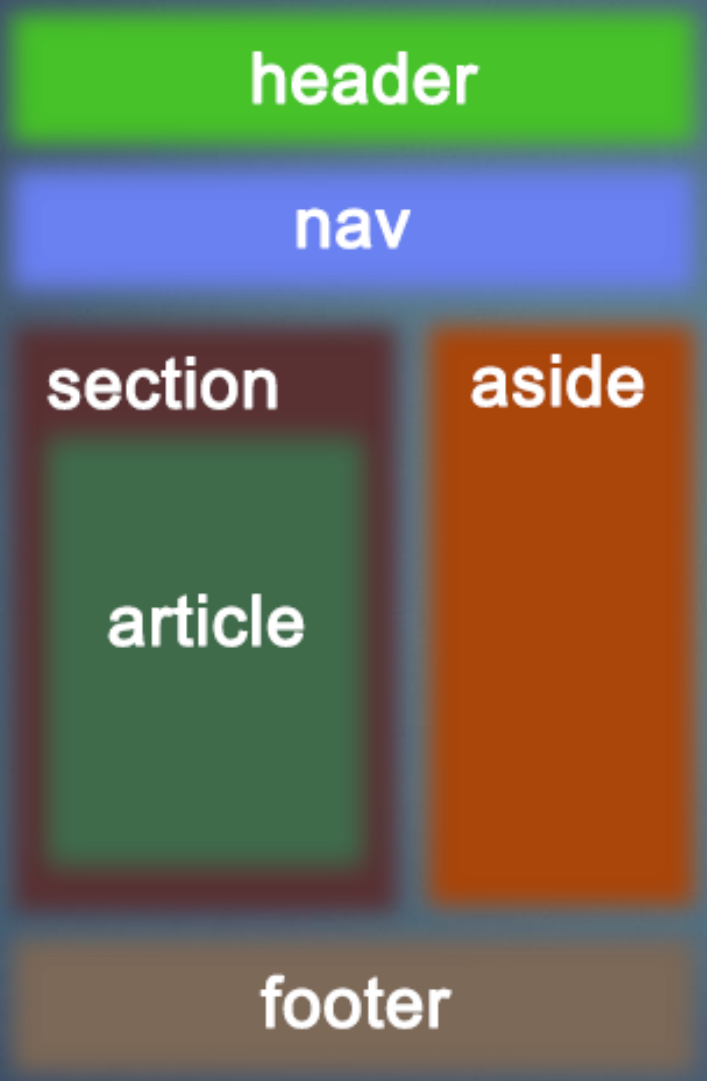
Eg. of semantic elements: **<form>, <table>,** and **<img>** - Clearly defines its content.

Many web sites contain HTML code like: ** ** to indicate navigation, header, and footer.

HTML5 offers new semantic elements to define different parts of a web page:

<article>
<aside>
<details>
<figcaption>
<figure>
<footer>
<header>
<main>
<mark>
<nav>
<section>
<summary>
<time>

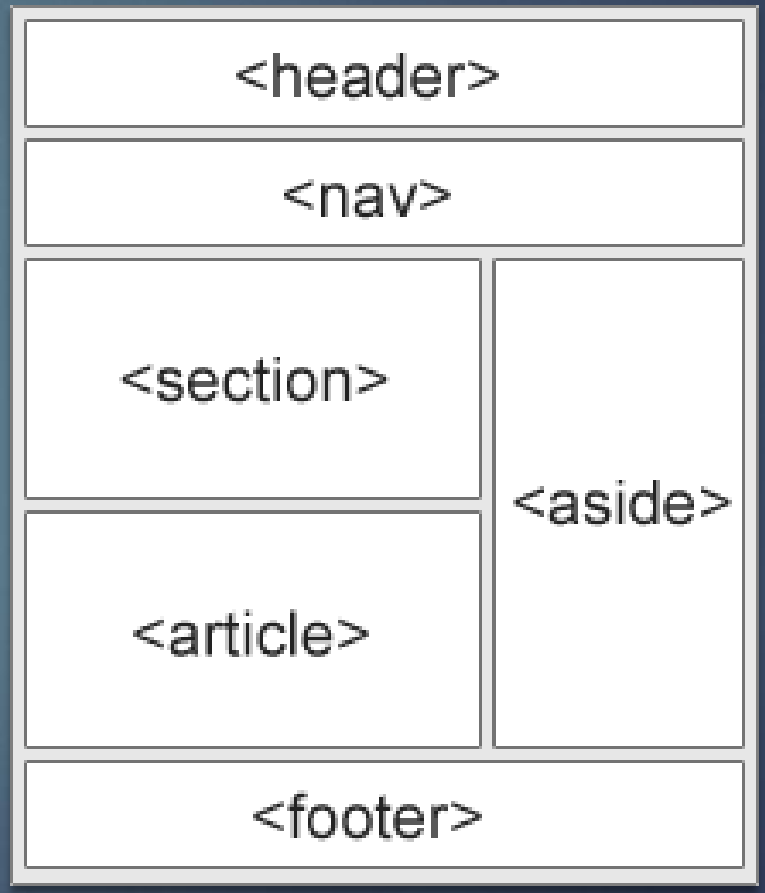
Semantic Elements




```
<header>
    <h1>title</h1>
    <form>Search</form>
    <nav>
        <ul>Site navigation</ul>
    </nav>
</header>
<section>
    <article>
        <h1>Article title</h1>
        <p>Summary</p>
    </article>
</section>
<aside>
    <section>
        <h2>Blogroll…</h2>
    </section>
</aside>
<footer>
    <h2>Footer</h2>
</footer>
```

New Semantic Elements in HTML5

Tag	Description
<article>	Defines an article
<aside>	Defines content aside from the page content
<details>	Defines additional details that the user can view or hide
<figcaption>	Defines a caption for a <figure> element
<figure>	Specifies self-contained content, like illustrations, diagrams, photos, code listings, etc.
<footer>	Defines a footer for a document or section
<header>	Specifies a header for a document or section
<main>	Specifies the main content of a document
<mark>	Defines marked/highlighted text
<nav>	Defines navigation links
<section>	Defines a section in a document
<summary>	Defines a visible heading for a <details> element
<time>	Defines a date/time



Sujata Batra

HTML5 <section> Element

- A section is a thematic grouping of content, typically with a heading.
- A home page could normally be split into sections for introduction, content, and contact information.

HTML5 <article> Element

- The <article> element specifies independent, self-contained content.
- An article should make sense on its own, and it should be possible to read it independently from the rest of the web site.
- Examples of where an <article> element can be used:
  - Forum post
  - Blog post
  - Newspaper article

HTML5 <header> Element

- The <header> element specifies a header for a document or section.
- The <header> element should be used as a container for introductory content.
- You can have several <header> elements in one document.

HTML5 <footer> Element

- The <footer> element specifies a footer for a document or section.
- A <footer> element should contain information about its containing element.
- A footer typically contains the author of the document, copyright information, links to terms of use, contact information, etc.
- You may have several <footer> elements in one document.

HTML5 <nav> Element

- The <nav> element defines a set of navigation links.
- NOT all links of a document should be inside a <nav> element. The <nav> element is intended only for major block of navigation links.

HTML5 <aside> Element

- The <aside> element defines some content aside from the content it is placed in (like a sidebar).

- The <aside> content should be related to the surrounding content.

HTML5 <figure> and <figcaption> Elements

- The purpose of a figure caption is to add a visual explanation to an image.
- In HTML5, an image and a caption can be grouped together in a <figure> element:

- **Example:**

```
<figure>
  <img src="pic_trulli.jpg" alt="Trulli">
  <figcaption>Fig1. - Trulli, Puglia, Italy.</figcaption>
</figure>
```

HTML <summary> Tag

- The <summary> tag defines a visible heading for the <details> element. The heading can be clicked to view/hide the details.
- Browser Support

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<summary>	12.0	Not Supported	49.0	6.0	15.0

HTML <details> Tag

- The <details> tag can be used to create an interactive widget that the user can open and close.
- The content of a <details> element should not be visible unless the open attribute is set.
- Browser Support

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<details>	12.0	Not Supported	49.0	6.0	15.0

Example


```
<details>
  <summary>Copyright 1999-2018.</summary>
  <p> - by Refsnes Data. All Rights Reserved.</p>
  <p>All content and graphics on this web site are the property of the
company Refsnes Data.</p>
</details>
```

HTML <main> Tag

- The <main> tag specifies the main content of a document.
- The content inside the <main> element should be unique to the document. It should not contain any content that is repeated across documents such as sidebars, navigation links, copyright information, site logos, and search forms.
- Note: There must not be more than one <main> element in a document. The <main> element must NOT be a descendant of an <article>, <aside>, <footer>, <header>, or <nav> element.

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<main>	6.0	12.0	4.0	5.0	11.1

HTML <mark> Tag

- Use the <mark> tag if you want to highlight parts of your text.

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<mark>	6.0	9.0	4.0	5.0	11.1

HTML <time> Tag

- The <time> tag defines a human-readable date/time.
- This element can also be used to encode dates and times in a machine-readable way so that user agents can offer to add birthday reminders or scheduled events to the user's calendar, and search engines can produce smarter search results.

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<time>	6.0	9.0	4.0	5.0	11.1

- Attribute

Attribute	Value	Description
datetime	*datetime*	Represent a machine-readable date/time of the <time> element

Migration from HTML4 to HTML5

HTML4	HTML5
	<header>
	<nav>
	<section>
	<article>
	<footer>

HTML5 Form Elements

- HTML5 added the following form elements:

- <datalist>
- <output>

HTML5 <datalist> Element

- <datalist> element specifies a list of pre-defined options for an <input> element.
- Users will see a drop-down list of the pre-defined options as they input data.
- The list attribute of the <input> element, must refer to the id attribute of the <datalist> element.
- <form action="/action_page">
  <input **list="browsers"**>
  <datalist **id="browsers"**>
  <option value="Internet Explorer">
  <option value="Firefox">
  <option value="Chrome">
  <option value="Opera">
  <option value="Safari">
  </datalist>
  </form>

HTML5 <output> Element

- <output> element represents the result of a calculation (like one performed by a script).

```
function showResult() {
x = document.forms["myform"]["newinput"].value;
document.forms["myform"]["result"].value = x;
  }
<form action = "/cgi-bin/html5.cgi" method = "get" name = "myform">
 Enter a value : <input type = "text" name = "newinput" />
<input type = "button" value = "Result"  onclick = "showResult();" />
<output name = "result"></output>
</form>
```

Elements for the BODY section (Cont.)

Form Elements

<form> - It is a method of accepting inputs from user. A form is an area that can contain form elements.

Eg.:

```
<form name="form1" action="abc.asp" method=get>
<!- form elements -->
</forms>
```

Name- is used for future manipulation of data by scripting language.

Action- indicates a program on the server that will be executed when this form is submitted. Mostly it will be an ASP or a CGI script.

Method- indicates the way the form is submitted to the server - popular options are GET/POST.

Elements for the BODY section (Cont.)

Form Elements

Form Elements	Description
Text Field	Can create a Text Field by using Input Element with Type Attribute.
Pass Word Field	When text is entered in Pass Word Field it shows **** Symbol
Combo Box	It can have multiple values and it allows user to select one value at a time
List Box	It can have multiple values and allows user to select more than one value at a time
Radio Button	Can create a Radio Button by using Input Element with Value and Name Attribute
Check Box	Can create Check box by Using Input Element
Command Button	This is useful for submitting any data that is helpful in transferring data across different interfaces

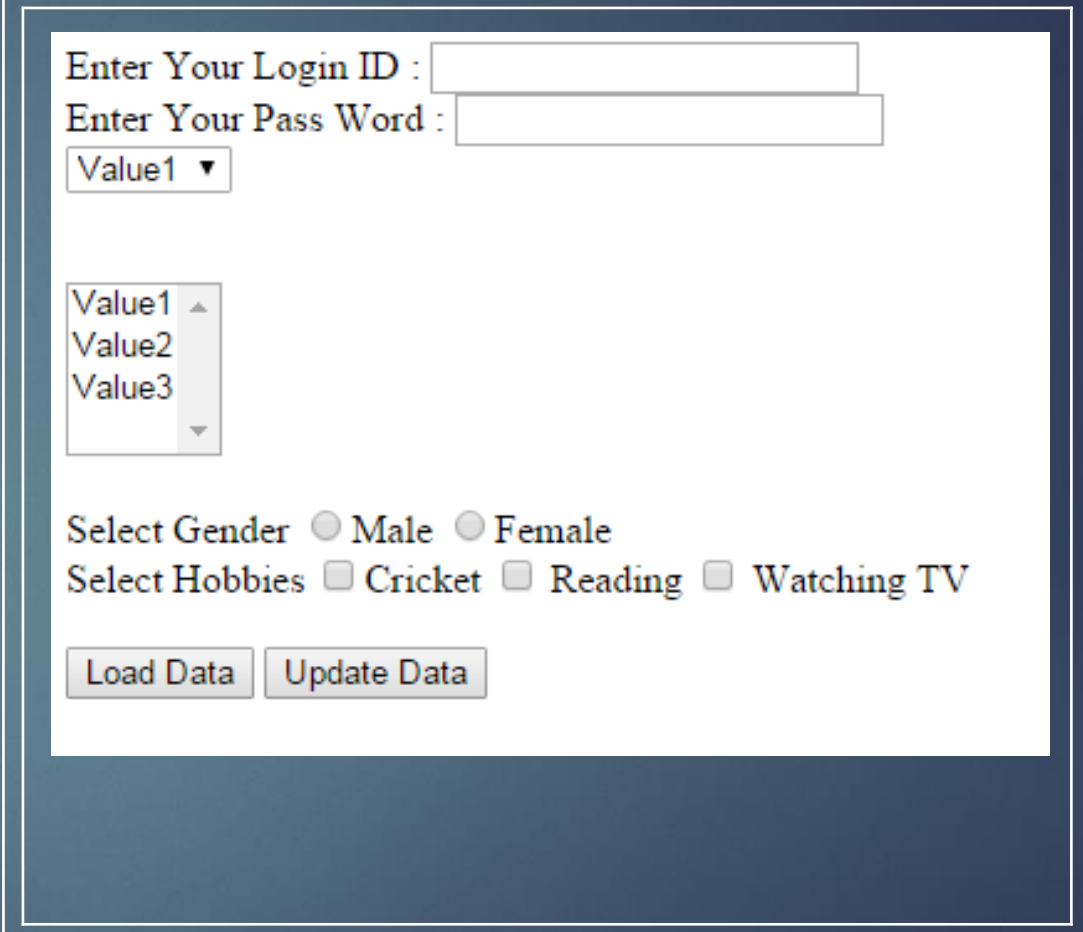
Elements for the BODY section (Cont.)

Form Elements

```
<html>
<body>
<form name="frm">
        Enter Your Login ID   : <input type=textsize=20><br>
        Enter Your Pass Word :  <input type=Password maxlength=8
size=20><br>
        <select name=combo1>
                <option>Value1</option>
                <option>Value2</option>
                <option>Value3</option>
        </select> <br><br><br>
        <select name=combo1 multiple>
                 <option>Value1</option>
                 <option>Value2</option>
                 <option>Value3</option>
         </select><br><br>
         Select Gender
        <input type=Radio value=Male name=Checked>Male
      <input type=Radio value=Female name=Checked>Female <br>

         Select Hobbies
         <input type= Checkbox >Cricket
         <input type= Checkbox > Reading
         <input type= Checkbox > Watching TV<br><br>

         <input type=submit value="Load Data">
         <input type=button value="Update Data">
</form>
</body>
</html>
```



Elements for the BODY section (Cont.)

Sujata Batra

Character Entities

Some characters like the < character, have a special meaning in HTML, and therefore cannot be used in the text. The most common character entities:

Result	Description	Entity Name
	non-breaking space	
<	less than	&lt;
>	greater than	&gt;
&	ampersand	&amp;
“	quotation mark	&quot;
‘	apostrophe	&apos;

Some Other Commonly Used Character Entities

©	copyright	&copy;
®	registered trademark	&reg;
£	pound	&pound;
¥	yen	&yen;

HTML 5 Input Types

HTML5 Input Types

- HTML5 added several new input types:
- color
- date
- datetime-local
- email
- month
- number
- range
- search
- tel
- time
- url
- week

Note : New input types that are not supported by older web browsers, will behave as <input type="text">.

- **Input Type Color**
  - The <input type="color"> is used for input fields that should contain a color.
  - Depending on browser support, a color picker can show up in the input field.

Example : <input type="color" name="favcolor" value="#ff0000">

- **Input Type Date**
  - The <input type="date"> is used for input fields that should contain a date.
  - You can also use the min and max attributes to add restrictions to dates.
  - Depending on browser support, a date picker can show up in the input field.

Example : Enter a date before 1980-01-01:
<input type="date" name="bday" max="1979-12-31">
Enter a date after 2000-01-01:
<input type="date" name="bday" min="2000-01-02">

- **Input Type Datetime-local**
  - The <input type="datetime-local"> specifies a date and time input field, with no time zone.
  - Depending on browser support, a date picker can show up in the input field.

Example : <input type="datetime-local" name="bdaytime">

- **Input Type Email**
  - The <input type="email"> is used for input fields that should contain an e-mail address.
  - Depending on browser support, the e-mail address can be automatically validated when submitted.

- **Input Type File**
  - The <input type="file"> defines a file-select field and a "Browse" button for file uploads.

- **Input Type Month**
  - The <input type="month"> allows the user to select a month and year.
  - Depending on browser support, a date picker can show up in the input field.

- **Input Type Number**
  - The <input type="number"> defines a numeric input field.
  - You can also set restrictions on what numbers are accepted.

Example: <input type="number" name="quantity" min="1" max="5">

HTML 5 Input Attributes

HTML5 added the following attributes for <input>:

- autocomplete
- autofocus
- form
- formaction
- formenctype
- formmethod
- formnovalidate
- formtarget
- height and width
- list
- min and max
- multiple
- pattern (regexp)
- placeholder
- required
- step

and the following attributes for <form>:

- autocomplete
- novalidate

- **The autocomplete Attribute**
  - The autocomplete attribute specifies whether a form or input field should have autocomplete on or off.
  - When autocomplete is on, the browser automatically completes the input values based on values that the user has entered before.
  - It is possible to have autocomplete "on" for the form, and "off" for specific input fields, or vice versa.
  - The autocomplete attribute works with <form> and the following <input> types: text, search, url, tel, email, password, datepickers, range, and color.

- **The novalidate Attribute**
  - The novalidate attribute is a <form> attribute.
  - When present, novalidate specifies that the form data should not be validated when submitted.
- Example : <form action="/action_page" novalidate>
  E-mail: <input type="email" name="user_email">
  <input type="submit">
  </form>

- **The autofocus Attribute**
  - The autofocus attribute specifies that the input field should automatically get focus when the page loads.

- **The form Attribute**
  - The form attribute specifies one or more forms an <input> element belongs to.
  - To refer to more than one form, use a space-separated list of form ids.
- Example :<form action="/action_page.php" id="form1">
  First name: <input type="text" name="fname">
  <input type="submit" value="Submit">
  </form>

  Last name: <input type="text" name="lname" form="form1">

- **The formaction Attribute**
  - The formaction attribute specifies the URL of a file that will process the input control when the form is submitted.
  - The formaction attribute overrides the action attribute of the <form> element.
  - The formaction attribute is used with type="submit" and type="image".

Example :<form action="/action_page.php">
First name: <input type="text" name="fname">
Last name: <input type="text" name="lname">
<input type="submit" value="Submit">
<input type="submit" formaction="/action_page2.php"
value="Submit as admin">
</form>

- **The formmethod Attribute**
  - The formmethod attribute defines the HTTP method for sending form-data to the action URL.
  - The formmethod attribute overrides the method attribute of the <form> element.
  - The formmethod attribute can be used with type="submit" and type="image".

- **The formnovalidate Attribute**
  - The formnovalidate attribute overrides the novalidate attribute of the <form> element.
  - The formnovalidate attribute can be used with type="submit".
- Example :
```
<form action="/action_page.php">
  E-mail: <input type="email" name="userid"><br>
  <input type="submit" value="Submit"><br>
  <input type="submit" formnovalidate value="Submit without validation">
</form>
```

- **The height and width Attributes**
  - The height and width attributes specify the height and width of an <input type="image"> element.
  - Always specify the size of images. If the browser does not know the size, the page will flicker while images load.

- **The min and max Attributes**
  - The min and max attributes specify the minimum and maximum values for an <input> element.
  - The min and max attributes work with the following input types: number, range, date, datetime-local, month, time and week.

- **The multiple Attribute**
  - The multiple attribute specifies that the user is allowed to enter more than one value in the <input> element.
  - The multiple attribute works with the following input types: email, and file.

- **The pattern Attribute**
  - The pattern attribute specifies a regular expression that the <input> element's value is checked against.
  - The pattern attribute works with the following input types: text, search, url, tel, email, and password.
  - Use the global title attribute to describe the pattern to help the user.
- Example : Country code: <input type="text" name="country_code" pattern="[A-Za-z]{3}" title="Three letter country code">

- **The placeholder Attribute**
  - The placeholder attribute specifies a hint that describes the expected value of an input field (a sample value or a short description of the format).
  - The hint is displayed in the input field before the user enters a value.
  - The placeholder attribute works with the following input types: text, search, url, tel, email, and password.

- **The required Attribute**
  - The required attribute specifies that an input field must be filled out before submitting the form.
  - The required attribute works with the following input types: text, search, url, tel, email, password, date pickers, number, checkbox, radio, and file.

- **The step Attribute**
  - The step attribute specifies the legal number intervals for an <input> element.

Example: if step="3", legal numbers could be -3, 0, 3, 6, etc.

  - The step attribute can be used together with the max and min attributes to create a range of legal values.
  - The step attribute works with the following input types: number, range, date, datetime-local, month, time and week.

HTML 5 Media

HTML Audio Tag

- HTML audio tag is used to define sounds such as music and other audio clips.
- Currently there are three supported file format for HTML 5 audio tag.
  - mp3
  - wav
  - ogg

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<audio>	4.0	9.0	3.5	4.0	10.5

Browser Support Audio File Format

Browser	mp3	wav	ogg
Internet Explorer	yes	no	no
Google Chrome	yes	yes	yes
Mozilla Firefox	yes	yes	yes
Opera	no	yes	yes
Apple Safari	yes	yes	no

Attributes of HTML Audio Tag

Attribute	Description
controls	It defines the audio controls which is displayed with play/pause buttons.
autoplay	It specifies that the audio will start playing as soon as it is ready.
loop	It specifies that the audio file will start over again, every time when it is completed.
muted	It is used to mute the audio output.
preload	It specifies the author view to upload audio file when the page loads.
src	It specifies the source URL of the audio file.

Audio Tag Example


```
<audio controls>
   <source src="myaudio.mp3" type="audio/mpeg">
Your browser does not support the audio element.
</audio>
```

HTML Video Tag

- The HTML video tag is used for streaming video files such as a movie clip, song clip on the web page.
- Three video formats supported for HTML video tag:
  - mp4
  - webM
  - ogg

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<video>	4.0	9.0	3.5	4.0	10.5

Browser Support Video File Format

Browser	mp4	webM	ogg
Internet Explorer	yes	no	no
Google Chrome	yes	yes	yes
Mozilla Firefox	yes	yes	yes
Opera	no	yes	yes
Apple Safari	yes	no	no

Attributes of HTML Video Tag

Attribute	Value	Description
autoplay	autoplay	Specifies that the video will start playing as soon as it is ready
controls	controls	Specifies that video controls should be displayed (such as a play/pause button etc).
height	pixels	Sets the height of the video player
loop	loop	Specifies that the video will start over again, every time it is finished
muted	muted	Specifies that the audio output of the video should be muted
poster	URL	Specifies an image to be shown while the video is downloading, or until the user hits the play button
preload	auto metadata none	Specifies if and how the author thinks the video should be loaded when the page loads
src	URL	Specifies the URL of the video file
width	pixels	Sets the width of the video player

Video Tag Example


```
<video width="320" height="240" controls autoplay loop>
  <source src="movie.mp4" type="video/mp4">
  Your browser does not support the html video tag.
</video>
```

Sujata Batra

Graphics API
(Canvas and SVG)

Previously possible with Flash, VML, Silverlight.

Very complex to do in JavaScript without plugins (for example, rounded corners or diagonal lines).

Provide native drawing functionality on the Web.

Completely integrated into HTML5 documents (Part of DOM).

Can be styled with CSS.

Can be controlled with JavaScript.

HTML Canvas

- The HTML <canvas> element is used to draw graphics, on the fly, via JavaScript.
- The <canvas> element is only a container for graphics. You must use JavaScript to actually draw the graphics.
- Canvas has several methods for drawing paths, boxes, circles, text, and adding images.

Element	Chrome	Internet Explorer	Mozilla Firefox	Apple Safari	Opera
<canvas>	4.0	9.0	2.0	3.1	9.0

Canvas Examples

- A canvas is a rectangular area on an HTML page. By default, a canvas has no border and no content.

Note: Always specify an id attribute (to be referred to in a script), and a width and height attribute to define the size of the canvas. To add a border, use the style attribute.

Example of an empty Canvas

```
<canvas id="myCanvas" width="200" height="100" style="border:1px solid
#000000;">
</canvas>
```

Draw on the Canvas With JavaScript

- Step 1: Find the Canvas Element

```
var canvas = document.getElementById("myCanvas");
```

- Step 2: Create a Drawing Object
  - The getContext() is a built-in HTML object, with properties and methods for drawing

```
var ctx = canvas.getContext("2d");
```

- Step 3: Draw on the Canvas
  - Finally, you can draw on the canvas.
  - Set the fill style of the drawing object to the color red:

```
ctx.fillStyle = "#FF0000";
```

  - The fillStyle property can be a CSS color, a gradient, or a pattern. The default fillStyle is black.
  - The fillRect(*x,y,width,height*) method draws a rectangle, filled with the fill style, on the canvas:

```
ctx.fillRect(0, 0, 150, 75);
```

Canvas Coordinates

- The HTML canvas is a two-dimensional grid.
- The upper-left corner of the canvas has the coordinates (0,0)

- Draw a Line
  - Use the following methods.
  - moveTo(x,y): It is used to define the starting point of the line.
  - lineTo(x,y): It is used to define the ending point of the line.
  - If you draw a line which starting point is (0,0) and the end point is (200,100), use the stroke method to draw the line.

Example

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.moveTo(0, 0);
ctx.lineTo(200, 100);
ctx.stroke();
```

- Drawing Circle on Canvas
  - beginPath() - begins a path
  - use the arc() method
  - arc(x, y, r, startAngle, stopAngle)

```
var c = document.getElementById("myCanvas");
var ctx = c.getContext("2d");
ctx.beginPath();
ctx.arc(95, 50, 40, 0, 2 * Math.PI);
ctx.stroke();
```

Canvas - Gradients

- Gradients can be used to fill rectangles, circles, lines, text, etc. Shapes on the canvas are not limited to solid colors.
- There are two different types of gradients:
  - createLinearGradient(*x,y,x1,y1*) - creates a linear gradient
  - createRadialGradient(*x,y,r,x1,y1,r1*) - creates a radial/circular gradient
    - creates a radial gradient using the size and coordinates of two circles.
- Once we have a gradient object, we must add two or more color stops.
- The addColorStop() method specifies the color stops, and its position along the gradient. Gradient positions can be anywhere between 0 to 1.

- To use the gradient, set the fillStyle or strokeStyle property to the gradient, then draw the shape (rectangle, text, or a line).

```
var canvas = document.getElementById('canvas');
var ctx = canvas.getContext('2d');

// Create a linear gradient
// The start gradient point is at x=20, y=0
// The end gradient point is at x=220, y=0
var gradient = ctx.createLinearGradient(20,0, 220,0);

// Add three color stops
gradient.addColorStop(0, 'green');
gradient.addColorStop(.5, 'cyan');
gradient.addColorStop(1, 'green');

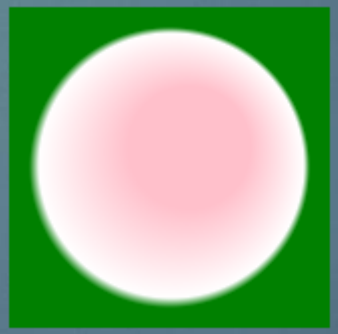
// Set the fill style and draw a rectangle
ctx.fillStyle = gradient;
ctx.fillRect(20, 20, 200, 100);
```

```
var canvas = document.getElementById('canvas');
var ctx = canvas.getContext('2d');

// Create a radial gradient
// The inner circle is at x=110, y=90, with radius=30
// The outer circle is at x=100, y=100, with radius=70
var gradient = ctx.createRadialGradient(110,90,30, 100,100,70);

// Add three color stops
gradient.addColorStop(0, 'pink');
gradient.addColorStop(.9, 'white');
gradient.addColorStop(1, 'green');

// Set the fill style and draw a rectangle
ctx.fillStyle = gradient;
ctx.fillRect(20, 20, 160, 160);
```



Drawing Text on the Canvas

- font - defines the font properties for the text
- fillText(*text,x,y*) - draws "filled" text on the canvas
- strokeText(*text,x,y*) - draws text on the canvas (no fill)

Example

```
var canvas = document.getElementById("myCanvas");
var ctx = canvas.getContext("2d");
ctx.font = "30px Arial";
ctx.fillText("Hello World", 10, 50);
```

Or

```
ctx.strokeText("Hello World", 10, 50);
```

Add Color and Center Text

Sujata Batra

```
var canvas = document.getElementById("myCanvas");
var ctx = canvas.getContext("2d");
ctx.font = "30px Comic Sans MS";
ctx.fillStyle = "red";
ctx.textAlign = "center";
ctx.fillText("Hello World", canvas.width/2, canvas.height/2);
```

Canvas - Images

- To draw an image on a canvas, use the following method:
  - drawImage(*image,x,y*)

```
window.onload = function() {
  var canvas = document.getElementById("myCanvas");
  var ctx = canvas.getContext("2d");
  var img = document.getElementById(“myimage");
  ctx.drawImage(img, 10, 10);
};
```

Sujata Batra

CSS

What is CSS ?

CSS stands for "Cascading Style Sheets"

Cascading: refers to the procedure that determines which style will apply to a certain section, if you have more than one style rule.

Style: how you want a certain part of your page to look. You can set things like color, margins, font, etc for things like tables, paragraphs, and headings.

Sheets: the "sheets" are like templates, or a set of rules, for determining how the webpage will look.

CSS is a stylesheet language used to describe the presentation of a document written in HTML or XML.

CSS

History

- CSS1 was the first edition introduced in 1996.
- CSS2 was published in 1998 and provides enhancement over CSS1.
- CSS2.1 was the last 2nd generation edition of CSS.
- CSS 3 is the latest edition. Several new functionalities have been provided through CSS3.

  Functions like rounded corners, backgroubnd decoration, box shadows, which are demonstrated in the subsequesnt sections, are introduced in this version.

CSS

Advantages

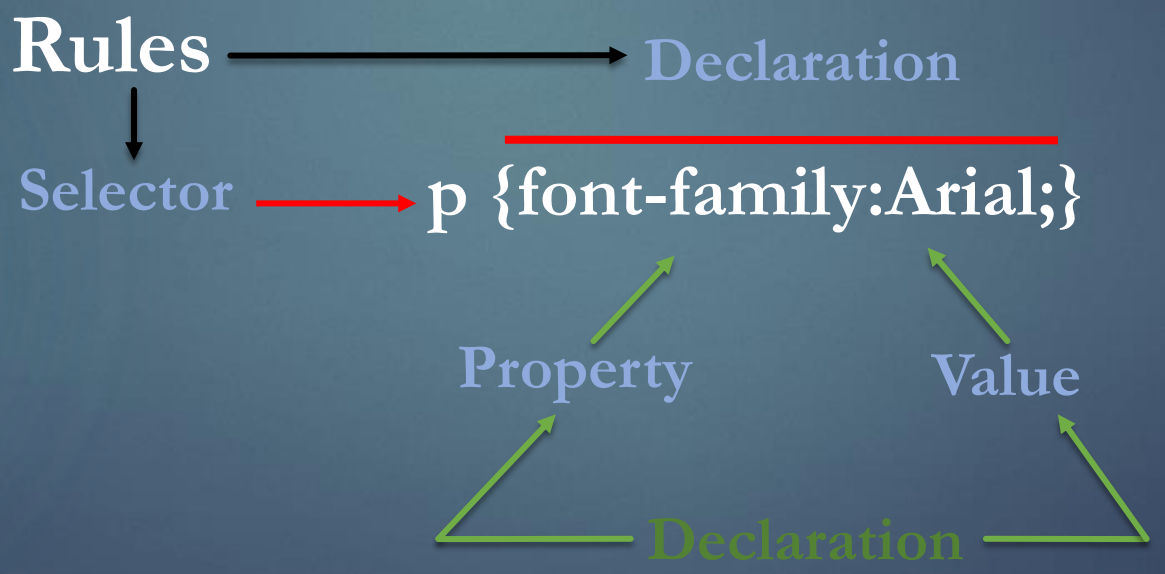
- A web application will contains hundreds of web pages, which are created using HTML.
- Formatting these HTML pages will be a laborious process, as formatting elements need to be applied to each and every page.
- CSS saves lots of work as we can change the appearance and layout of all the web pages by editing just one single CSS file.

CSS Syntax Rules

Rule have two parts - Selector and declaration.

Selector: The HTML element you want to add style to.
<p> <h1> <table> etc

Declaration: The statement of style for that element. Made up of property and value.



CSS Style Example

Selector - I want the text color of my paragraph to be red and the background color to be black.

```
<html>
<head>
<style> p {font-family:Arial; color: red;
background-color:black;} </style>
</head>
<body>
<p> <b> Welcome to Sujata Training </b> </p>
</body>
</html>
```

Welcome to Sujata Training

Selectors

Sujata Batra

CSS Selectors

- Selectors are at the heart of CSS.
- CSS allowed the matching of elements by type, class, and/or id.
- CSS2.1 added pseudo-elements, pseudo-classes, and combinators.
- With CSS3, we can target almost any element on the page with a wide range of selectors.

Universal Selector

```
* {
 margin: 0;
 padding: 0;
}
```

- The star symbol will target every single element on the page.
- Used to zero out the margins and padding.
- It adds too much *weight* on the browser

Select by Id

```
#container {
    width: 960px;
    margin: auto;
}
```

- Prefixing the hash symbol to a selector allows us to target by id.
- Most common usage
- id selectors are rigid and don't allow for reuse.
- Can use a pseudo-class.

Class Selector

```
.error {
  color: red;
}
```

- This is a class selector.
- The difference between ids and classes is that, with the latter, you can target multiple elements.
- Use classes when you want your styling to apply to a group of elements.

Selector Grouping

- Can be grouped using a comma (,) separator.
- Any element that matches either of the selectors in the group:
- **td, th { : declarations }**
  - Similar to the logical OR operator,
  - but it's important to remember that each selector in a group is autonomous.
- **#foo td, th { : declarations }**
- #foo td { : declarations } th { : declarations }

Attribute Selector

- **a[title] {**
- **color: green;**
- **}**

- Select the anchor tags that have a title attribute.

- **a[href="Welcome.html"] {**
- **color: green;**
- **}**

- ***** appear *somewhere* in the attribute's value.
- **^** for Beginning of the String
- **$** is used for end of the String

Sujata Batra

Attribute selectors

Attribute selectors selects elements based upon the attributes present in the HTML Tags and their value.

```
IMG[src="small.gif"] {
  border: 1px solid #000;
 }
```

will work for

```
<img src="small.gif" />
```

Descendent Selector

```
Ul  a {
  text-decoration: none;
}
```

- The next most comment selector is the descendant selector.
- Also known as Compound selectors
- When you need to be more specific with your selectors, you use these.
- Example : To target the anchors which are within an unordered list

Adjacent Selector

```
ul + p {
    color: red;
}
```

- This is referred to as an adjacent selector.
- It will select *only* the element that is immediately preceded by the former element.
- In this case, only the first paragraph after each ul will have red text.

Child selectors

A child selector is used to select an element that is a direct child of another element (parent). Child selectors will not select all descendants, only direct children.

HTML

```
<div >
  <div class="abc">
    <P>
   Hello there!
  </div>
</div>
```

CSS

```
DIV.abc > P {
        font-weight:bold;
}
```

```
</p>
```

General sibling selector

- Selector is a tilde character (~).
- Matches elements that are siblings of a given element.
- To match a p element if it's a sibling of an h2 element:
- **h2~p { : declarations }**
- <h2>Heading</h2>
- The selector above matches this paragraph.
- The selector above matches this paragraph.
- Here, both paragraphs match the sibling selector h2~p

Sujata Batra

Inserting a StyleSheet

You can do in three different ways-

1. **External Style Sheet**

   Styles are specified in an external CSS file. you can change the looks of entire website by using single external style sheet.

   Eg.: <head> <link rel="stylesheet" type="text/css" href="ex1.css" /> </head>

2. **Internal Style Sheet**

   To Appy specific styles to a single HTML file inside the head section of an HTML page.

   Eg.: <style> p { **text-align**:left; **font-size**:24px; } </style>

3. **Inline Styles**

   Styles are specified inside an HTML tag/element.

   Eg.: Demo of Inline Style

Formatting with CSS Properties

CSS Background

We can use CSS Background properties to define the background effects of an element.

The following properties can be used for background effects :

- background-color
  - The background-color property is used to specify the background color of the element.
- background-image
  - The background-image property is used to set an image as a background of an element.
- background-repeat
- background-attachment
  - used to specify if the background image is fixed or scroll with the rest of the page in browser window.
- background-position
  - used to define the initial position of the background image.
  - By default, the background image is placed on the top-left of the webpage.

Formatting with CSS Properties

CSS Background Image

You can use an image as the background for an element using background-image property.

Example-

```
body{
        background-image:url('java.png');
        }
```

By default, the image is repeated, both horizontally and vertically, so as to cover the entire body (or the element on which it is applied).

Formatting with CSS Properties

CSS Background Color

The **background-color** property is used to specify the background color of an element.

Example-

```
body {
        background-color:darkblue;
        }
```

Similarly, we can specify the background for any element (wherever applicable).

```
p {
   background-color:orange;
   }
```

Formatting with CSS Properties

CSS Background Position

If the background image disturbs the text, i.e. if the text cannot be read clearly due to the image in the background, we can set the position of the background image.

Example-

```
body {
        background-image:url("SomeImage.jpg");
        background-repeat:no-repeat;
        background-position:right top;
        }
```

CSS3 Gradient

- Display smooth transitions between two or more specified colors.
- Earlier done with images for these effects.
  - Reduce download time and bandwidth
- CSS3 gradients look better when zoomed,
  - Since generated by the browser.
- Two types of gradients:
  - **Linear Gradients (goes down/up/left/right/diagonally)**
  - **Radial Gradients (defined by their center)**
  - **Conic Gradients (rotated around a center point)**

sujata batra

CSS Linear Gradients

- To create a linear gradient you must define at least two color stops. Color stops are the colors you want to render smooth transitions among. You can also set a starting point and a direction (or an angle) along with the gradient effect.

Syntax:
background-image: linear-gradient(direction, color-stop1, color-stop2, ...);

Examples

```
#grad {
  background-image: linear-gradient(red, yellow);
}
```

```
#grad {
  background-image: linear-gradient(to right, red , yellow);
}
```

```
#grad {
  background-image: linear-gradient(to bottom right, red, yellow);
}
```

Example


```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
  height: 200px;
  background-color: red; /* For browsers that do not support gradients */
  background-image: linear-gradient(to bottom right, red, yellow);
}
</style>
</head>
<body>
<h1>Linear Gradient - Diagonal</h1>
<p>This linear gradient starts red at top left, transitioning to yellow (at bottom right):</p>
<div id="grad1"></div>
</body>
</html>
```

CSS Radial Gradients

- A radial gradient is defined by its center and at least two color stops.

Syntax:
background-image: radial-gradient(shape size at position, start-color, ..., last-color);

By default, shape is ellipse, size is farthest-corner, and position is center.

Radial Gradient - Evenly Spaced Color Stops (this is default)

Examples:

```
#grad {
  background-image: radial-gradient(red, yellow, green);
}

#grad {
  background-image: radial-gradient(red 5%, yellow 15%, green 60%);
}
```

Set Shape

The shape parameter defines the shape. It can take the value **circle or ellipse**. The default value is ellipse.

Example:

```
#grad {
  background-image: radial-gradient(circle, red, yellow, green);
}
```

Example


```
<!DOCTYPE html>
<html>
<head>
<style>
#grad1 {
  height: 150px;
  width: 200px;
  background-color: red; /* For browsers that do not support gradients */
  background-image: radial-gradient(red, yellow, green);
}
#grad2 {
  height: 150px;
  width: 200px;
  background-color: red; /* For browsers that do not support gradients */
  background-image: radial-gradient(circle, red, yellow, green);
}
</style>
<head>
<body>

<h1>Radial Gradient - Shapes</h1>

<h2>Ellipse (this is default):</h2>
<div id="grad1"></div>

<h2><strong>Circle:</strong></h2>
<div id="grad2"></div>

</body>
</html>
```

CSS Conic Gradients

- A conic gradient is a gradient with color transitions rotated around a center point.
- To create a conic gradient you must define at least two colors.

Syntax

background-image: conic-gradient([from angle] [at position,] color [degree], color [degree], ...);

- By default, angle is 0deg and position is center.
- If no degree is specified, the colors will be spread equally around the center point.

Examples


- ```
  #grad {
    background-image: conic-gradient(red, yellow, green);
  }
  ```
- ```
  #grad {
    background-image: conic-gradient(red, yellow, green, blue,
  black);
  }
  ```
- ```
  #grad {
    background-image: conic-gradient(red 45deg, yellow 90deg,
  green 210deg);
  }
  ```
  ```

Formatting with CSS Properties

CSS Background Shorthand

You can also specify all the properties in a single property.
This property is known as shorthand property.

For specifying shorthand property, you just need to use **background**.

Example-

```
body {
        background:cyan url('SomeImage.jpg') no-repeat right top;
        }
```

Formatting with CSS Properties

Text Formatting

The following properties can be used for formatting text :

1. Text Color
2. Text Alignment
3. Text Decoration
4. Text Transformation
5. Text Indentation

Formatting with CSS Properties

Text Alignment

We can either align the text to the left, right, center or we can make it justified.

Example-

```
p { text-align:left;}
h1{text-align:center;}
```

Text Color

The color property is used to set the color of text.

Example-

```
body { color:blue;}
p1 {color:magenta;}
```

Formatting with CSS Properties

Text Decoration

You can use **text-decoration** property to set or remove decorations from text.

Example-

```
p {text-decoration:overline;}
p {text-decoration:line-through;}
p {text-decoration:underline;}
```

Text Transformation

You can use text-transform property to specify uppercase and lowercase letters of any text.

Example-

```
h1 {text-transform:uppercase;}
h2 {text-transform:lowercase;}
p {text-transform:capitalize;}
```

Formatting with CSS Properties

CSS Font

CSS font properties are used to define the font family, size, style and boldness of the text.
In CSS, there are two types of font family names:

generic family - a group of font families with a similar look (like "Serif" or "Monospace").
font family - a specific font family (like "Times New Roman" or "Arial").

Comments in CSS

/* comment */ - This is comment used in CSS.

Formatting with CSS Properties

CSS Font Family

The font-family property should hold several font names as a "fallback" system. If the browser does not support the first font, it tries the next font.
Example :
p { font-family:"Arial", Times, "Sans-serif";}

CSS Font Style

You can use the property font-style to specify mostly italic text. It has three values – Normal, Italic, Oblique (similar to italic).

Formatting with CSS Properties

CSS Font Size

You can use the **font-size** property to set the size of text. The font-size value can be absolute or it can be relative.

Example-

```
h1 {
    font-size: 30px;
}

p {
    font-size: 14px;
}
```

Formatting with CSS Properties

CSS Font Size with em (Relative Size)

You may face resizing problems, when you use older versions of browsers.
To avoid such problems, you can use set font size using em, instead of pixels.

The em size unit is a W3C recommendation.1 em is equal to the current font size.
The default text size is 16 px. So, the default size of 1 em is 16 px.

Example

```
h2 {
    font-size: 1.875em; /* 30px/16=1.875em */
}

p {
    font-size: 0.875em; /* 14px/16=0.875em */
}
```

CSS Units

- CSS has several different units for expressing a length.
- Many CSS properties take "length" values, such as width, margin, padding, font-size, border-width, etc.
- Length is a number followed by a length unit, such as 10px, 2em, etc.
- A whitespace cannot appear between the number and the unit. However, if the value is 0, the unit can be omitted.
- There are two types of length units:
  - absolute
  - relative.

CSS UNITS - Sizes

- Relative length measurements:
  - px (pixels – size varies depending on screen resolution)
  - em (usually the height of a font's uppercase)
  - ex (usually the height of a font's lowercase)
  - Percentages (of the font's default size)
- Absolute-length measurements (units that do not vary in size):
  - in (inches)
  - cm (centimeters)
  - mm (millimeters)
  - pt (points; 1 pt = 1/72 in)
  - pc (picas; 1 pc = 12 pt)

Formatting with CSS Properties

CSS Links

You can use CSS styles to style any link. Links can be styled in different ways by using any CSS property like color, font-family etc.

Links can be in one of the following states :

- **a: link** – Unvisited link
- **a: visited** – A link that the user has visited
- **a: hover** – A link over which the mouse pointer is moving
- **a: active** – A link, which has been just clicked

Links can be styled according to their states.

Formatting with CSS Properties

CSS Links

```
a {
 font-weight: bold;
}
a:link {
  color: black;
}
a:visited {
 color: gray;
}
a:hover {
 text-decoration: none;
 color: white;
 background-color: navy;
}
a:active {
 color: aqua;
 background-color: navy;
}
```

Styling Links

link - before a visit

visited - after it has been visited

hover - when your mouse is over it but you have not clicked

active - you have clicked it and you have not yet seen the new page

Formatting with CSS Properties

CSS List

You can use CSS list properties for

- Setting different list item markers for ordered lists
- Setting different list item markers for unordered lists
- Set an image as the list item marker

Values-

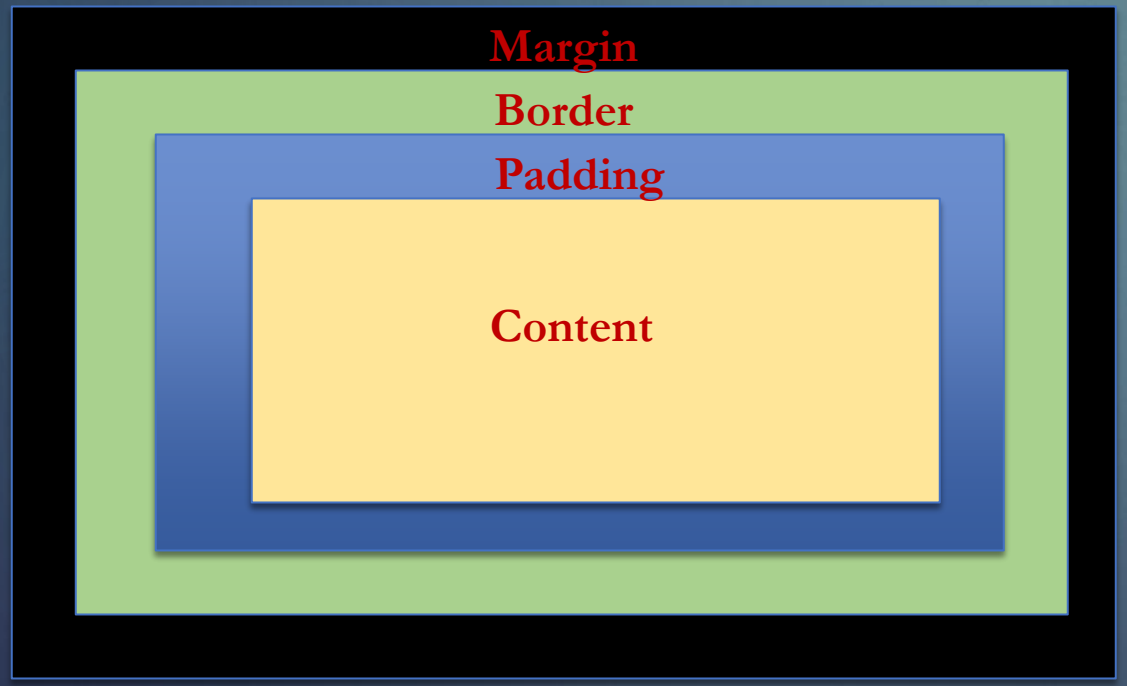
- list-style-type
- list-style-image

Formatting with CSS Properties

Box Model : Introduction

Box model is useful for designing the layout of an HTML Page.
CSS Box model describes a box that wraps around HTML elements.

Using this model, we can define the margins, borders, padding and the actual content. We can place border around elements and space elements in relation to each other.



Box Model : Illustration

You can set the height and width of an element using the **height** and **width** properties.

Formatting with CSS Properties

CSS Padding

You can use the CSS padding properties to define the space between the element border and the element content. It is possible to change the top, right, bottom and left padding independently using separate properties.

In shorthand; specify one, two, three, or four space-separated values:

- [value] [top, right, bottom, and left] 10px
- [value] [value] [top and bottom] [left and right] 10px 20px
- [value] [value] [value][top] [right and left] [bottom] 10px 20px 30px
- [value] [value] [value] [value] [top] [right] [bottom] [left] 10px 20px 30px 40px

- **Note:** padding: 10px 20px 30px 40px; **is same as** padding-top: 10px; padding-right: 20px; padding-bottom: 30px; padding-left: 40px;

Formatting with CSS Properties

CSS Border

You can use the CSS Border properties to specify the style and color of an element's border.

- **border-style**
  - **To make a border around an element.**
  - **Values: solid, dotted, dashed, double, groove, ridge, inset and outset.**
- **border-width**
  - **sets the width of the border**
  - **There are also properties for border-top-width, border-right-width, border-bottom-width and border-left-width.**
- **border-color**
  - **sets the color.**

Border-width

- In shorthand; specify one, two, three, or four space-separated values:
  - [value] [top, right, bottom, and left] 10px
  - [value] [value] [top and bottom] [left and right] 10px 20px
  - [value] [value] [value] [top] [right and left] [bottom] 10px 20px 30px
  - [value] [value] [value] [value] [top] [right] [bottom] [left] 10px 20px 30px 40px

- **Note :** border-width: 10px 20px 30px 40px; **is same** as border-top-width: 10px; border-right-width: 20px; border-bottom-width: 30px; border-left-width: 40px;

Formatting with CSS Properties

CSS Margin

Using CSS Margin properties you can specify the space around elements.

- **One, two, three, or four space-separated values:**
  - **[value] [top, right, bottom, and left] 10px**
  - **[value] [value] [top and bottom] [left and right] 10px 20px**
  - **[value] [value] [value] [top] [right and left] [bottom] 10px 20px 30px**
  - **[value] [value] [value] [value] [top] [right] [bottom] [left] 10px 20px 30px 40px**

- **Note : margin: 10px 20px 30px 40px; is same as margin-top: 10px; margin-right: 20px; margin-bottom: 30px; margin-left: 40px;, for example.**

CSS Image Sprites

- An image sprite is a collection of images put into a single image.
- A web page with many images can take a long time to load and generates multiple server requests.
- Using image sprites will reduce the number of server requests and save bandwidth.

Example


```
<!DOCTYPE html>
<html>
<head>
<style>
#home {
  width: 46px;
  height: 44px;
  background: url(img_navsprites.gif) 0 0;
}

#next {
  width: 43px;
  height: 44px;
  background: url(img_navsprites.gif) -91px 0;
}
</style>
</head>
<body>

<img id="home" src="img_trans.gif" width="1" height="1">
<img id="next" src="img_trans.gif" width="1" height="1">

</body>
</html>
```

Image Sprites - Create a Navigation List


```
<!DOCTYPE html>
<html>
<head>
<style>
#navlist {
  position: relative;
}

#navlist li {
  margin: 0;
  padding: 0;
  list-style: none;
  position: absolute;
  top: 0;
}

#navlist li, #navlist a {
  height: 44px;
  display: block;
}

#home {
  left: 0px;
  width: 46px;
  background: url('img_navsprites.gif') 0 0;
}

#prev {
  left: 63px;
  width: 43px;
  background: url('img_navsprites.gif') -47px 0;
}

#next {
  left: 129px;
  width: 43px;
  background: url('img_navsprites.gif') -91px 0;
}
</style>
</head>

<body>

<ul id="navlist">
  <li id="home"><a href="default.asp"></a></li>
  <li id="prev"><a href="css_intro.asp"></a< /li>
  <li id="next"><a href="css_syntax.asp"></a></li>
</ul>

</body>
</html>
```

Cascade

- The cascade sorts out all conflicts when multiple declarations would affect a given element.
- Important declarations will override less important ones.
- Among declarations with equal importance, the rule's specificity controls which one will apply.
- And, all else being equal, the source order makes the final distinction.

sujata batra

Cascading

- It's the mechanism that controls the end result when multiple, conflicting CSS declarations apply to the same element.
- Its controlled by the order in which CSS declarations are applied:
  - Importance
  - Specificity
  - Source order

Sujata Batra

Specificity

- It can be thought of as a measure of how specific a rule's selector is.
- A selector with low specificity may match many elements (like * which matches every element in the document),
- while a selector with high specificity might only match a single element on a page (like #nav that only matches the element with an id of nav).
- If two or more declarations conflict for a given element, and all the declarations have the same importance, then the one in the rule with the most specific selector will "win".

Sujata Batra

Source order

- If two declarations affect the same element, have the same importance and the same specificity, the final distinguishing mark is the source order.
- The declaration that appears later in the style sheets will "win" over those that come before it.

Inheritance

- Process by which properties are passed from parent to child elements
  - even though those properties have not been explicitly defined by other means.
- Certain properties are inherited automatically
  - styles that apply to text are inherited,
  - borders, margins and paddings and similar styles are not.
- Inheritance and the Cascade
  - inheritance applies to the DOM tree
  - cascade deals with the style sheet rules.

Sujata Batra

Inheritance

```
<style>
    p {
        color: red;
     border:2px solid blue;
    }
    strong{
        border:inherit ;
    }
  </style>
```

-
- This is the First Paragraph
- The second line of this paragraph is also created with the styles.
-

Sujata Batra

ARIA

- Accessible Rich Internet Applications (ARIA) is a set of roles and attributes that define ways to make web content and web applications (especially those developed with JavaScript) more accessible to people with disabilities.

- It supplements HTML so that interactions and widgets commonly used in applications can be passed to assistive technologies when there is not otherwise a mechanism. For example, ARIA enables accessible JavaScript widgets, form hints and error messages, live content updates, and more.

BootStrap

What is Bootstrap ?

- Bootstrap is a free front-end framework for faster and easier web development
- Bootstrap includes HTML and CSS based design templates for typography, forms, buttons, tables, navigation, modals, image carousels and many other, as well as optional JavaScript plugins
- Bootstrap also gives you the ability to easily create responsive designs

What is Responsive Web Design?

- Responsive web design is about creating web sites which automatically adjust themselves to look good on all devices, from small phones to large desktops.

Advantages of Bootstrap

- **Easy to use:** Anybody with just basic knowledge of HTML and CSS can start using Bootstrap
- **Responsive features:** Bootstrap's responsive CSS adjusts to phones, tablets, and desktops
- **Mobile-first approach:** In Bootstrap, mobile-first styles are part of the core framework
- **Browser compatibility:** Bootstrap 5 is compatible with all modern browsers (Chrome, Firefox, Edge, Safari, and Opera).

Note : if you need support for IE11 and down, you must use either BS4 or BS3.

Where to Get Bootstrap 5?

- Include Bootstrap 5 from a CDN

OR

- Download Bootstrap 5 from getbootstrap.com

Downloading Bootstrap 5 from **https://getbootstrap.com/**, and follow the instructions there.

Creating Web Page With Bootstrap 5

- Add the HTML5 doctype

Bootstrap 5 uses HTML elements and CSS properties that require the HTML5 doctype.

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <title>Bootstrap 5 Example</title>
    <meta charset="utf-8">
  </head>
</html>
```

Creating Web Page With Bootstrap 5

- Bootstrap 5 is mobile-first

Bootstrap 5 is designed to be responsive to mobile devices. Mobile-first styles are part of the core framework.

To ensure proper rendering and touch zooming, add the following <meta> tag inside the <head> element:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Creating Web Page With Bootstrap 5

- Containers
  - Bootstrap 5 also requires a containing element to wrap site contents.
  - There are two container classes to choose from:
    - The .container class provides a responsive fixed width container
    - The .container-fluid class provides a full width container, spanning the entire width of the viewport

Example


```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>Bootstrap Example</title>
  <meta charset="utf-8">
  <meta name="viewport" content="width=device-width, initial-scale=1">
  <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/css/bootstrap.min.css" rel="stylesheet">
  <script src="https://cdn.jsdelivr.net/npm/bootstrap@5.2.3/dist/js/bootstrap.bundle.min.js"></script>
</head>
<body>

<div class="container-fluid">
  <h1>My First Bootstrap Page</h1>
  <p>This part is inside a .container-fluid class.</p>
  <p>The .container-fluid class provides a full width container, spanning the entire width of the viewport.</p>
</div>

</body>
</html>
```