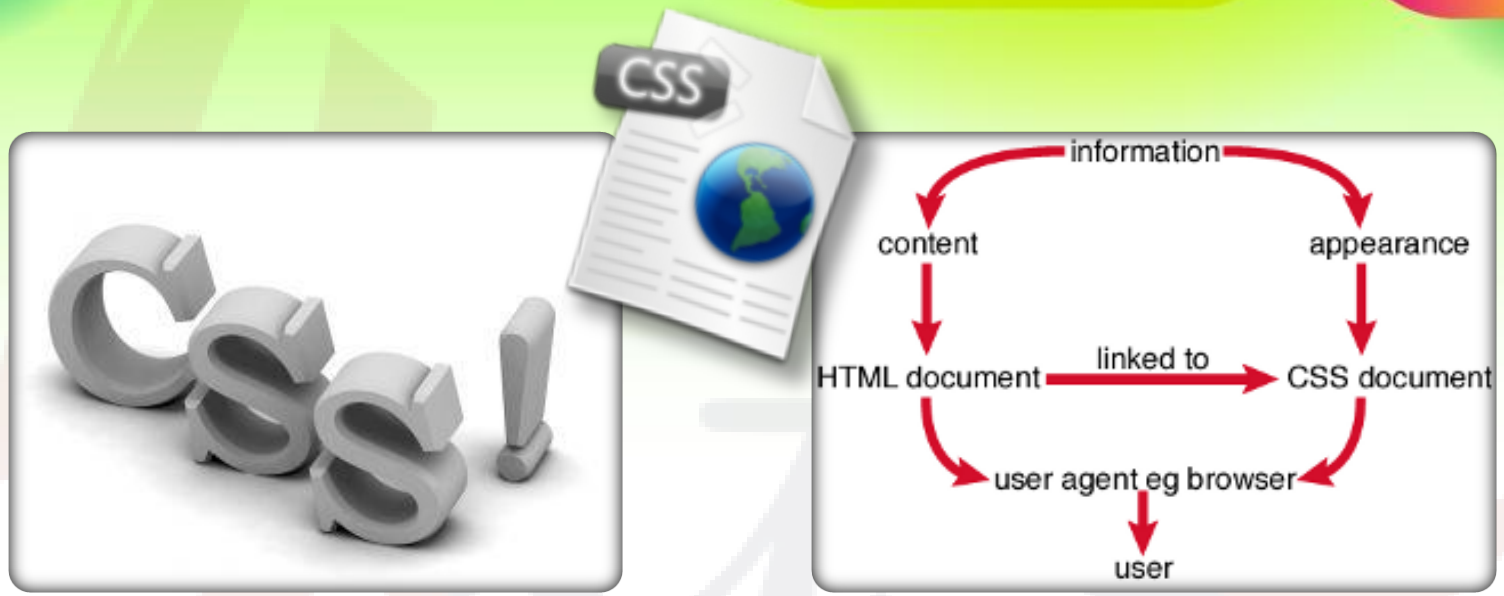
CSS

By
Sudha Agarwal

INDEX

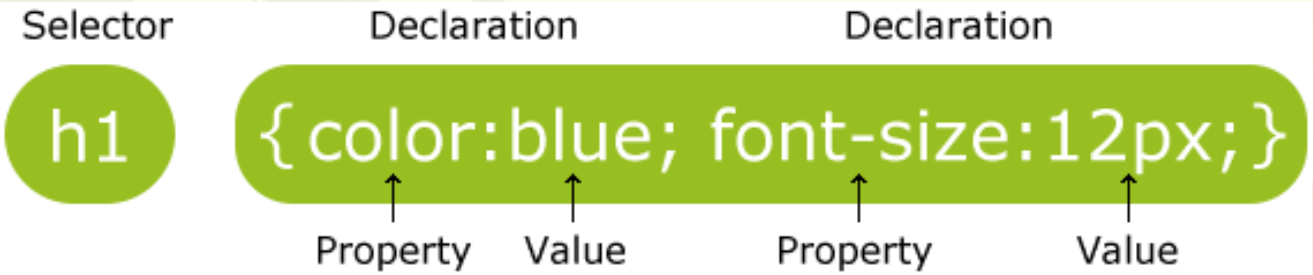

CSS
CSS!
information
content
appearance
HTML document
linked to
CSS document
user agent eg browser
user

WHAT IS CSS

- Cascading Style Sheets (CSS)
  - Used to describe the presentation of documents
  - Define sizes, spacing, fonts, colors, layout, etc.
  - Improve content accessibility
  - Improve flexibility
  - Saves a lot of work. It can control the layout of multiple web pages all at once.
- Designed to separate presentation from content
- Due to CSS, all HTML presentation tags and attributes are deprecated, e.g. font, center, etc.

CSS SYNTAX

❖A CSS rule-set consists of a selector and a declaration block:



❖The selector points to the HTML element you want to style.

❖The declaration block contains one or more declarations separated by semicolons.

❖Each declaration includes a CSS property name and a value, separated by a colon.

❖A CSS declaration always ends with a semicolon, and declaration blocks are surrounded by curly braces.

CSS SELECTORS

- CSS selectors allow you to select and manipulate HTML elements.
- CSS selectors are used to "find" (or select) HTML elements based on their id, class, type, attribute, and more.
- Three primary Selectors are:
  - The element Selector
  - The id Selector
  - The class Selector

THE ELEMENT SELECTOR

❖The element selector selects elements based on the element name.

❖We can select all <p> elements on a page like this: (all <p> elements will be center-aligned, with a red text color)

```
p {
    text-align: center;
    color: red;
}
```

THE ID SELECTOR

- The id selector uses the id attribute of an HTML element to select a specific element.
- An id should be unique within a page, so the id selector is used if you want to select a single, unique element.
- To select an element with a specific id, write a hash character, followed by the id of the element.
- The style rule below will be applied to the HTML element with id="para1":

```
#para1 {
    text-align: center;
    color: red;
}
```

THE CLASS SELECTOR

- The class selector selects elements with a specific class attribute.
- To select elements with a specific class, write a period character, followed by the name of the class:
- In the example below, all HTML elements with class="center" will be center-aligned:

```
.center {
    text-align: center;
    color: red;
}
```

- HTML elements can also refer to more than one class.

```
<p class="center large">This paragraph
refers to two classes.</p>
```

- Selectors can be combined with commas:

```
h1, .link, #top-link {font-weight: bold}
```

CSS COMMENTS

❖Comments are used to explain your code, and may help you when you edit the source code at a later date. Comments are ignored by browsers.

❖A CSS comment starts with /* and ends with */. Comments can also span multiple lines:

```
p {
    color: red;
    /* This is a single-line comment */
    text-align: center;
}
/* This is
a multi-line
comment */
```

LINKING HTML AND CSS

- HTML (content) and CSS (presentation) can be linked in three ways
  - External style sheet - CSS rules in separate file (best)
    - Usually a file with .css extension
    - Linked via <link rel="stylesheet" href=…> tag or @import directive in embedded CSS block
  - Internal style sheet - in the <head> in a <style> tag
  - Inline style - the CSS rules in the style attribute. No selectors are needed

INTERNAL STYLESHEET

```
<!DOCTYPE html>
<head>
<style>

body { background-color: blue; }
p { color: white; }

</style>
</head>
<body>
<h2>Internal CSS</h2>
<p>This page uses internal CSS.  Using the style
tag we are able to modify
the appearance of HTML elements.</p>

</body>
</html>
```

INLINE STYLESHEET

```
<!DOCTYPE html>
<head>
  <title>Inline Styles</title>
</head>
<body>
  <p>Here is some text</p>
<!--Separate multiple styles with a semicolon-->
  <p style="font-size: 20pt">Here is some
    more text</p>
  <p style="font-size: 20pt;color:
    #0000FF" >Even more text</p>
</body>
</html>
```

EXTERNAL STYLESHEET

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

- mystyle.css

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

h1 {
    color: navy;
    margin-left: 20px;
}
```

IMPORTED CSS - @IMPORT RULE

- @import is used to import an external stylesheet in a manner similar to the <link> element

```
<head>
   <@import "URL";
</head>
```

- Example

```
<head>
   @import "mystyle.css";
</head>
```

MULTIPLE STYLE SHEETS

❖If some properties have been defined for the same selector in different style sheets, the value will be inherited from the more specific style sheet.

❖For example, assume that an external style sheet has the following properties for the <h1> element:

```
h1 {
    color: navy;
    margin-left: 20px;
}
```

MULTIPLE STYLE SHEETS

❖then, assume that an internal style sheet also has the following property for the <h1> element:

```
h1 {
    color: orange;
}
```

❖If the page with the internal style sheet also links to the external style sheet the properties for the <h1> element will be:

```
h1 {
color: orange;
margin-left: 20px;
}
```

❖The left margin is inherited from the external style sheet and the color is replaced by the internal style sheet.

CASCADING ORDER

❖Generally speaking we can say that all the styles will "cascade" into a new "virtual" style sheet by the following rules, where number one has the highest priority:

- ➢Inline style (inside an HTML element)
- ➢External and internal style sheets (in the head section)
- ➢Browser default

❖So, an inline style (inside a specific HTML element) has the highest priority, which means that it will override a style defined inside the <head> tag, or in an external style sheet, or a browser default value.

MEASUREMENT UNITS

❖CSS Measurement Units

Unit	Description	Example
%	Defines a measurement as a percentage relative to another value, typically an enclosing element.	p {font-size: 16pt; line-height: 125%;}
cm	Defines a measurement in centimeters.	div {margin-bottom: 2cm;}
em	A relative measurement for the height of a font in em spaces. Because an em unit is equivalent to the size of a given font, if you assign a font to 12pt, each "em" unit would be 12pt; thus, 2em would be 24pt.	p {letter-spacing: 7em;}
px	Defines a measurement in screen pixels.	p {padding: 25px;}

CSS BACKGROUNDS

❖ The CSS background properties are used to define the background effects for elements.

❖CSS background properties:

- ➢background-color
- ➢background-image
- ➢background-repeat
- ➢background-attachment
- ➢background-position

CSS BACKGROUNDS

❖Background Color

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

Color is most often specified by:
•HEX value - like "#ff0000“
•RGB value - like "rgb(255,0,0)“
•Valid color name - like "red"

❖Background Image

```
body {
    background-image: url("paper.gif");
}
```

❖Background Image - Repeat Horizontally or Vertically

```
body {
    background-image: url("gradient_bg.png");
    background-repeat: repeat|repeat-x|repeat-y;
}
```

CSS BACKGROUNDS

❖Background Image - no-repeat

```
body {
    background-image: url("img_tree.png");
    background-repeat: no-repeat;
}
```

❖Background Image – set position

```
body {
    background-image: url("img_tree.png");
    background-repeat: no-repeat;
    background-position: right top|x% y%|xpos ypos;
}
```

CSS BACKGROUNDS

❖Background Image – fixed position

```
body {
    background-image: url("img_tree.png");
    background-repeat: no-repeat;
    background-position: right top;
    background-attachment: scroll|fixed;
}
```

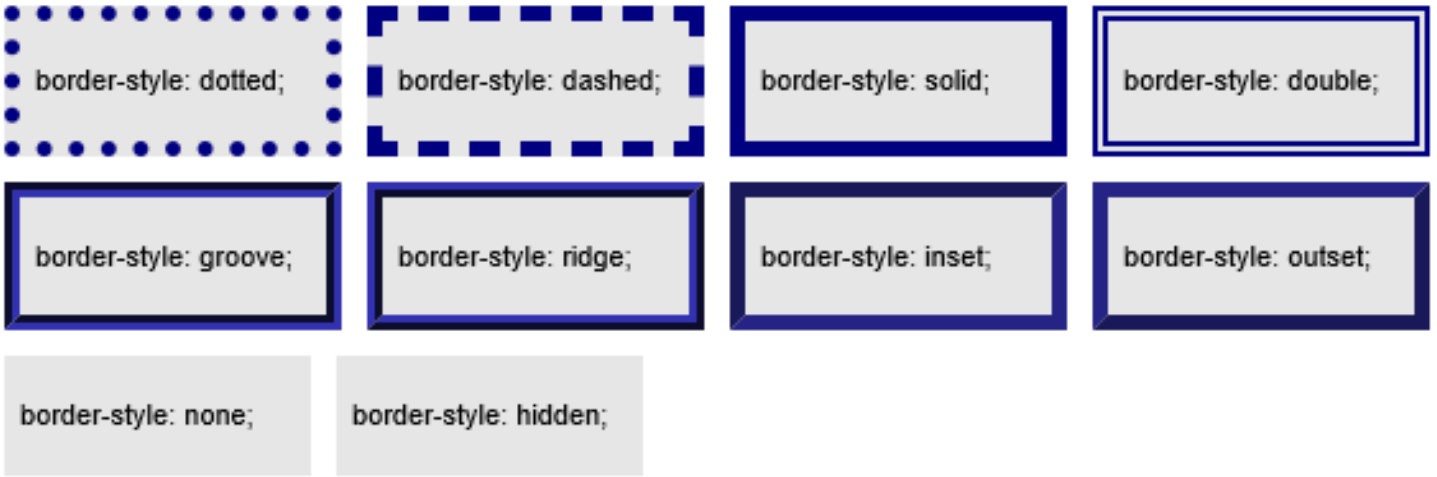
❖Background - Shorthand property

```
body {
    background: #ffffff url("img_tree.png") no-
repeat right top;
}
```

Example Example Example Example

CSS BORDERS

- The CSS border properties allow us to specify the style, width, and color of an element's border.
- There are three properties of a border we can change:
  - The **border-color** specifies the color of a border.
  - The **border-style** specifies whether a border should be solid, dashed line, double line, or one of the other possible values.
  - The **border-width** specifies the width of a border.



CSS BORDERS - BORDER STYLE

❖The border-style property specifies what kind of border to display.

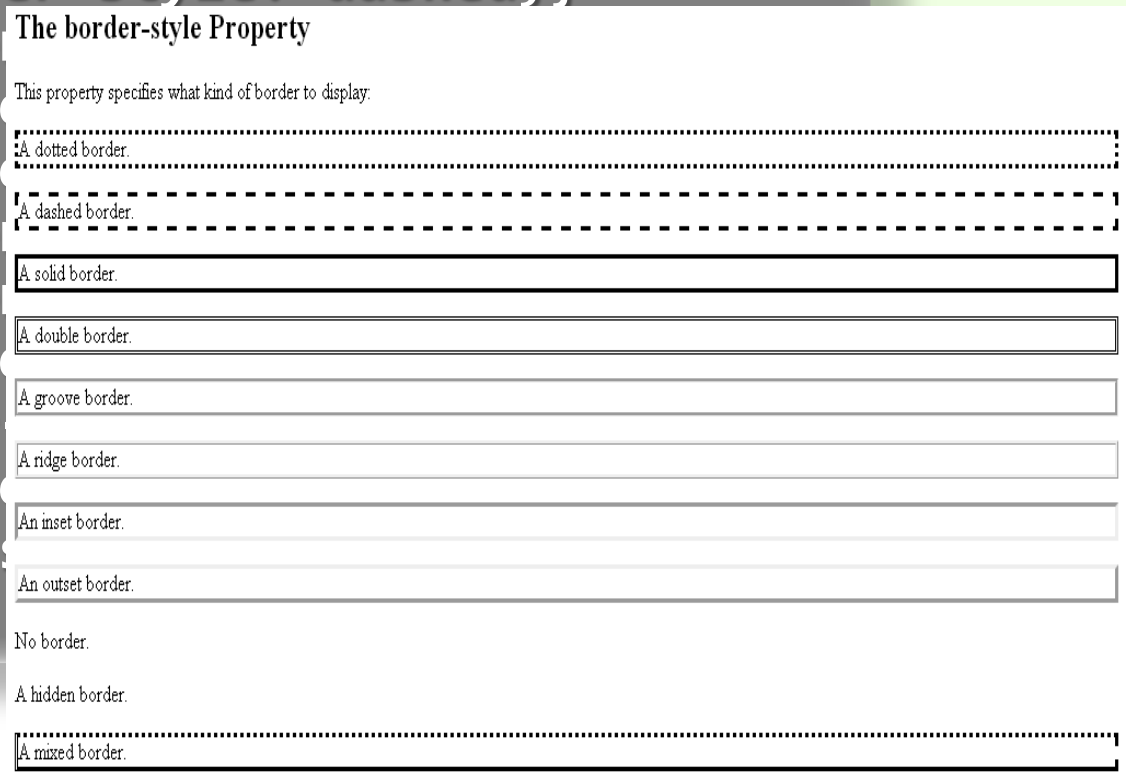
❖The following values are allowed:

- **dotted** - Defines a dotted border
- **dashed** - Defines a dashed border
- **solid** - Defines a solid border
- **double** - Defines a double border
- **groove** - Defines a 3D grooved border. The effect depends on the border-color value
- **ridge** - Defines a 3D ridged border. The effect depends on the border-color value
- **inset** - Defines a 3D inset border. The effect depends on the border-color value
- **outset** - Defines a 3D outset border. The effect depends on the border-color value
- **none** - Defines no border
- **hidden** - Defines a hidden border

❖The border-style property can have from one to four values (for the top border, right border, bottom border, and the left border).

CSS BORDERS - BORDER STYLE

```
p.dotted {border-style: dotted;}
p.dashed {border-style: dashed;}
p.solid {borde
p.double {bord
p.groove {bord
p.ridge {borde
p.inset {borde
p.outset {bord
p.none {border
p.hidden {bord
p.mix {border-
double;}
```



CSS BORDERS - BORDER WIDTH

- The width can be set as a specific size (in px, pt, cm, em, etc) or by using one of the three pre-defined values: thin, medium, or thick.
- The border-width property can have from one to four values (for the top border, right border, bottom border, and the left border).

```
p.one {
    border-style: solid;
    border-width: 5px;
}
p.two {
    border-style: solid;
    border-width: medium;
}
p.three {
    border-style: solid;
    border-width: 2px 10px 4px 20px;
}
```

CSS BORDERS - BORDER COLOR

- The color can be set by:
  - name - specify a color name, like "red"
  - RGB - specify a RGB value, like "rgb(255,0,0)"
  - Hex - specify a hex value, like "#ff0000"
  - Transparent

```
p.one {
    border-style: solid;
    border-color: red;
}
p.two {
    border-style: solid;
    border-color: #ff0085;
}
p.three {
    border-style: solid;
    border-color: red green blue yellow;
}
```

BORDER - SHORTHAND PROPERTY

❖The border property is a shorthand property for the following individual border properties:

- ➢border-width
- ➢border-style (required)
- ➢border-color

```
p {
    border: 5px solid red;
}
```

Example Example Example Example

CSS MARGINS

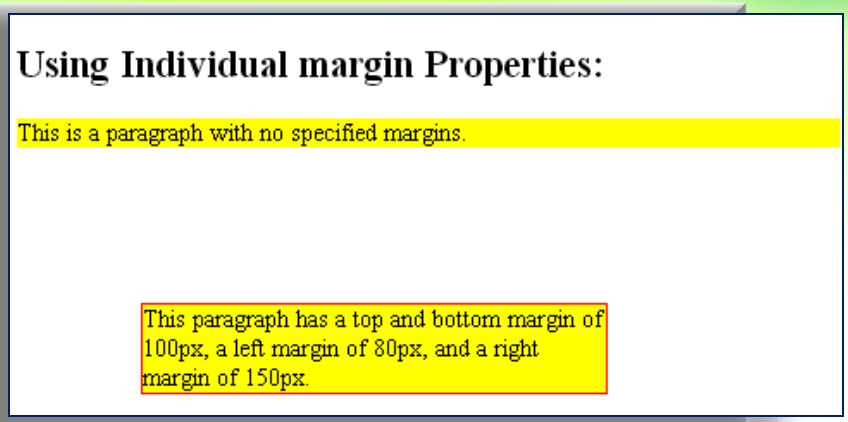
- The margin property defines the space around an HTML element.
- The margins are completely transparent and cannot have a background color.
- CSS has properties for specifying the margin for each side of an element:
  - margin-top
  - margin-right
  - margin-bottom
  - margin-left
- All the margin properties can have the following values:
  - **auto** - the browser calculates the margin
  - **length** - specifies a margin in px, pt, cm, etc.
  - **%** - specifies a margin in % of the width of the containing element
  - **inherit** - specifies that the margin should be inherited from the parent element

CSS MARGIN

```
<style>
p {
    background-color: yellow;
}
p.ex {
    border:1px solid red;
    margin-top: 100px;
    margin-bottom: 100px;
    margin-right: 150px;
    margin-left: 80px;
}
</style>

<h2>Using Individual margin Properties:</h2>

<p>This is a paragraph with no specified margins.</p>
<p class="ex">This paragraph has a top and bottom margin of
100px, a left margin of 80px, and a right margin of
150px.</p>
```

CSS MARGIN

```
<style>
p {
    background-color: yellow;
}
p.ex {
    border:1px solid red;
    margin-top: 100px;
    margin-bottom: 100px;
    margin-right: 150px;
    margin-left: 80px;
}
</style>

<h2>Using Individual margin Properties:</h2>

<p>This is a paragraph with no specified margins.</p>
<p class="ex">This paragraph has a top and bottom margin of
100px, a left margin of 80px, and a right margin of
150px.</p>
```



MARGIN - SHORTHAND PROPERTY

❖The margin property is a shorthand property for the following individual margin properties:

- ➢margin-top
- ➢margin-right
- ➢margin-bottom
- ➢margin-left

```
p {
    margin: 100px 150px 100px 80px;
}
```

Example Example

USE OF THE AUTO VALUE

❖We can set the margin property to auto to horizontally center the element within its container.

❖The element will then take up the specified width, and the remaining space will be split equally between the left and right margins:

```
<style>
div {
    width:300px;
    margin: auto;
    border: 1px solid red;
}
</style>
</head>
<div>This div will be centered because it has
margin: auto;
</div>
```

USE OF THE AUTO VALUE

❖We can set the margin property to auto to horizontally center the element within its container.

❖The element will then take up the specified width, and the remaining space will be split equally between the left and right margins:

```
<style>
div {
    width:300px;
    margin: auto;
    border: 1px solid red;
}
</style>
</head>
<div>This div will be centered because it has
margin: auto;
</div>
```

This div will be centered because it has margin: auto;

CSS PADDING

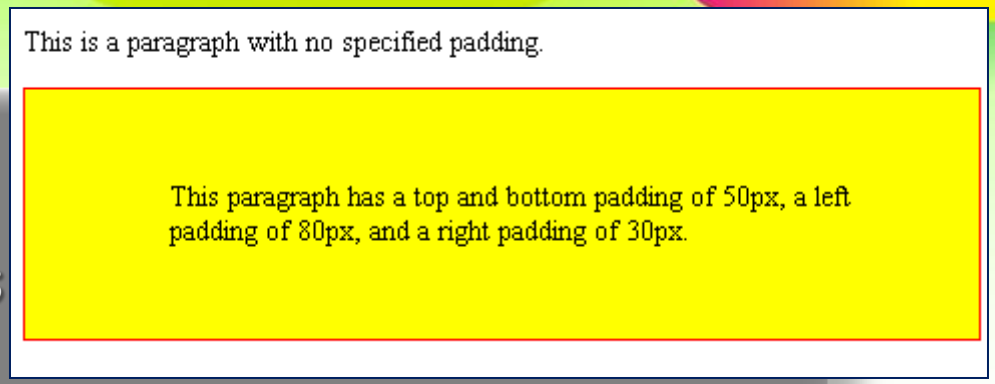
- The padding property allows us to specify how much space should appear between the content of an element and its border.
- Padding clears an area around the content (inside the border) of an element.
- CSS has properties for specifying the padding for each side of an element:
  - padding-top
  - padding-right
  - padding-bottom
  - padding-left
- All the padding properties can have the following values:
  - length - specifies a padding in px, pt, cm, etc.
  - % - specifies a padding in % of the width of the containing element
  - inherit - specifies that the padding should be inherited from the parent element

CSS PADDING

```
<style>
p.one {
    border: 1px solid red;
    background-color: yellow;
    padding-top: 50px;
    padding-right: 30px;
    padding-bottom: 50px;
    padding-left: 80px;
}
</style>
<p>This is a paragraph with no specified padding.</p>
<p class="one">This paragraph has a top and bottom padding
of 50px, a left padding of 80px, and a right padding of
30px.</p>
```

CSS PADDING

```
<style>
p.one {
    border: 1px solid red;
    background-color: yellow;
    padding-top: 50px;
    padding-right: 30px;
    padding-bottom: 50px;
    padding-left: 80px;
}
</style>
<p>This is a paragraph with no specified padding.</p>
<p class="one">This paragraph has a top and bottom padding
of 50px, a left padding of 80px, and a right padding of
30px.</p>
```



PADDING - SHORTHAND PROPERTY

❖The padding property is a shorthand property for the following individual padding properties:

- ➢padding-top
- ➢padding-right
- ➢padding-bottom
- ➢padding-left

```
p {
    padding: 50px 30px 50px 80px;
}
```

Example Example

CSS HEIGHT AND WIDTH

- The height and width properties are used to set the height and width of an element.
- The height and width :
  - can be set to **auto**(this is default. Means that the browser calculates the height and width)
  - can be specified in length values, like **px**, **cm**, etc., or in **percent** (%) of the containing block.

```
div {
    width: 500px;
    height: 100px;
    border: 3px solid #73AD21;
}
```

CSS TEXT

- Text Color

```
<p style="color:red;">
      This text will be written in red.
</p>
```

- Text Alignment

```
h1 {
    text-align: center|left|right|justify;
}
```

- Text Decoration

```
a {
    text-decoration:
none|underline|overline|line-through;
}
```

CSS TEXT

❖Text Transformation

```
p{
    text-transform:
none|capitalize|uppercase|lowercase;
}
```

❖Text Indentation

```
p {
    text-indent: Length|%;
}
```

❖Letter Spacing

```
h1 {
    letter-spacing: normal|length;
}
```

Negative values are allowed

CSS TEXT

- Line Height

```
h1 {
    letter-spacing: normal|length;
}
```

Negative values are allowed

- Text Direction

```
div {
    direction: rtl|ltr;
}
```

- Letter Spacing

```
p.small {
    line-height: normal|number|length;
}
```

CSS FONT

- The font family of a text is set with the font-family property.
- 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, and so on.

```
p {
    font-family: "Times New Roman", Times,
serif;
}
```

- The **font-style** property has three values:
  - normal - The text is shown normally
  - italic - The text is shown in italics
  - oblique - The text is "leaning" (oblique is very similar to italic, but less supported)

CSS FONT

- Font style

```
p.normal {
    font-style: normal|italic|oblique;
}
```

- The **font-size** property sets the size of the text.
- If we do not specify a font size, the default size for normal text, like paragraphs, is 16px (16px=1em).

```
h1 {
    font-size: 2.5em; /* 40px/16=2.5em */
}
```

CSS FONT

❖The font-weight property specifies the weight of a font:

```
p.normal {
    font-weight:
normal|bold|bolder|lighter|number;
}
```

❖Number is in the range of 100 to 900

❖The **font-variant** property specifies whether or not a text should be displayed in a small-caps font.

```
p {
    font-variant: normal|small-caps;
}
```

Example Example Example Example

CSS LINKS

- The four links states are:
  - a:link - a normal, unvisited link
  - a:visited - a link the user has visited
  - a:hover - a link when the user mouse over it
  - a:active - a link the moment it is clicked

```
/* unvisited link */  a:link {
    color: red;
}
/* visited link */    a:visited {
    color: green;
}
/* mouse over link */ a:hover {
    color: hotpink;
}
/* selected link */   a:active {
    color: blue;
}
```

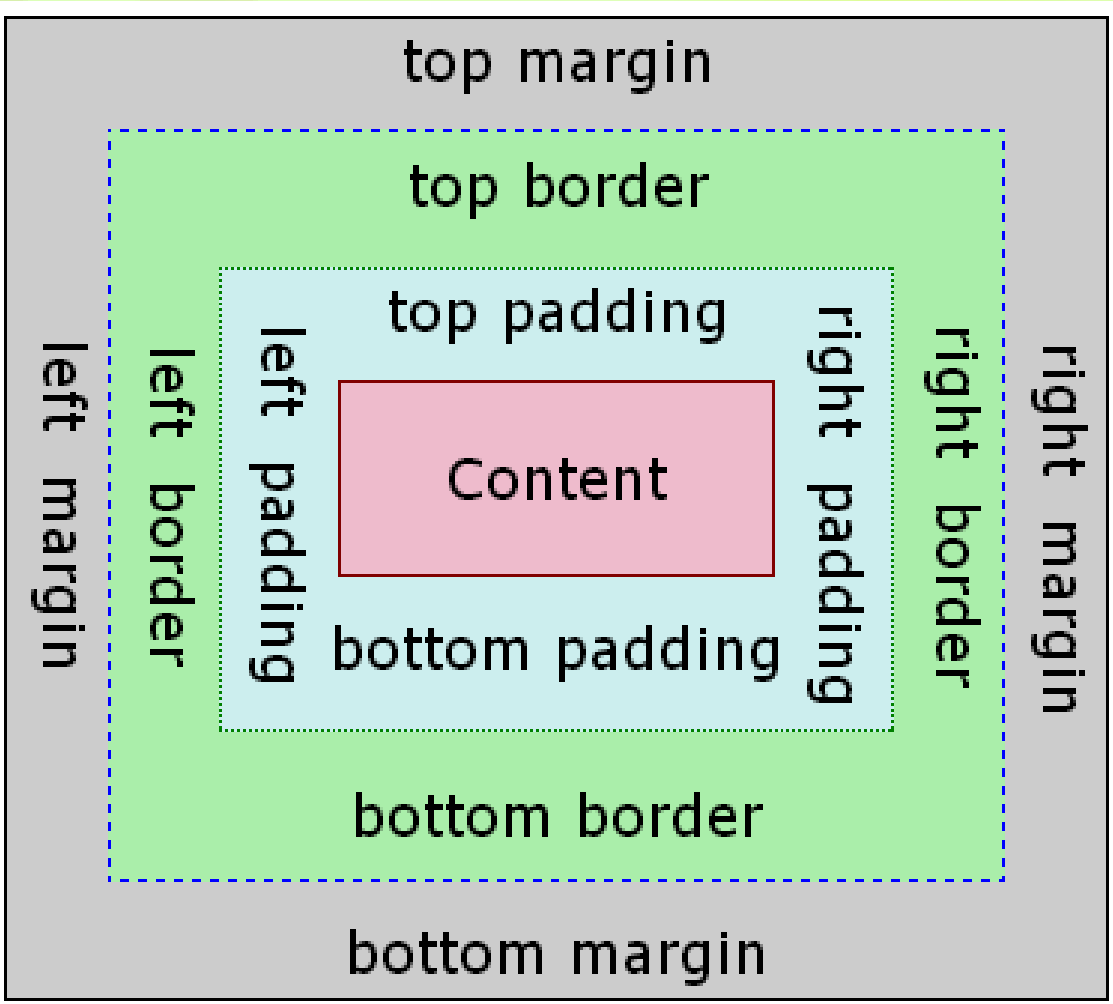
Example Example

CSS BOX MODEL

❖All HTML elements can be considered as boxes. In CSS, the term "box model" is used when talking about design and layout.

❖The CSS box model is essentially a box that wraps around every HTML element. It consists of: margins, borders, padding, and the actual content.

CSS BOX MODEL



BOX MODEL

❖The box model allows us to add a border around elements, and to define space between elements.

```
div {
    background-color: lightgrey;
    width: 300px;
    padding: 25px;
    border: 25px solid navy;
    margin: 25px;
}
```

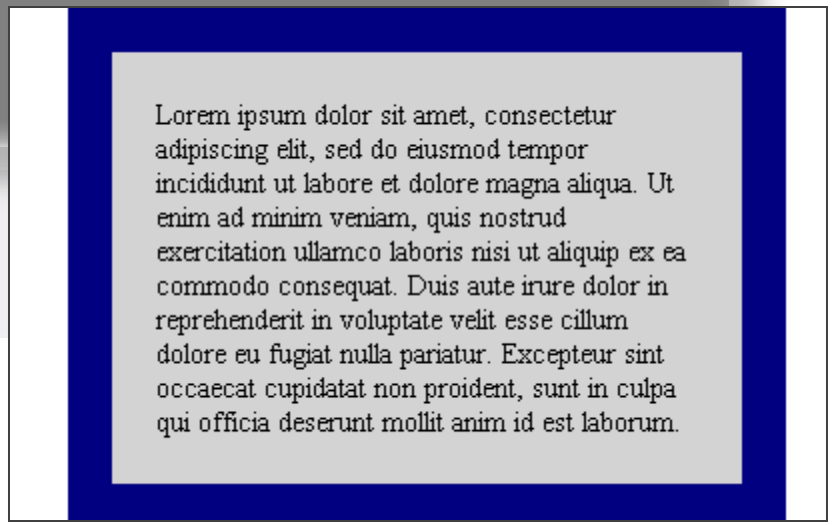
Example

Example

BOX MODEL

❖The box model allows us to add a border around elements, and to define space between elements.

```
div {
    background-color: lightgrey;
    width: 300px;
    padding: 25px;
    border: 25px solid navy;
    margin: 25px;
}
```



CSS OUTLINE

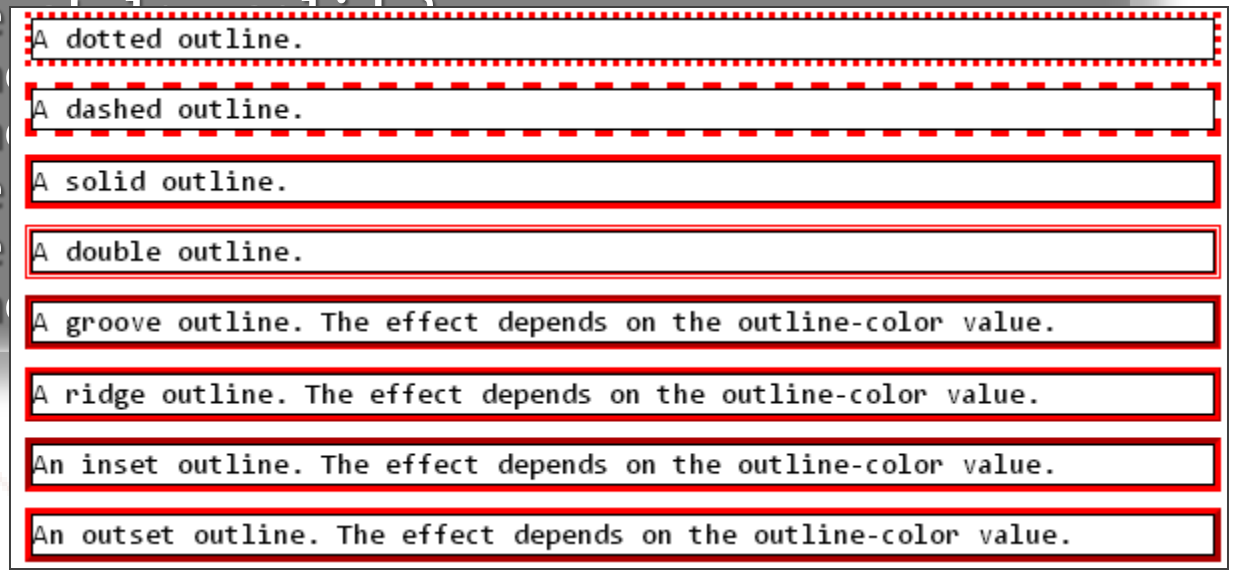
- An outline is a line that is drawn around elements (outside the borders) to make the element "stand out".
- Outlines are very similar to borders, but there are few major differences as well −
  - An outline does not take up space.
  - Outline is always the same on all sides; you cannot specify different values for different sides of an element.

OUTLINE STYLE

```
p {
    border: 1px solid black;
    outline-color: red;
}

p.dotted {outline-style: dotted;}
p.dashed {outline-style: dashed;}
p.solid {outline-style: solid;}
p.double {outline-style: double;}
p.groove {outline-style: groove;}
p.ridge {outline-style: ridge;}
p.inset {outline-style: inset;}
p.outset {outline-style: outset;}
```

OUTLINE STYLE

```
p {
    border: 1px solid black;
    outline-color: red;
}

p.dotted {outline-style: dotted;}
p.dashed {outline-style: dashed;}
p.solid {outline
p.double {outlin
p.groove {outlin
p.ridge {outline
p.inset {outline
p.outset {outlin
```



OUTLINE - SHORTHAND PROPERTY

❖The outline property is a shorthand property for the following individual outline properties:

- ➢outline-width
- ➢outline-style (required)
- ➢outline-color

```
p {
    border: 1px solid black;
    outline: 5px dotted red;
}
```

An outline.

Example Example

DISPLAY PROPERTY

- The display property specifies if/how an element is displayed.
- Every HTML element has a default display value depending on what type of element it is. The default display value for most elements is **block** or **inline**.
- **Block-level Elements**: always starts on a new line and takes up the full width available
- **Inline element** does not start on a new line and only takes up as much width as necessary.
- **display: none;** is commonly used with to hide and show elements without deleting and recreating them.

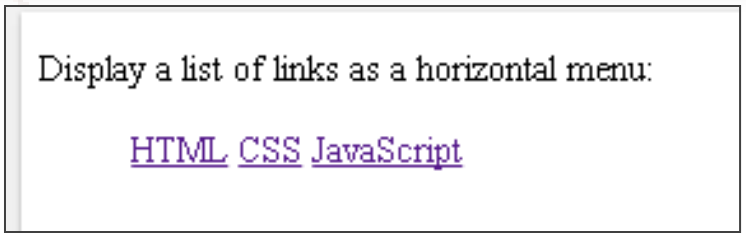
OVERRIDING DEFAULT DISPLAY

❖As mentioned, every element has a default display value. However, you can override this.

❖Changing an inline element to a block element, or vice versa, can be useful for making the page look a specific way, and still follow the web standards.

❖A common example is making inline <li> elements for horizontal menus:

```
li {
    display: inline;
}
```



DISPLAY:NONE OR VISIBILITY:HIDDEN?

- Hiding an element can be done by setting the display property to none. The element will be hidden, and the page will be displayed as if the element is not there.
- **visibility:hidden;** also hides an element.
- However, the element will still take up the same space as before. The element will be hidden, but still affect the layout:

```
h1.hidden {
    display: none;
}
```

```
h1.hidden {
    visibility: hidden;
}
```

DISPLAY:NONE OR VISIBILITY:HIDDEN?

- display: none;


```
<style>
h1.hidden {
    display: none;
}
</style>
</head>
<body>

<h1>This is a visible heading</h1>
<h1 class="hidden">This is a hidden heading</h1>
<p>Notice that the h1 element with display: none;
does not take up any space.</p>
```

DISPLAY:NONE OR VISIBILITY:HIDDEN?

- visibility: hidden;

```
<style>
h1.hidden {
    visibility: hidden;
}
</style>
</head>
<body>

<h1>This is a visible heading</h1>
<h1 class="hidden">This is a hidden heading</h1>
<p>Notice that the hidden heading still takes up
space.</p>
```

Width and max-width

❖A block-level element always takes up the full width available (stretches out to the left and right as far as it can).

❖Setting the width of a block-level element will prevent it from stretching out to the edges of its container.

❖Then, we can set the margins to auto, to horizontally center the element within its container.

❖The element will take up the specified width, and the remaining space will be split equally between the two margins:

This element has a width of 500px, and margin set to auto.

Width and max-width

❖The problem with the above occurs when the browser window is smaller than the width of the element. The browser then adds a horizontal scrollbar to the page.

❖Using max-width instead, in this situation, will improve the browser's handling of small windows. This is important when making a site usable on small devices:

This element has a max-width of 500px, and margin set to auto.

Example

THE POSITION PROPERTY

❖The **position** property specifies the type of positioning method used for an element.

❖There are four different position values:

- ➢static
- ➢relative
- ➢fixed
- ➢Absolute

❖Elements are then positioned using the top, bottom, left, and right properties. However, these properties will not work unless the position property is set first

POSITION: STATIC;

❖HTML elements are positioned **static by default.**

❖Static positioned elements are not affected by the top, bottom, left, and right properties.

❖An element with position: static; is not positioned in any special way; it is always positioned according to the normal flow of the page:

```
div.static {
    position: static;
    border: 3px solid #73AD21;
}
```

POSITION: RELATIVE;

❖An element with position: relative; is positioned relative to its normal position.

❖Setting the top, right, bottom, and left properties of a relatively-positioned element will cause it to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

```
div.relative {
    position: relative;
    left: 30px;
    border: 3px solid #73AD21;
}
```

Example

POSITION: FIXED;

❖An element with position: fixed; is positioned relative to the viewport, which means it always stays in the same place even if the page is scrolled. The top, right, bottom, and left properties are used to position the element.

❖A fixed element does not leave a gap in the page where it would normally have been located.

```
div.fixed {
    position: fixed;
    bottom: 0;
    right: 0;
    width: 300px;
    border: 3px solid #73AD21;
}
```

Example

POSITION: ABSOLUTE;

❖An element with position: absolute; is positioned relative to the nearest positioned ancestor (instead of positioned relative to the viewport, like fixed).

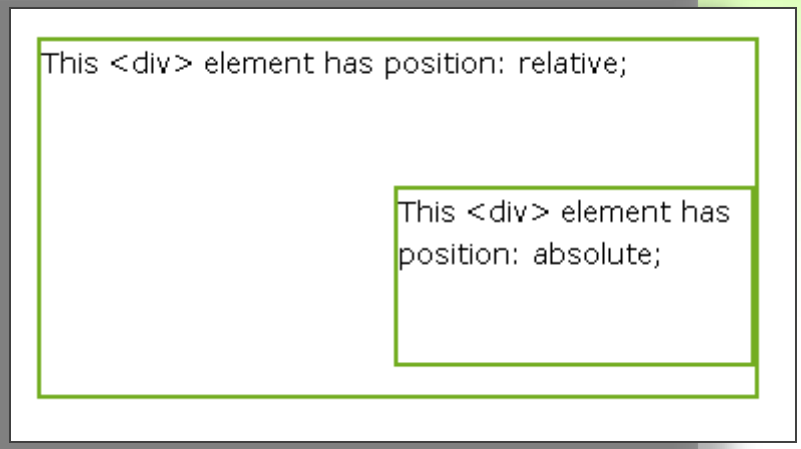
❖However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

```
div.fixed {
    position: fixed;
    bottom: 0;
    right: 0;
    width: 300px;
    border: 3px solid #73AD21;
}
```

Example

POSITION: ABSOLUTE;

```
div.relative {
    position: relative;
    width: 400px;
    height: 200px;
    border: 3px solid #73AD21;
}

div.absolute {
    position: absolute;
    top: 80px;
    right: 0;
    width: 200px;
    height: 100px;
    border: 3px solid #73AD21;
}
```



OVERLAPPING ELEMENTS

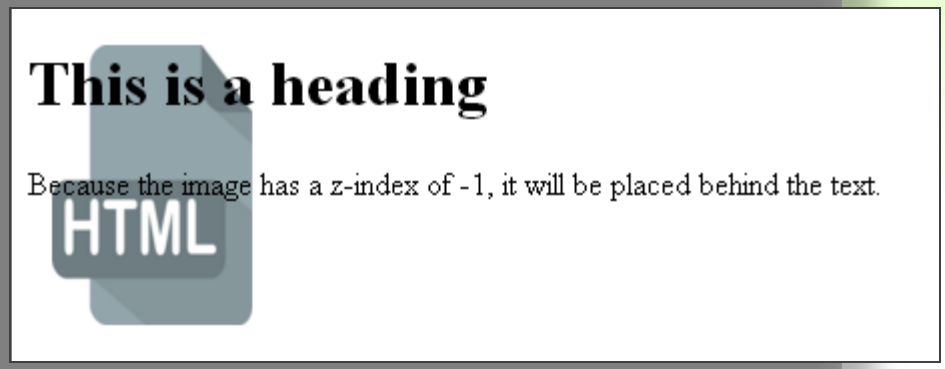
- When elements are positioned, they can overlap other elements.
- The **z-index** property specifies the stack order of an element (which element should be placed in front of, or behind, the others).
- An element can have a positive or negative stack order:

```
img {
    position: absolute;
    left: 0px;
    top: 0px;
    z-index: -1;
}
```

Example

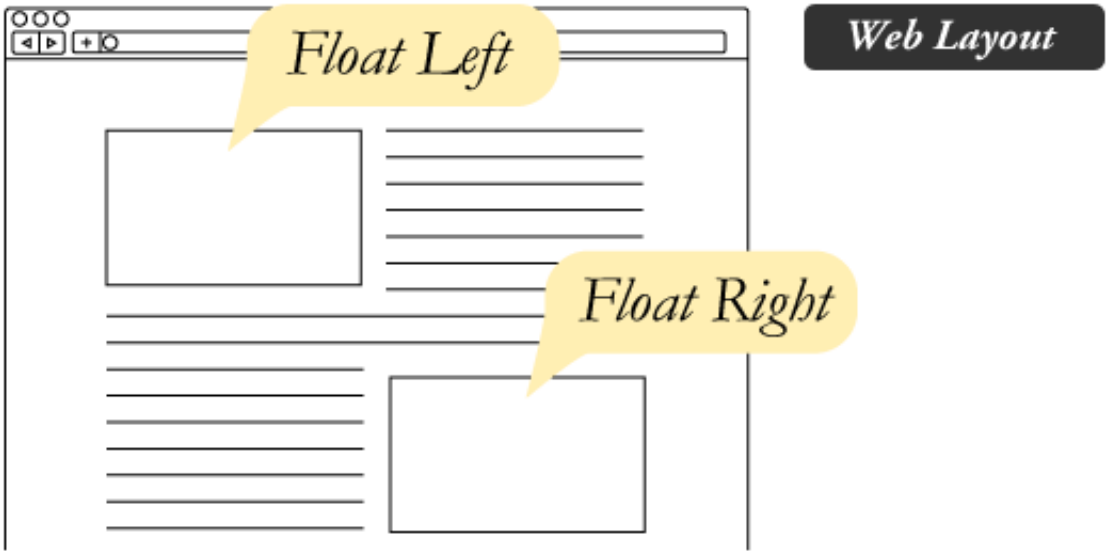
OVERLAPPING ELEMENTS

```
<style>
img {
margin: 20px;
    position: absolute;
    left: 0px;
    top: 0px;
    z-index: -1;
}
</style>
</head>
<body>

<h1>This is a heading</h1>
<img src="html_logo.png" width="100" height="140">
<p>Because the image has a z-index of -1, it will be
placed behind the text.</p>
```



FLOAT AND CLEAR

❖The float CSS property specifies that an element should be taken from the normal flow and placed along the left or right side of its container, where text and inline elements will wrap around it.



FLOAT AND CLEAR

```
<style>
img {
    float: right;
    margin: 0 0 10px 10px;
}
</style>
</head>
<body>

<p>In this example, the image will float to the right
in the paragraph…..</p>

<p><img src="html_logo.png" alt="W3Schools.com"
width="100" height="140">
Lorem ipsum dolor sit amet, consectetur dfggdh...</p>
```

Example

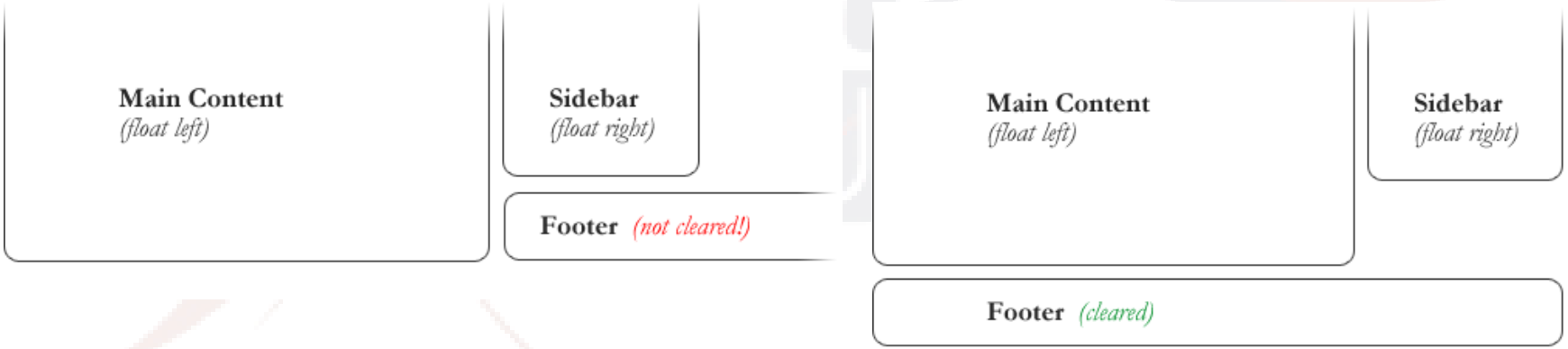
FLOAT AND CLEAR

In this example, the image will float to the right in the paragraph, and the text in the paragraph will wrap around the image.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Phasellus imperdiet, nulla et dictum interdum, nisi lorem egestas odio, vitae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed nunc venenatis imperdiet sed ornare turpis. Donec vitae dui eget tellus gravida venenatis. Integer fringilla congue eros non fermentum. Sed dapibus pulvinar nibh tempor porta. Cras ac leo purus. Mauris quis diam velit.itae scelerisque enim ligula venenatis dolor. Maecenas nisl est, ultrices nec congue eget, auctor vitae massa. Fusce luctus vestibulum augue ut aliquet. Mauris ante ligula, facilisis sed ornare eu, lobortis in odio. Praesent convallis urna a lacus interdum ut hendrerit risus congue. Nunc sagittis dictum nisi, sed ullamcorper ipsum dignissim ac. In at libero sed n

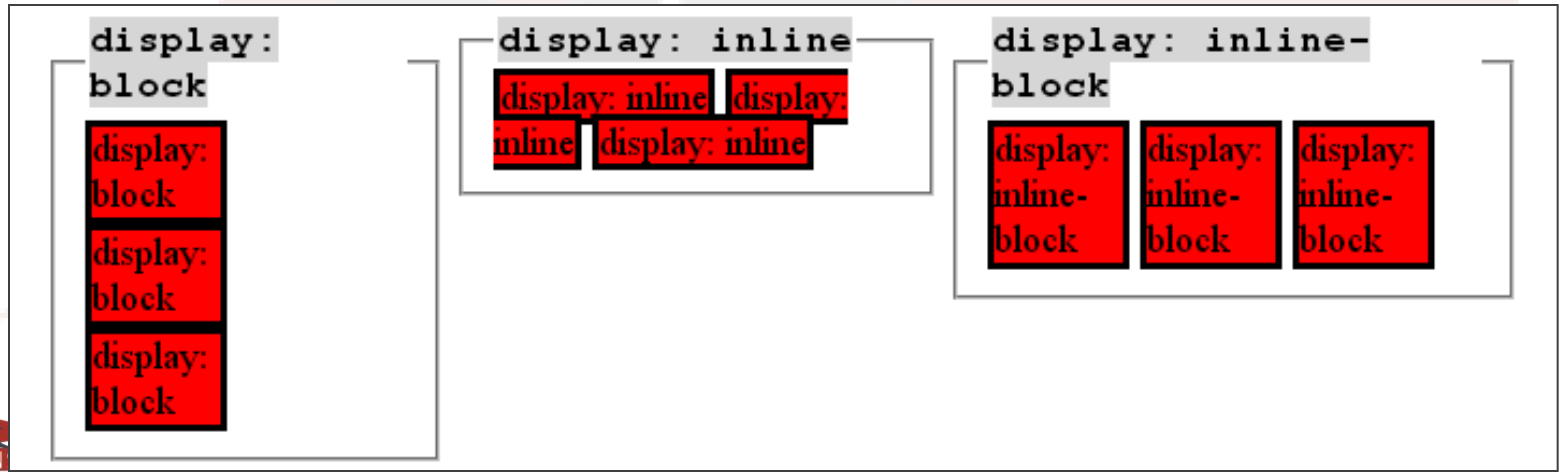
FLOAT AND CLEAR

- The clear property is used to control the behavior of floating elements.
- Elements after a floating element will flow around it. To avoid this, use the clear property.
- The clear property specifies on which sides of an element floating elements are not allowed to float:



INLINE-BLOCK

- inline-block elements are like inline elements but they can have a width and a height.
- An **inline** element has no line break before or after it, and it tolerates HTML elements next to it.
- A **block** element has some whitespace above and below it and does not tolerate any HTML elements next to it.
- An **inline-block** element is placed as an inline element (on the same line as adjacent content), but it behaves as a block element.



INLINE-BLOCK

- Inline elements:
  - respect left & right margins and padding, but not top & bottom
  - cannot have a width and height set
  - allow other elements to sit to their left and right.
- Block elements:
  - respect all of those
  - force a line break after the block element
- Inline-block elements:
  - allow other elements to sit to their left and right
  - respect top & bottom margins and padding
  - respect height and width

CSS COMBINATORS

❖A CSS selector can contain more than one simple selector. Between the simple selectors, we can include a combinator.

❖A combinator is something that explains the relationship between the selectors.

❖There are four different combinators in CSS3:

- ➢descendant selector (space)
- ➢child selector (>)
- ➢adjacent sibling selector (+)
- ➢general sibling selector (~)

DESCENDANT SELECTOR

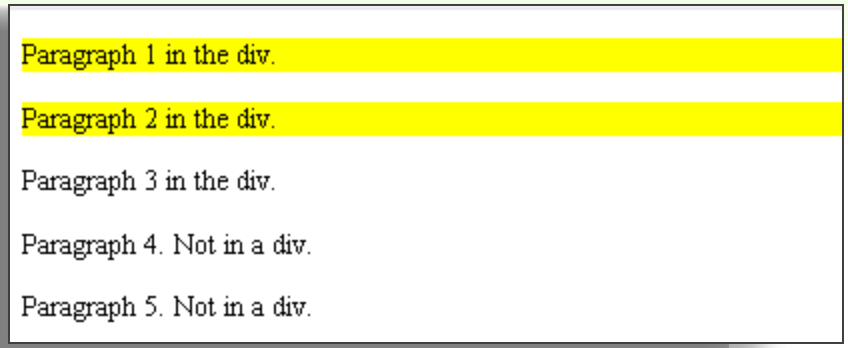
❖The descendant selector matches all elements that are descendants of a specified element.

```
<style>
div p {
    background-color: ye
}
</style>

<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <span><p>Paragraph 3 in the div.</p></span>
</div>

<p>Paragraph 4. Not in a div.</p>
<p>Paragraph 5. Not in a div.</p>
```

CHILD SELECTOR

❖The child selector selects all elements that are the immediate children of a specified element.

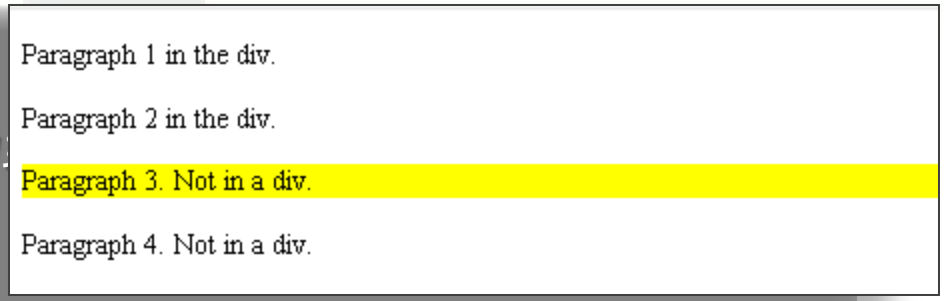
```
<style>
div > p {
    background-color: yellow;
}
</style>

<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
  <span><p>Paragraph 3 in the div.</p></span> <!--
not Child but Descendant -->
</div>
<p>Paragraph 4. Not in a div.</p>
<p>Paragraph 5. Not in a div.</p>
```



ADJACENT SIBLING SELECTOR

- The adjacent sibling selector selects all elements that are the adjacent siblings of a specified element.
- Sibling elements must have the same parent element, and "adjacent" means "immediately following".

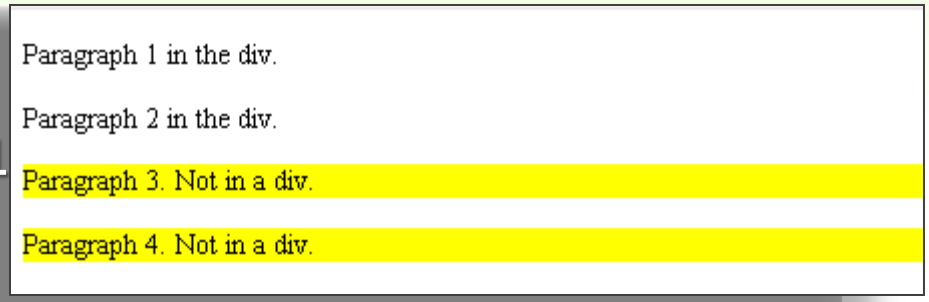
```
<style>
div + p {
    background-color: yellow;
}
</style>

<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
</div>

<p>Paragraph 3. Not in a div.</p>
<p>Paragraph 4. Not in a div.</p>
```



General Sibling Selector

❖The general sibling selector selects all elements that are siblings of a specified element.

```
<style>
div ~ p {
    background-color: yell
}
</style>

<div>
  <p>Paragraph 1 in the div.</p>
  <p>Paragraph 2 in the div.</p>
</div>

<p>Paragraph 3. Not in a div.</p>
<p>Paragraph 4. Not in a div.</p>
```



CSS PSEUDO-CLASSES

- A pseudo-class is used to define a special state of an element.
- For example, it can be used to:
  - Style an element when a user mouses over it
  - Style visited and unvisited links differently
  - Style an element when it gets focus
- Syntax:

```
selector:pseudo-class {
    property:value;
}
```

Anchor Pseudo-classes

- Links can be displayed in different ways:

```
/* unvisited link */
a:link {
    color: #FF0000;
}
/* visited link */
a:visited {
    color: #00FF00;
}
/* mouse over link */
a:hover {
    color: #FF00FF;
}
/* selected link */
a:active {
    color: #0000FF;
}
```

Pseudo-classes and CSS classes

- Pseudo-classes can be combined with CSS classes

```
a.highlight:hover {
    color: #ff0000;
}
```

THE :FIRST-CHILD PSEUDO-CLASS

❖The :first-child pseudo-class matches a specified element that is the first child of another element.

```
p:first-child {
    color: blue;
}
```

❖Match the first <i> element in all <p> elements

```
p i:first-child {
    color: blue;
}
```

❖Match all <i> elements in all first child <p> elements

```
p:first-child i {
    color: blue;
}
```

Example Example Example

PSEUDO-ELEMENTS

- A CSS pseudo-element is used to style specified parts of an element.
- For example, it can be used to:
  - Style the first letter, or line, of an element
  - Insert content before, or after, the content of an element
- Syntax:

```
selector::pseudo-element {
    property:value;
}
```

PSEUDO-ELEMENT

❖The ::first-line pseudo-element is used to add a special style to the first line of a text.

```
p::first-line {
    color: #ff0000;
    font-variant: small-caps;
}
```

❖The ::first-letter pseudo-element is used to add a special style to the first letter of a text.

```
p::first-letter {
    color: #ff0000;
    font-size: xx-large;
}
```

Example

Pseudo-elements and CSS classes

- Pseudo-elements can be combined with CSS classes:

```
p.intro::first-letter {
    color: #ff0000;
    font-size:200%;
}
```

- Several pseudo-elements can also be combined.

```
p::first-letter {
    color: #ff0000;
    font-size: xx-large;
}

p::first-line {
    color: #0000ff;
    font-variant: small-caps;
}
```

THE ::BEFORE PSEUDO-ELEMENT

❖The ::before pseudo-element can be used to insert some content before the content of an element.

```
h1::before {
    content: url(smiley.gif);
}
```

❖The ::after pseudo-element can be used to insert some content after the content of an element.

```
h1::after {
    content: url(smiley.gif);
}
```

❖The ::selection pseudo-element matches the portion of an element that is selected by a user.

```
::selection {
    color: red;
    background: yellow;
}
```

Example Example

CSS ATTRIBUTE SELECTORS

❖It is possible to style HTML elements that have specific attributes or attribute values.

❖CSS **[attribute] Selector**: The [attribute] selector is used to select elements with a specified attribute.

```
a[target] {
    background-color: yellow;
}
```

❖The **[attribute="value"]** selector is used to select elements with a specified attribute and value.

```
a[target="_blank"] {
    background-color: yellow;
}
```

CSS ATTRIBUTE SELECTORS

❖The **[attribute~="value"]** selector is used to select elements with an attribute value containing a specified word.

```
[title~="flower"] {
    border: 5px solid yellow;
}
```

❖The example above will match elements with title="flower", title="summer flower", and title="flower new", but not title="my-flower" or title="flowers".

CSS ATTRIBUTE SELECTORS

❖The **[attribute|="value"]** selector is used to select elements with the specified attribute starting with the specified value.

```
[title~="flower"] {
    border: 5px solid yellow;
}
```

❖The value has to be a whole word, either alone, like class="top", or followed by a hyphen(-), like class="top-text"!

CSS ATTRIBUTE SELECTORS

❖The **[attribute^="value"]** selector is used to select elements whose attribute value begins with a specified value.

```
[class^="top"] {
    background: yellow;
}
```

❖The value does not have to be a whole word!

CSS ATTRIBUTE SELECTORS

❖The **[attribute$="value"]** selector is used to select elements whose attribute value ends with a specified value.

```
[class$="top"] {
    background: yellow;
}
```

❖The **[attribute*="value"]** selector is used to select elements whose attribute value contains a specified value.

```
[class*="te"] {
    background: yellow;
}
```

❖The value does not have to be a whole word!

Q/A

- The CSS links properties are
  a) :link, :visited, :hover, :active
  b) :link, :visit, :hover, :active
  c) :link, :visited, :over, :active
  d) :link, :visited, :hover, :active, :inactive
- The ..... specifies whether a border should be solid, dashed line, doted line, double line, groove etc.
  a) border-layout
  b) border-decoration
  c) border-style
  d) border-weight
- Which of the following are the background properties in CSS?
  a) background-color
  b) background-image
  c) background-repeat
  d) background-position
  e) background

Q/A

❖A CSS style rule is made up of three parts which are .

i) Selector ii) Property iii) Value iv) Attribute

a) i, ii and iii only
b) ii, iii and iv only
c) i, ii and iv only
d) All i, ii, iii and iv

❖The valid examples of ID selectors is

a) h1 #black{color: #000000;}
b) #black{color:#000000;}
c) #black h2{color:#000000;}
d) All of the above

❖State whether True or False.

❖Any inline style sheet takes highest priority.

❖Any rule defined in <style> .....</style> tag will override rules defined in any external style sheet

Q/A

❖Elements in CSS cannot be positioned unless the positioning property is set first

a) True
b) False

❖ _____ property can be used to Increase or decrease the space between words

a) space
b) word-spacing
c) word-space
d) Both b and c

❖_____ property specifies an image to use as the background of an element

a) backg-img
b) backg-image
c) background-img
d) background-image

Thank you!

Contact Us

info@4kitsolutions.com