WORKING WITH CSS

INTRODUCTION

CSS stands for Cascading Style Sheets

CSS describes how HTML elements are to be displayed on screen, paper, or in other media

CSS saves a lot of work. It can control the layout of multiple web pages all at once

External stylesheets are stored in CSS files

Why Use CSS?

CSS is used to define styles for your web pages, including the design, layout and variations in display for different devices and screen sizes.

Syntax

Selector {declaration}

Selector {property:value;}

EX. H1 {color:Red;
Font:TimesNewRoma;}

1. In this example all <p> elements will be center-aligned, with a red text color:

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

The id Selector

The id selector uses the id attribute of an HTML element to select a specific element.

The id of an element should be unique within a page, so the id selector is used to select one 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.

Example

In this example all HTML elements with class="center" will be red and center-aligned:

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

Grouping Selectors

If you have elements with the same style definitions, like this:

```
h1 {
  text-align: center;
  color: blue;
}
h2 {
  text-align: center;
  color: red;}
```

To group selectors, separate each selector with a comma.

```
h1, h2, p {
  text-align: center;
  color: red;
}
```

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

1. External style sheet
2. Internal style sheet
3. Inline style

External Style Sheet

With an external style sheet, you can change the look of an entire website by changing just one file. Each page must include a reference to the external style sheet file inside the <link> element.

Example

External styles are defined within the <link> element, inside the <head> section of an HTML page:

```
<head>
<link rel="stylesheet" type="text/css" href="mystyle.css">
</head>
p {
  text-align: center;
  color: red;
}
```

NOTE: An external style sheet can be written in any text editor. The file should not contain any html tags. The style sheet file must be saved with a .css extension.

Here is how the "mystyle.css" file looks like:

"mystyle.css"

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

Internal Style Sheet

An internal style sheet may be used if one single page has a unique style.

Example

Internal styles are defined within the <style> element, inside the <head> section of an HTML page:

```
<head>
<style>
body {
  background-color: linen;
}
h1 {
  color: maroon;
  margin-left: 40px;
}
</style>
</head>
```

Inline Styles

An inline style may be used to apply a unique style for a single element.

To use inline styles, add the style attribute to the relevant element. The style attribute can contain any CSS property.

Example

Inline styles are defined within the "style" attribute of the relevant element:

```
<h1 style="color:blue;margin-left:30px;">This is a heading</h1>
```

Cascading Order

What style will be used when there is more than one style specified for an HTML element?

All the styles in a page 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 has the highest priority, and will override external and internal styles and browser defaults.

Background Color

The background-color property specifies the background color of an element.

Example

The background color of a page is set like this:

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

With CSS, a color is most often specified by:

a valid color name - like "red"

a HEX value - like "#ff0000"

an RGB value - like "rgb(255,0,0)"

Background Image

The background-image property specifies an image to use as the background of an element.

By default, the image is repeated so it covers the entire element.

Example

The background image for a page can be set like this:

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

CSS Border Properties

The CSS border properties allow you to specify the style, width, and color of an element's border.

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

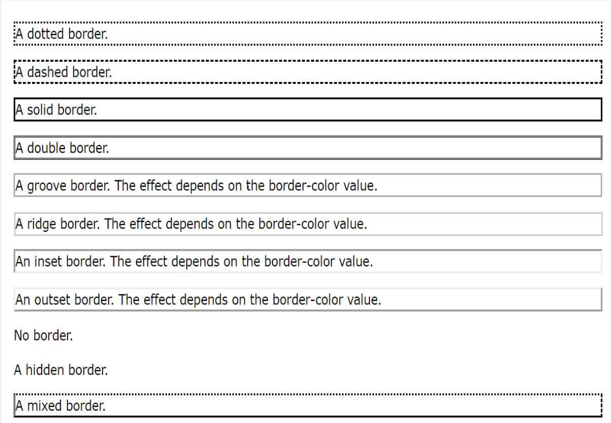
none - Defines no border

hidden - Defines a hidden border

Example

Demonstration of the different border styles:

```
p.dotted {border-style: dotted;}

p.dashed {border-style: dashed;}

p.solid {border-style: solid;}

p.double {border-style: double;}

p.groove {border-style: groove;}

p.ridge {border-style: ridge;}

p.inset {border-style: inset;}

p.outset {border-style: outset;}

p.none {border-style: none;}

p.hidden {border-style: hidden;}

p.mix {border-style: dotted dashed solid double;}
```



A dotted border.

A dashed border.

A solid border.

A double border.

A groove border. The effect depends on the border-color value.

A ridge border. The effect depends on the border-color value.

An inset border. The effect depends on the border-color value.

An outset border. The effect depends on the border-color value.

No border.

A hidden border.

A mixed border.

Border Width

The border-width property specifies the width of the four borders.

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;

}
```

Border Color

The border-color property is used to set the color of the four borders.

The color can be set by:

name - specify a color name, like "red"

Hex - specify a hex value, like "#ff0000"

RGB - specify a RGB value, like "rgb(255,0,0)"

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

If border-color is not set, it inherits the color of the element.

```
p.one {
  border-style: solid;
  border-color: red;
}
```

CSS Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

With CSS, you have full control over the padding. There are properties for setting the padding for each side of an element (top, right, bottom, and left).

Padding - Individual Sides

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

Note: Negative values are not allowed.

Example

Set different padding for all four sides of a <div> element:

```
div {
  padding-top: 50px;
  padding-right: 30px;
  padding-bottom: 50px;
  padding-left: 80px;
}
```

Example

Use the padding shorthand property with four values:

```
div {
  padding: 25px 50px 75px 100px;}
```

NOTE: top padding is 25px

right padding is 50px

bottom padding is 75px

left padding is 100px

Setting 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), or be specified in length values, like px, cm, etc., or in percent (%) of the containing block.

Example:Set the height and width of a <div> element:

```
div {
  height: 200px;
  width: 50%;
  background-color: powderblue; }
```

Setting max-width

The max-width property is used to set the maximum width of an element.

The max-width can be specified in length values, like px, cm, etc., or in percent (%) of the containing block, or set to none (this is default. Means that there is no maximum width).

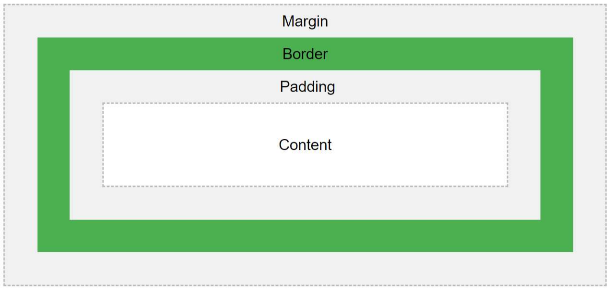
The problem with the <div> above occurs when the browser window is smaller than the width of the element (500px). The browser then adds a horizontal scrollbar to the page. Using max-width instead, in this situation, will improve the browser's handling of small windows.

```
div {
  max-width: 500px;
  height: 100px;
  background-color: powderblue;
}
```

The 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. The image below illustrates the box model:



Explanation of the different parts:

Content - The content of the box, where text and images appear

Padding - Clears an area around the content. The padding is transparent

Border - A border that goes around the padding and content

Margin - Clears an area outside the border. The margin is transparent

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

Example

Demonstration of the box model:

```
div {
  width: 300px;
  border: 15px solid green;
  padding: 50px;
  margin: 20px;
}
```

Text Transformation

The text-transform property is used to specify uppercase and lowercase letters in a text.

It can be used to turn everything into uppercase or lowercase letters, or capitalize the first letter of each word:

Example

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

Letter Spacing

The letter-spacing property is used to specify the space between the characters in a text.

The following example demonstrates how to increase or decrease the space between characters:

Example

```
h1 {
  letter-spacing: 3px; }
h2 {
  letter-spacing: -3px; }
```

Word Spacing

The word-spacing property is used to specify the space between the words in a text.

The following example demonstrates how to increase or decrease the space between words:

Example

```
h1 {
  word-spacing: 10px;}
h2 {
  word-spacing: -5px; }
```

Text Shadow

The text-shadow property adds shadow to text. The following example specifies the position of the horizontal shadow (3px), the position of the vertical shadow (2px) and the color of the shadow (red):

Example

```
h1 {
  text-shadow: 3px 2px red;}
```

Properties of Text

Property	Description
color	Sets the color of text
direction	Specifies the text direction/writing direction
letter-spacing	Increases or decreases the space between characters in a text
line-height	Sets the line height
text-align	Specifies the horizontal alignment of text
text-decoration	Specifies the decoration added to text
text-indent	Specifies the indentation of the first line in a text-block
text-shadow	Specifies the shadow effect added to text
text-transform	Controls the capitalization of text
text-overflow	Specifies how overflowed content that is not displayed should be signaled to the user

unicode-bidi	Used together with the direction property to set or return whether the text should be overridden to support multiple languages in the same document
vertical-align	Sets the vertical alignment of an element
white-space	Specifies how white-space inside an element is handled
word-spacing	Increases or decreases the space between words in a text

Font Properties

Property	Description
font	Sets all the font properties in one declaration
font-family	Specifies the font family for text
font-size	Specifies the font size of text
font-style	Specifies the font style for text
font-variant	Specifies whether or not a text should be displayed in a small-caps font
font-weight	Specifies the weight of a font

CSS Icons

The simplest way to add an icon to your HTML page, is with an icon library, such as Font Awesome.

Add the name of the specified icon class to any inline HTML element (like <i> or <span>).

All the icons in the icon libraries below, are scalable vectors that can be customized with CSS (size, color, shadow, etc.)

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://use.fontawesome.com/releases/v5.7.0/css/all.css" integrity="sha384-lZN37f5QGtY3VHgisS14W3ExzMWZxybE1SJSEsQp9S+oqd12jhcu+A56Ebc1zFSJ" crossorigin="anonymous">
</head>
<body>
<i class="fas fa-cloud"></i>
<i class="fas fa-heart"></i>
<i class="fas fa-car"></i>
<i class="fas fa-file"></i>
<i class="fas fa-bars"></i>


</body>
</html>
```

CSS Links

Styling Links

Links can be styled with any CSS property (e.g. color, font-family, background, etc.).

Example

```
a {
  color: hotpink; }
```

links can be styled differently depending on what state they are in.

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 mouses over it

a:active - a link the moment it is clicked

Example

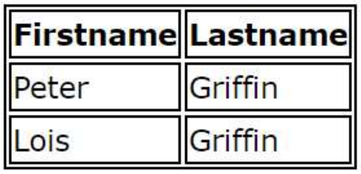
```
/* unvisited link */
a:link {
  color: red;
}
/* visited link */
a:visited {
  color: green;
}
/* mouse over link */
a:hover {
  color: hotpink;
}
/* selected link */
a:active {
  color: blue;
}
```

CSS Tables

Table Borders

To specify table borders in CSS, use the border property.

The example below specifies a black border for <table>, <th>, and <td> elements:



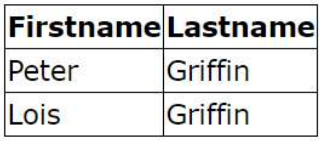
Firstname	Lastname
Peter	Griffin
Lois	Griffin

Example

```
table, th, td {
  border: 1px solid black; }
```

Collapse Table Borders

The border-collapse property sets whether the table borders should be collapsed into a single border:



Firstname	Lastname
Peter	Griffin
Lois	Griffin

Example

```
table {
  border-collapse: collapse; }
table, th, td {
  border: 1px solid black; }
```

CSS Table Properties

Property	Description

border	Sets all the border properties in one declaration
border-collapse	Specifies whether or not table borders should be collapsed
border-spacing	Specifies the distance between the borders of adjacent cells
caption-side	Specifies the placement of a table caption
empty-cells	Specifies whether or not to display borders and background on empty cells in a table
table-layout	Sets the layout algorithm to be used for a table

CSS Layout – Overflow

The overflow property specifies whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified area. The overflow property has the following values:

visible - Default. The overflow is not clipped. The content renders outside the element's box

hidden - The overflow is clipped, and the rest of the content will be invisible

scroll - The overflow is clipped, and a scrollbar is added to see the rest of the content

auto - Similar to scroll, but it adds scrollbars only when necessary.

Example

```
div {
  width: 200px;
  height: 50px;
  background-color: #eee;
  overflow: visible; }
```

The display: inline-block Value

Compared to display: inline, the major difference is that display: inline-block allows to set a width and height on the element. Also, with display: inline-block, the top and bottom margins/paddings are respected, but with display: inline they are not.

Compared to display: block, the major difference is that display: inline-block does not add a line-break after the element, so the element can sit next to other elements.

Example

```
span.a {
  display: inline; /* the default for span */
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;
}
span.b {
  display: inline-block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;  }
span.c {
  display: block;
  width: 100px;
  height: 100px;
  padding: 5px;
  border: 1px solid blue;
  background-color: yellow;  }
```

The following example shows the different behavior of display: inline, display: inline-block and display: block.

Create Navigation Links

One common use for display: inline-block is to display list items horizontally instead of vertically. The following example creates horizontal navigation links:

Example

```
.nav {
  background-color: yellow;
  list-style-type: none;
  text-align: center;
  padding: 0;
  margin: 0;}
.nav li {
  display: inline-block;
  font-size: 20px;
  padding: 20px; }
```

CSS Combinators

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

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

There are four different combinators in CSS:

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

1.Descendant Selector

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

The following example selects all <p> elements inside <div> elements:

Example

```
div p {
  background-color: yellow; }
```

2.Child Selector

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

The following example selects all <p> elements that are immediate children of a <div> element:

Example

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

3.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".

The following example selects all <p> elements that are placed immediately after <div> elements:

Example

```
div + p {
  background-color: yellow; }
```

4.General Sibling Selector

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

The following example selects all <p> elements that are siblings of <div> elements:

Example

```
div ~ p {
  background-color: yellow; }
```

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

What are 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.

CSS Opacity / Transparency

The opacity property specifies the opacity/transparency of an element.

The opacity property can take a value from 0.0 - 1.0. The lower value, the more transparent:

```
img {
  opacity: 0.5;
  filter: alpha(opacity=50); }
```