



CSS BOOK 2

CSS is the key presentational technology that is used in website design.

Created By Modcom Institute of Technology

Table of Contents

CSS BOOK 2.....	1
Introduction to CSS.....	5
What You Can Do with CSS.....	5
Advantages of Using CSS.....	6
Getting Started with CSS.....	7
Including CSS in HTML Documents.....	7
Inline Styles.....	7
Example Inline CSS.....	8
Embedded/Internal Style Sheets.....	8
Example of Embedded/Internal CSS Code.....	9
External Style Sheets.....	9
Linking External Style Sheets.....	10
Example of External Style sheet.....	10
Importing External Style Sheets.....	11
Inline vs Internal/Embedded vs External.....	12
Understanding CSS Syntax.....	12
Writing Comments in CSS.....	13
CSS Selectors.....	14
Element/Tag Selectors.....	14
Id Selectors.....	14
Class Selectors.....	15
Styling using Internal/Embedded CSS.....	15
Setting Color and Background Property.....	15
For HTML color codes.....	16
Styling Fonts with CSS.....	16
Font Family.....	16
Font Style.....	16
Font Size.....	17
Font Weight.....	17
Font Variant.....	17
Class Practice.....	17
CSS Text.....	19
Formatting Text with CSS.....	19
Text Color.....	19
Example.....	19
Text Alignment.....	20
Example.....	20
Text Decoration.....	20
Example.....	21
Removing the Default Underline from Links.....	21
Example.....	21
Text Transformation.....	21
Example.....	22
Text Indentation.....	22
Example.....	22

Letter Spacing.....	23
Example.....	23
Word Spacing.....	23
Example.....	23
Line Height.....	24
Example.....	24
Example.....	24
CSS Links.....	25
Styling Links with CSS.....	25
Modifying Standard Link Styles.....	26
Making Text Links Look Like Buttons.....	26
Example.....	27
CSS Lists.....	27
Types of HTML Lists.....	27
Changing the Marker Type of Lists.....	28
Example.....	28
Using Images as List Markers.....	28
Example.....	28
CSS Box Model.....	29
What is Box Model?.....	29
CSS Dimension.....	30
Setting Element Dimensions.....	30
Setting the Width and Height.....	30
Example.....	30
CSS Padding.....	30
CSS Padding Properties.....	30
Define Paddings for Individual Sides.....	31
Example.....	31
The Padding Shorthand Property.....	31
Example.....	31
CSS Margin.....	32
CSS Margin Properties.....	32
Setting Margins for Individual Sides.....	32
Example.....	32
The Margin Shorthand Property.....	33
Example.....	33
CSS Border.....	33
CSS Border Properties.....	33
Understanding the Different Border Styles.....	34
Example.....	35
Class and ID Selectors.....	35
CSS Class vs ID Selectors.....	36
The ID Selector is Unique.....	36
The Class Selector is Not Unique.....	37
You Can Use Both ID and CSS Class Selectors.....	38
CSS3 Animations.....	38
Creating CSS3 Animations.....	38
CSS3 Animation Properties.....	39
Here is a simple example to Animate a Moving Car.....	40
CSS3 Transitions.....	41

Understanding CSS3 Transitions.....	41
Here is an example on Transition.....	41
Revision Questions.....	43
What is the difference between the usage of an ID and a Class?.....	43
What are the ways to assign a certain colour to an element in CSS?.....	43
What are the various font-related attributes in CSS?.....	43
What are the various text-related attributes in CSS?.....	43
Explain responsive web design.....	43
What are the different types of Selectors in CSS?.....	43
What is the difference between inline, inline-block, and block?.....	43
Explain the CSS Box Model and its different elements.....	43
References.....	44

Introduction to CSS

CSS stands for Cascading Style Sheets. CSS is a standard style sheet language used for describing the presentation (i.e. the layout and formatting) of the web pages.

Prior to CSS, nearly all of the presentational attributes of HTML documents were contained within the HTML markup (specifically inside the HTML tags); all the font colors, background styles, element alignments, borders and sizes had to be explicitly described within the HTML.

As a result, development of the large websites became a long and expensive process, since the style information were repeatedly added to every single page of the website.

To solve this problem CSS was introduced in 1996 by the World Wide Web Consortium (W3C), which also maintains its standard. CSS was designed to enable the separation of presentation and content. Now web designers can move the formatting information of the web pages to a separate style sheet which results in considerably simpler HTML markup, and better maintainability.

CSS3 is the latest version of the CSS specification. CSS3 adds several new styling features and improvements to enhance the web presentation capabilities.

What You Can Do with CSS

There are lot more things you can do with CSS.

- You can easily apply same style rules on multiple elements.
- You can control the presentation of multiple pages of a website with a single style sheet.
- You can present the same page differently on different devices.
- You can style dynamic states of elements such as hover, focus, etc. that isn't possible otherwise.

- You can change the position of an element on a web page without changing the markup.
- You can alter the display of existing HTML elements.
- You can transform elements like scale, rotate, skew, etc. in 2D or 3D space.
- You can create animations and transitions effects without using any JavaScript.
- You can create print friendly version of your web pages.

Advantages of Using CSS

The biggest advantage of CSS is that it allows the separation of style and layout from the content of the document. Here are some more advantages, why one should start using CSS?

- CSS Save Lots of Time — CSS gives lots of flexibility to set the style properties of an element. You can write CSS once; and then the same code can be applied to the groups of HTML elements, and can also be reused in multiple HTML pages.
- Easy Maintenance — CSS provides an easy means to update the formatting of the documents, and to maintain the consistency across multiple documents. Because the content of the entire set of web pages can be easily controlled using one or more style sheets.
- Pages Load Faster — CSS enables multiple pages to share the formatting information, which reduces complexity and repetition in the structural contents of the documents. It significantly reduces the file transfer size, which results in a faster page loading.
- Superior Styles to HTML — CSS has much wider presentation capabilities than HTML and provide much better control over the layout of your web pages. So you can give far better look to your web pages in comparison to the HTML presentational elements and attributes.

- **Multiple Device Compatibility** — CSS also allows web pages to be optimized for more than one type of device or media. Using CSS the same HTML document can be presented in different viewing styles for different rendering devices such as desktop, cell phones, etc.

Getting Started with CSS

In this lesson you'll learn how easy it is to add style and formatting information to the web pages using CSS. But, before we begin, make sure that you have some working knowledge of HTML.

If you're just starting out in the world of web development, Please check out **HTML5 Book1**

Including CSS in HTML Documents

CSS can either be attached as a separate document or embedded in the HTML document itself. There are three methods of including CSS in an HTML document:

- **Inline styles** — Using the `style` attribute in the HTML start tag.
- **Embedded styles** — Using the `<style>` element in the head section of a document.
- **External style sheets** — Using the `<link>` element, pointing to an external CSS file.

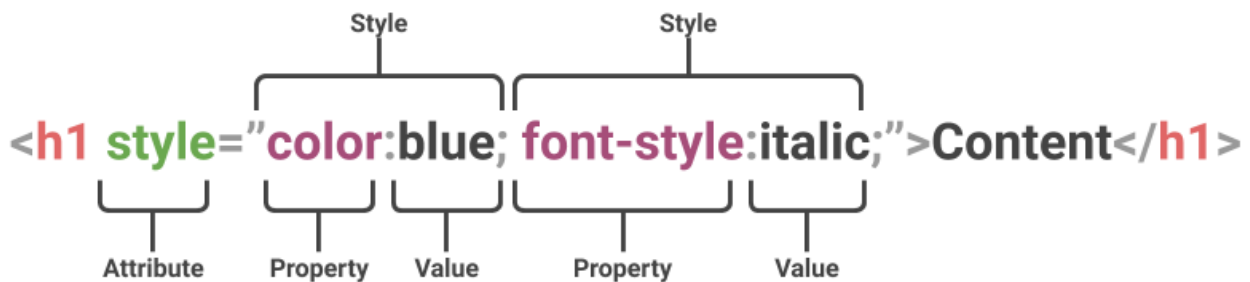
Inline Styles

Inline styles are used to apply the unique style rules to an element by putting the CSS rules directly into the start tag. It can be attached to an element using the `style` attribute.

The `style` attribute includes a series of CSS property and value pairs.

Each `"property: value"` pair is separated by a semicolon (;), just as you would

write into an embedded or external style sheets. But it needs to be all in one line i.e. no line break after the semicolon, as shown here:



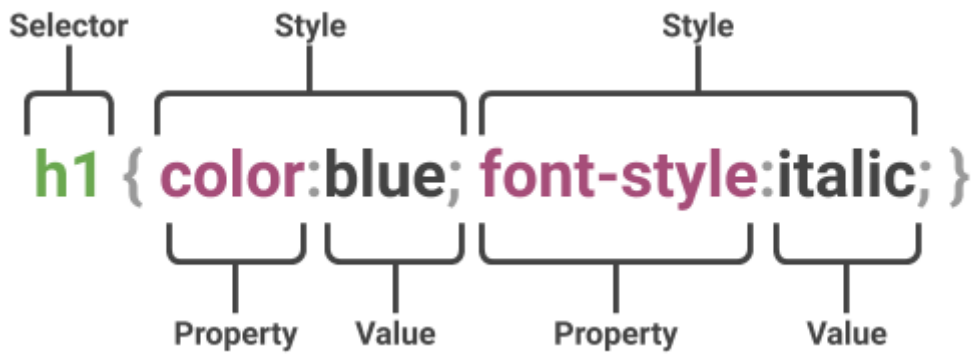
Example Inline CSS

```
<h1 style="color:red; font-size:30px;">This is a heading</h1> <p  
style="color:green; font-size:22px;">This is a paragraph.</p>
```

Embedded/Internal Style Sheets

Embedded or internal style sheets only affect the document they are embedded in.

Embedded style sheets are defined in the `<head>` section of an HTML document using the `<style>` element. You can define any number of `<style>` elements in an HTML document but they must appear between the `<head>` and `</head>` tags. Let's take a look at an example:



Example of Embedded/Internal CSS Code

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>My HTML Document</title>
  <style>
    body { background-color: YellowGreen; }
    p { color: blue; }
  </style>
</head>
<body>
  <h1>This is a heading</h1>
  <p>This is a paragraph of text.</p>
</body>
</html>
```

External Style Sheets

An external style sheet is ideal when the style is applied to many pages of the website.

An external style sheet holds all the style rules in a separate document that you can link from any HTML file on your site. External style sheets are the most

flexible because with an external style sheet, you can change the look of an entire website by changing just one file.

You can attach external style sheets in two ways — linking and importing.

Linking External Style Sheets

Before linking, Inside your working folder create a subfolder named **css**, we then need to create a style sheet. Let's open your favorite code editor(VS Code) and create a new file. Now type the following CSS code inside this file and save it inside your **css** subfolder as "**style.css**".

Example of External Style sheet

```
body {  
    background: lightyellow;  
    color: blue;  
}  
h1 {  
    color: orange;  
}
```

An external style sheet can be linked to an HTML document using the `<link>` tag. The `<link>` tag goes inside the `<head>` section, as you can see in the following example:

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
    <title>My HTML Document</title>  
    <link rel="stylesheet" href="css/style.css">  
</head>  
<body>
```

```
<h1>This is a heading</h1>
<p>This is a paragraph of text.</p>
</body>
</html>
```

Importing External Style Sheets

The `@import` rule is another way of loading an external style sheet.

The `@import` statement instructs the browser to load an external style sheet and use its styles.

```
<!DOCTYPE html>
<html lang="en">
<head>
  <title>My HTML Document</title>
  <style>
    @import url("css/style.css");
  </style>
</head>
<body>
  <h1>This is a heading</h1>
  <p>This is a paragraph of text.</p>
</body>
</html>
```

Inline vs Internal/Embedded vs External

Inline CSS

```
<p style="color: blue;">This is a paragraph.</p>
```

Internal CSS

```
<head>  
  <style type = text/css>  
    body {background-color: blue;}  
    p { color: yellow;}  
  </style>  
</head>
```

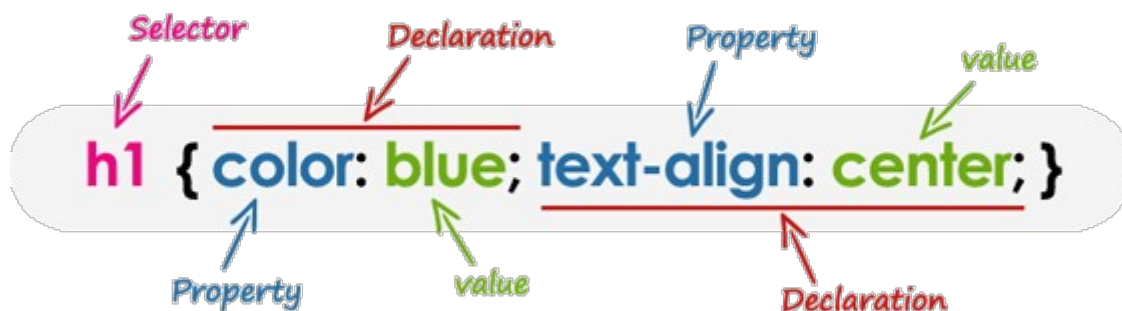
External CSS

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

Understanding CSS Syntax

A CSS stylesheet consists of a set of rules that are interpreted by the web browser and then applied to the corresponding elements such as paragraphs, headings, etc. in the document.

A CSS rule have two main parts, a selector and one or more declarations:



The selector specifies which element or elements in the HTML page the CSS rule applies to.

Whereas, the declarations within the block determines how the elements are formatted on a webpage. Each declaration consists of a property and a value separated by a colon (:) and ending with a semicolon (;), and the declaration groups are surrounded by curly braces {}.

The property is the style attribute you want to change; they could be font, color, background, etc. Each property has a value, for example color property can have value either blue or #0000FF etc.

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

To make the CSS more readable, you can put one declaration on each line, like this:

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

Writing Comments in CSS

Comments are usually added with the purpose of making the source code easier to understand. It may help other developer (or you in the future when

you edit the source code) to understand what you were trying to do with the CSS. Comments are significant to programmers but ignored by browsers.

A CSS comment begins with `/*`, and ends with `*/`, as shown in the example below:

```
/* This is a CSS comment */
```

CSS Selectors

In CSS there are 3 types of selectors, Element/tag Selectors , Class selectors and Id selectors, Here the explanations and examples

Element/Tag Selectors

An element type selector matches all instance of the element in the document with the corresponding element type name. Let's try out an example to see how it actually works:

```
p { color: blue; }
```

Id Selectors

The id selector is used to define style rules for a single or unique element.

The id selector is defined with a hash sign (`#`) immediately followed by the id value.

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

Class Selectors

The class selectors can be used to select any HTML element that has a `class` attribute. All the elements having that class will be formatted according to the defined rule.

The class selector is defined with a period sign (.) immediately followed by the class value.

```
.blue { color: blue; }
```

Styling using Internal/Embedded CSS

Setting Color and Background Property

The `color` property defines the text color (foreground color in general) of an element.

The `background-color` property defines the background color (background color in general) of an element.

For instance, the `color` property and `background-color` property specified in the `h2` selector defines the default text color for the whole page. Let's try out the following example to see how it works:

```
h2 {  
    color: #ff5722;  
    background-color: #f0e68c;  
}
```

For HTML color codes

- (a HEX value - like "#ff0000") please check below link

<https://m2.material.io/design/color/the-color-system.html#tools-for-picking-colors>

Styling Fonts with CSS

Choosing the right font and style is very crucial for the readability of text on a page.

CSS provide several properties for styling the font of the text, including changing their face, controlling their size and boldness, managing variant, and so on.

The font properties are: `font-family`, `font-style`, `font-weight`, `font-size`, and `font-variant`.

Font Family

The `font-family` property is used to specify the font to be used to render the text.

```
h2{  
font-family: Arial, Helvetica, sans-serif;  
}
```

Font Style

The `font-style` property is used to set the font face style for the text content of an element.

The font style can be `normal`, `italic` or `oblique`. The default value is `normal`.

```
h2{  
font-style: normal;  
}
```


Font Size

The `font-size` property is used to set the size of font for the text content of an element.

There are several ways to specify the font size values e.g. with keywords, percentage, pixels, ems, etc.

```
h1 {  
  font-size: 24px;  
}
```

Font Weight

The `font-weight` property specifies the weight or boldness of the font.

This property can take one of the following values: normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900 and `inherit`.

```
p {  
  font-weight: bold;  
}
```

Font Variant

The `font-variant` property allows the text to be displayed in a special small-caps variation.

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

Class Practice

```
<!DOCTYPE html>  
<html lang="en">  
<head>  
<meta charset="UTF-8">
```

```

<meta http-equiv="X-UA-Compatible" content="IE=edge">
<meta name="viewport" content="width=device-width, initial-scale=1.0">
<title>Document</title>
<style>
h2 {
    color: burlywood;
    background-color: black;
    font-family: Arial, Helvetica, sans-serif;
    font-style: normal;
    font-size: 40px;
    font-weight: 800;
    font-variant: small-caps;
}

h3{
    /* apply color, background color and font size to h3 */
    color: white;
    background-color: black;
    font-size: 40px;
}

p {
    color: brown;
    background-color: blanchedalmond;
}

</style>
</head>
<body>
<h2>Classic Cakes</h2>
<p>Lorem ipsum dolor sit amet consectetur adipisicing elit. Incidunt magnam
alias molestiae deleniti, aspe.</p>

<h3>Birthday Cakes</h3>
<p>Lorem ipsum dolor sit amet consectetur, adipisicing elit. cum, repellat
minima id necessitatibus eligendi dolor!</p>

<h3>Biscuit Cakes</h3>
<p>Lorem ipsum dolor sit amet consectetur emporibus voluptatem
mollitia.</p>

<p>Lorem ipsum dolor sit, amet consete itaque, fugit beatae. Corporis vero
eum excepturi fuga ab facilis sapientea!</p>

<marquee behavior="" direction="">Very Nice Flavor!</marquee>
<b>Get Our Products at 20% Discount</b>

```

```
</body>  
</html>
```

CSS Text

In this section you will learn how to style text on your web pages using CSS.

Formatting Text with CSS

CSS provides several properties that allows you to define various text styles such as color, alignment, spacing, decoration, transformation, etc. very easily and effectively.

The commonly used text properties are: `text-align`, `text-decoration`, `text-transform`, `text-indent`, `line-height`, `letter-spacing`, `word-spacing`, and more.

These properties give you precise control over the visual appearance of the characters, words, spaces, and so on.

Let's see how to set these text properties for an element in more detail.

Text Color

The color of the text is defined by the CSS `color` property.

The style rule in the following example will define the default text color for the page

Example

```
body {  
    color: #434343;  
}
```

Although, the color property seems like it would be a part of the CSS text, but it is actually a standalone property in CSS.

Text Alignment

The `text-align` property is used to set the horizontal alignment of the text.

Text can be aligned in four ways: to the left, right, center or justified (straight left and right margins).

Let's take a look at an example to understand how this property basically works.

Example

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

Note: When `text-align` is set to `justify`, each line is stretched so that every line has equal width (except the last line), and the left and right margins are straight. This alignment is generally used in publications such as magazines and newspapers.

Let's take a look at the following illustration to understand what these values actually mean.

Alice opened the door and found that it led into a small passage, not much larger than a rat hole: she knelt down and looked along the passage into the loveliest garden you ever saw.	Alice opened the door and found that it led into a small passage, not much larger than a rat hole: she knelt down and looked along the passage into the loveliest garden you ever saw.	Alice opened the door and found that it led into a small passage, not much larger than a rat hole: she knelt down and looked along the passage into the loveliest garden you ever saw.	Alice opened the door and found that it led into a small passage, not much larger than a rat hole: she knelt down and looked along the passage into the loveliest garden you ever saw.
left	center	right	justify

Text Decoration

The `text-decoration` property is used to set or remove decorations from text.

This property typically accepts one of the following values: `underline`, `overline`, `line-through`, and `none`. You should avoid underline text that is not a link, as it might confuse the visitor.

Let's try out the following example to understand how it basically works:

Example

```
h1 {  
    text-decoration: overline;  
}  
h2 {  
    text-decoration: line-through;  
}  
h3 {  
    text-decoration: underline;  
}
```

Removing the Default Underline from Links

The `text-decoration` property is extensively used to remove the default underline from the [HTML links](#).

Let's take a look at the following example to understand how it basically works:

Example

```
a {  
    text-decoration: none;  
    border-bottom: 1px dotted;  
}
```

Note: When you create an [HTML link](#), browsers built in style sheet automatically underline it and applies a blue color, so that the readers can clearly see which text is clickable.

Text Transformation

The `text-transform` property is used to set the cases for a text.

Text are often written in mixed case. However, in certain situations you may want to display your text in entirely different case. Using this property you can change an element's text content into uppercase or lowercase letters, or capitalize the first letter of each word without modifying the original text.

Let's try out the following example to understand how it basically works:

Example

```
h1 {  
    text-transform: uppercase;  
}  
h2 {  
    text-transform: capitalize;  
}  
h3 {  
    text-transform: lowercase;  
}
```

Text Indentation

The `text-indent` property is used to set the indentation of the first line of text within a block of text. It is typically done by inserting the empty space before the first line of text.

The size of the indentation can be specified using percentage (%), length values in pixels, ems, etc.

The following style rule will indent the first line of the paragraphs by 100 pixels.

Example

```
p {  
    text-indent: 100px;  
}
```

Note: Whether the text is indented from the left or from the right depends on the direction of text inside the element, defined by the CSS `direction` property.

Letter Spacing

The `letter-spacing` property is used to set extra spacing between the characters of text.

This property can take a length value in pixels, ems, etc. It may also accept negative values. When setting letter spacing, a length value indicates spacing in addition to the default inter-character space.

Let's check out the following example to understand how it really works:

Example

```
h1 {  
    letter-spacing: -3px;  
}  
p {  
    letter-spacing: 10px;  
}
```

Word Spacing

The `word-spacing` property is used to specify additional spacing between the words.

This property can accept a length value in pixels, ems, etc. Negative values are also allowed.

Let's try out the following example to understand how this property works:

Example

```
p {  
    word-spacing: 20px;  
    text-align: justify;  
}
```

Line Height

The `line-height` property is used to set the height of the text line.

It is also called leading and commonly used to set the distance between lines of text.

The value of this property can be a number, a percentage (%), or a length in pixels, ems, etc.

Example

```
p {  
    line-height: 1.2;  
}
```

When the value is a number, the line height is calculated by multiplying the element's font size by the number. While, percentage values are relative to the element's font size.

Note: Negative values for the `line-height` property are not allowed. This property is also a component of the CSS `font` shorthand property.

If the value of the `line-height` property is greater than the value of the `font-size` for an element, this difference (called the "leading") is cut in half (called the "half-leading") and distributed evenly on the top and bottom of the in-line box. Let's check out an example:

Example

```
p {  
    font-size: 14px;  
    line-height: 20px;  
    background-color: #f0e68c;  
}
```


CSS Links

In this section you will learn how to style different states of a link using CSS.

Styling Links with CSS

Links or hyperlinks are an essential part of a website. It allows visitors to navigate through the site. Therefore styling the links properly is an important aspect of building a user-friendly website.

See the **HTML Book 1** on [HTML links](#) to learn more about links and how to create them.

A link has four different states — `link`, `visited`, `active` and `hover`. These four states of a link can be styled differently through using the following anchor pseudo-class selectors.

- `a:link` — define styles for normal or unvisited links.
- `a:visited` — define styles for links that the user has already visited.
- `a:hover` — define styles for a link when the user place the mouse pointer over it.
- `a:active` — define styles for links when they are being clicked.

You can specify any CSS property you'd like e.g. `color`, `font`, `background`, `border`, etc. to each of these [selectors](#) to customize the style of links, just like you do with the normal text.

```
a:link { /* unvisited link */
  color: #ff0000;
  text-decoration: none;
  border-bottom: 1px solid;
}
a:visited { /* visited link */
  color: #ff00ff;
}
a:hover { /* mouse over link */
  color: #00ff00;
```

```
border-bottom: none;
}
a:active { /* active link */
  color: #00ffff;
}
```

Modifying Standard Link Styles

In all major web browsers such as Chrome, Firefox, Safari, etc. links on the web pages have underlines and uses the browser's default link colors, if you do not set the styles exclusively for them.

By default, text links will appear as follow in most of the browsers:

- An [unvisited link](#) as underlined blue text.
- A [visited link](#) as underlined purple text.
- An [active link](#) as underlined red text.

However, there is no change in the appearance of link in case of the hover state. It remains blue, purple or red depending on which state (i.e. unvisited, visited or active) they are in.

Now let's see how to customize the links by overriding its default styling.

Making Text Links Look Like Buttons

You can also make your ordinary text links look like button using CSS. To do this we need to utilize few more CSS properties such as `background-color`, `border`, `display`, `padding`, etc. You will learn about these properties in detail in upcoming chapters.

Let's check out the following example and see how it really works:

Example

```
a:link, a:visited {
    color: white;
    background-color: #1ebba3;
    display: inline-block;
    padding: 10px 20px;
    border: 2px solid #099983;
    text-decoration: none;
    text-align: center;
    font: 14px Arial, sans-serif;
}
a:hover, a:active {
    background-color: #9c6ae1;
    border-color: #7443b6;
}
```

CSS Lists

In this section you will learn how to format HTML lists using CSS.

Types of HTML Lists

There are three different types of list in HTML:

- **Unordered lists** — A list of items, where every list items are marked with bullets.
- **Ordered lists** — A list of items, where each list items are marked with numbers.
- **Definition list** — A list of items, with a description of each item.

Changing the Marker Type of Lists

By default, items in an **ordered list** are numbered with Arabic numerals (1, 2, 3, 5, and so on), whereas in an **unordered list**, items are marked with round bullets (•).

But, you can change this default list marker type to any other type such as roman numerals, latin letters, circle, square, and so on using the **list-style-type** property.

Let's try out the following example to understand how this property actually works:

Example

```
ul {  
    list-style-type: square;  
    list-style-position: inside;  
  
}  
  
ol {  
    list-style-type: upper-roman;  
}
```

Using Images as List Markers

You can also set an image as a list marker using the **list-style-image** property.

The style rule in the following example assigns a transparent PNG image "arrow.png" as the list marker for all the items in the unordered list. Let's try it out and see how it works:

Example

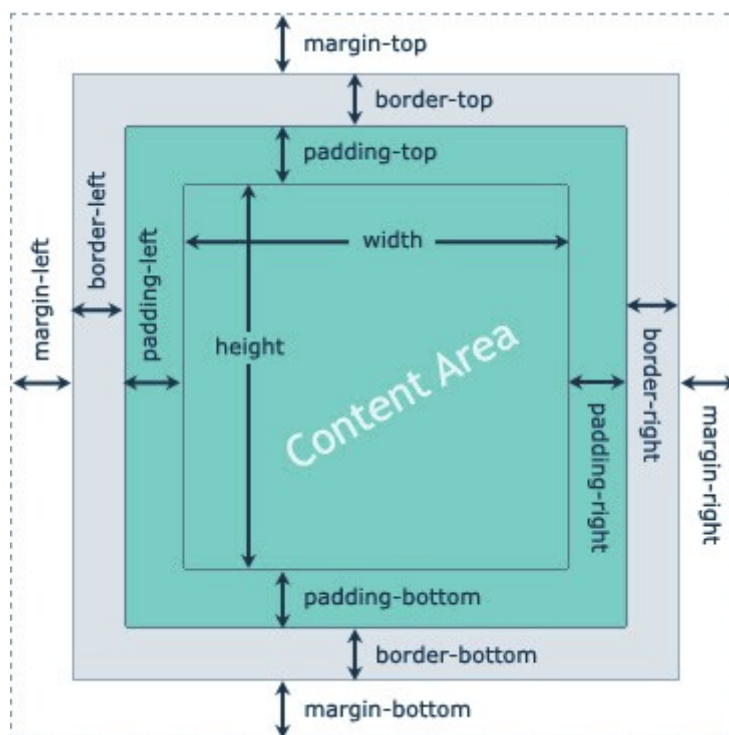
```
ul li {  
    list-style-image: url("images/bullet.png");  
}
```

CSS Box Model

What is Box Model?

Every element that can be displayed on a web page is comprised of one or more rectangular boxes. CSS box model typically describes how these rectangular boxes are laid out on a web page. These boxes can have different properties and can interact with each other in different ways, but every box has a content area and optional surrounding padding, border, and margin areas.

The following diagram demonstrates how the width, height, padding, border, and margin CSS properties determines how much space an element can take on a web page.



CSS Dimension

Setting Element Dimensions

CSS has several dimension properties, such as `width`, `height`, `max-width`, `min-width`, `max-height`, and `min-height` that allows you to control the width and height of an element. The following sections describe how to use these properties to create a better web page layout.

Setting the Width and Height

The `width` and `height` property defines the width and height of the content area of an element.

This width and height does not include paddings, borders, or margins.

Let's try out the following example and see how it actually works:

Here we use a `div` tags, a `div` tag create a division in HTML 5

Example

```
div {  
    width: 300px;  
    height: 200px;  
}
```

CSS Padding

In this section you will learn how to adjust the padding area of an element using CSS.

CSS Padding Properties

The CSS padding properties allow you to set the spacing between the content of an element and its border (or the edge of the element's box, if it has no defined border).

The padding is affected by the element's **background-color**. For instance, if you set the background color for an element it will be visible through the padding area.

Define Paddings for Individual Sides

You can specify the paddings for the individual sides of an element such as top, right, bottom, and left sides using the CSS **padding-top**, **padding-right**, **padding-bottom**, and the **padding-left** properties, respectively. Let's try out an example to understand how it works:

Example

```
h1 {  
    padding-top: 50px;  
    padding-bottom: 100px;  
}  
  
p {  
    padding-left: 75px;  
    padding-right: 75px;  
}
```

The Padding Shorthand Property

Example

```
h1 {  
    padding: 50px; /* apply to all four sides */  
}  
  
p {  
    padding: 25px 75px; /* vertical | horizontal */  
}  
  
div {  
    padding: 25px 50px 75px; /* top | horizontal | bottom */  
}
```

```
pre {  
    padding: 25px 50px 75px 100px; /* top | right | bottom | left */  
}
```

CSS Margin

In this section you will learn how to adjust space around an element using CSS.

CSS Margin Properties

The CSS margin properties allow you to set the spacing around the border of an element's box (or the edge of the element's box, if it has no defined border).

An element's margin is not affected by its [background-color](#), it is always transparent. However, if the parent element has the background color it will be visible through its margin area.

Setting Margins for Individual Sides

You can specify the margins for the individual sides of an element such as top, right, bottom, and left sides using the CSS `margin-top`, `margin-right`, `margin-bottom`, and the `margin-left` properties, respectively. Let's try out the following example to understand how it works:

Example

```
h1 {  
    margin-top: 50px;  
    margin-bottom: 100px;  
}  
  
p {  
    margin-left: 75px;  
    margin-right: 75px;  
}
```


The Margin Shorthand Property

The `margin` property is a shorthand property to avoid setting margin of each side separately, i.e., `margin-top`, `margin-right`, `margin-bottom` and `margin-left`.

Let's take a look at the following example to understand how it basically works:

Example

```
h1 {  
    margin: 50px; /* apply to all four sides */  
}  
  
p {  
    margin: 25px 75px; /* vertical | horizontal */  
}  
  
div {  
    margin: 25px 50px 75px; /* top | horizontal | bottom */  
}  
  
hr {  
    margin: 25px 50px 75px 100px; /* top | right | bottom | left */  
}
```

CSS Border

In this section you will learn how to define border around an element using CSS.

CSS Border Properties

The CSS border properties allow you to define the border area of an element's box.

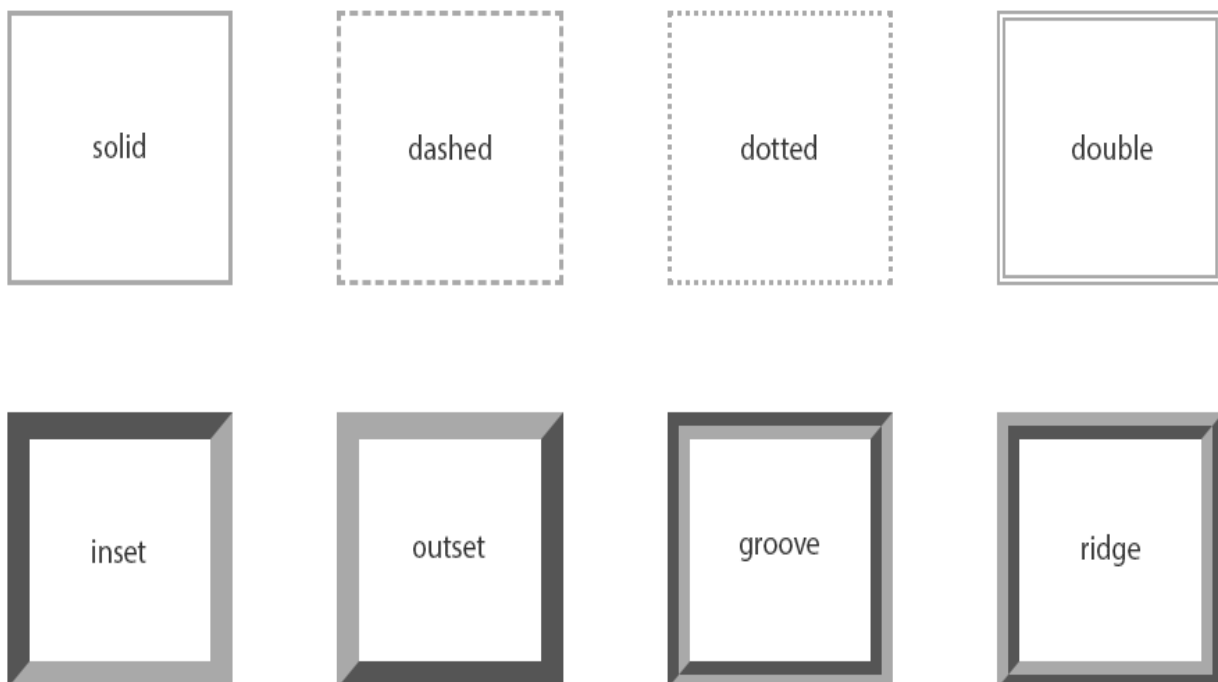
Borders appear directly between the margin and padding of an element. The border can either be a predefined style like, solid line, dotted line, double line, etc. or [an image](#).

The following section describes how to set the style, color, and width of the border.

Understanding the Different Border Styles

The [border-style](#) property sets the style of a box's border such as: solid, dotted, etc. It is a shorthand property for setting the line style for all four sides of the elements border.

The [border-style](#) property can have the following values: none, hidden, solid, dashed, dotted, double, inset, outset, groove, and ridge. Now, let's take a look at the following illustration, it gives you a sense of the differences between the border style types.



The values [none](#) and [hidden](#) displays no border, however, there is a slight difference between these two values. In the case of table cell and border collapsing, the [none](#) value has the lowest priority, whereas the [hidden](#) value has the highest priority, if any other conflicting border is set.

The values `inset`, `outset`, `groove`, and `ridge` creates a 3D like effect which essentially depends on the `border-color` value. This is typically achieved by creating a "shadow" from two colors that are slightly lighter and darker than the border color. Let's check out an example:

Example

```
h1 {  
    border-style: dotted;  
}  
  
p {  
    border-style: ridge;  
}
```

Class and ID Selectors

So far we've looked solely at those that represent an HTML tag. You can also define your own selectors in the form of class and ID selectors.

The benefit of this is that you can have the same HTML element, but present it differently depending on its class or ID.

In CSS, a class selector is a name preceded by a full stop (".") and an ID selector is a name preceded by a hash character ("#").

```
#top {  
    background-color: #ccc;  
    padding: 20px  
}  
  
.intro {  
    color: red;  
    font-weight: bold;  
}
```

The difference between an ID and a class is that an ID can be used to identify one element, whereas a class can be used to identify more than one.

In [CSS](#) , selectors are used to target a specific element or range of elements on a web page. Once an element has been targeted, a style or set of styles can be applied to the element.

CSS Class vs ID Selectors

What is the difference between the class and ID selectors?

The ID Selector is Unique

The id selector allows you to define style rules that apply to a single element on the web page. Each web page can only have one element with a single [ID attribute](#) . This means the ID selector can never be used to style more than one element.

ID selectors are defined using a hash sign. They are immediately followed by the ID value that you want to apply a set of style rules to. Here is an example of the ID selector in action:

```
<style>
#betaBanner {
    color: white;
    background-color: orange;
}
</style>
<p id="betaBanner">This is a banner.</p>
```

The Class Selector is Not Unique

A class selector allows you to define style rules that apply to any element with a class attribute equal to a certain value.

As we discussed earlier, the ID selector can only be used to style one element. This is because IDs can only be used once on a web page. Classes, on the other hand, can be used across multiple elements. So, if you apply a style using a class selector, any element which shares that class will be subject to the styles you define.

Full Practical Example

Here is a Full HTML File

<https://github.com/modcomlearning/Content2023/blob/master/CSSLesson2.html>

Here the Full External CSS used in above HTML

<https://github.com/modcomlearning/Content2023/blob/master/css/style.css>

Class selectors are defined using a period, followed by the value of the class that you want to apply a set of styles to. Here's an example of the class selector in action:

```
<html>

<p class="orangeBackground">This is a banner.</p>
<div class="orangeBackground">This is a banner.</div>

<style>

.orangeBackground {
    background-color: orange;
```

```
}  
</style>
```

You Can Use Both ID and CSS Class Selectors

There are no rules in HTML that prevent you from assigning an element both an ID and a class.

Suppose you want to apply the styles associated with a class called `backgroundOrange` to a `<div>` tag. However, you also want to apply a few unique styles to the `<div>`. You could do so using this code:

```
<div class="backgroundOrange" id="customDiv"></div>
```

CSS3 Animations

Creating CSS3 Animations

In the previous chapter you've seen how to do simple animations like animating a property from one value to another via CSS3 transitions feature. However, the CSS3 transitions provide little control on how the animation progresses over time.

The CSS3 animations take it a step further with keyframe-based animations that allow you to specify the changes in CSS properties over time as a set of keyframes, like flash animations. Creating CSS animations is a two step process, as shown in the example below:

- The first step of building a CSS animation is to defining individual keyframes and naming an animation with a keyframes declaration.

- The second step is referencing the keyframes by name using the `animation-name` property as well as adding `animation-duration` and other optional `animation properties` to control the animation's behavior.

However, it is not necessary to define the keyframes rules before referencing or applying it. The following example will show you how to animate a `<div>` box horizontally from one position to another using the CSS3 animation feature.

CSS3 Animation Properties.

The following table provides a brief overview of all the animation-related properties.

Property	Description
<code>animation</code>	A shorthand property for setting all the animation properties in single declaration.
<code>animation-name</code>	Specifies the name of <code>@keyframes</code> defined animations that should be applied to the selected element.
<code>animation-duration</code>	Specifies how many seconds or milliseconds that an animation takes to complete one cycle of the animation.
<code>animation-timing-function</code>	Specifies how the animation will progress over the duration of each cycle i.e. the easing functions.
<code>animation-delay</code>	Specifies when the animation will start.
<code>animation-iteration-count</code>	Specifies the number of times an animation cycle should be played before stopping.
<code>animation-direction</code>	Specifies whether or not the animation should play in reverse on alternate cycles.
<code>animation-fill-mode</code>	Specifies how a CSS animation should apply styles to its target before and after it is executing.
<code>animation-play-state</code>	Specifies whether the animation is running or paused.
<code>@keyframes</code>	Specifies the values for the animating properties at various points during the animation.

Here is a simple example to Animate a Moving Car.

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Translate Animation</title>
<style>
.box {
  margin: 50px;
  width:400px;
  height:203px;
  background: url("/images/car2.jpg") no-repeat;
  position: relative;
  background-size: cover;
  /* Chrome, Safari, Opera */
  -webkit-animation-name: moveit;
  -webkit-animation-duration: 2s;

  /* Standard syntax */
  animation-name: moveit;
  animation-duration: 5s;
  animation-iteration-count: 10;
  animation-direction: alternate;
}
/* Chrome, Safari, Opera */
@-webkit-keyframes moveit {
  from {left: 0;}
  to {left: 50%;}
}
/* Standard syntax */
@keyframes moveit {
  from {left: 0;}
  to {left: 50%;}
}
</style>

</head>
<body>
<h1>Moving Car</h1>
<div class="box"></div>
<p>Lorem ipsum dolor sit amet, consectetur adipisicing elit. Maiores impedit quidem
laborum ratione assumenda consequatur officia voluptabo?</p>
</body>
</html>
```


CSS3 Transitions

The CSS3 transition feature allows the changes in CSS property values to occur smoothly over a specified duration.

Understanding CSS3 Transitions

Normally when the value of a CSS property changes, the rendered result is instantly updated. A common example is changing the background color of a button on mouse hover. In a normal scenario the background color of the button is changes immediately from the old property value to the new property value when you place the cursor over the button.

CSS3 introduces a new transition feature that allows you to animate a property from the old value to the new value smoothly over time. The following example will show you how to animate the **background-color** of an HTML button on mouse hover

Here is an example on Transition

```
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Example of CSS3 Transition</title>
<style>

button {
  color: #fff;
  border: none;
  padding: 10px 20px;
  font: bold 18px sans-serif;
  background: #2afd3f;
  -webkit-transition: background 2s; /* For Safari 3.0 to 6.0 */
  transition: background 2s; /* For modern browsers */
}
```

```
}  
button:hover {  
  background: #393c3a;  
}  
</style>  
</head>  
<body>  
  <button type="button">Hover on me</button>  
</body>  
</html>
```

Revision Questions

- What is the full form of CSS?
- Why was CSS developed?
- Differentiate between CSS and HTML
- What are the major versions of CSS?
- What are the different ways you could integrate CSS into your HTML page?
- What is the meaning of cascading?
- What are the advantages of using CSS?
- What is the difference between the usage of an ID and a Class?
- What are the ways to assign a certain colour to an element in CSS?
- How will you target an h2 and h3 with the same styling?
- What are the various font-related attributes in CSS?
- What are the various text-related attributes in CSS?
- Explain responsive web design.
- What are the different types of Selectors in CSS?
- What is the difference between inline, inline-block, and block?
- Ids uses a # in CSS . True/False
- Classes uses . In CSS True/False
- CSS3 allows animations. True/False
- Explain the CSS Box Model and its different elements

References

<https://www.geeksforgeeks.org/css/>

https://www.w3schools.com/css/css_intro.asp

<https://www.tutorialspoint.com/css/index.htm>

<https://developer.mozilla.org/en-US/docs/Web/CSS>

<https://www.javatpoint.com/css-tutorial>