



CRYSTALLIZE
TECHNOLOGIES

FULL STACK WEB DEVELOPMENT

ABSTRACT :

Full stack web development encompasses all aspects of building a website or web application, from the user interface to the back-end logic and database management. This guide provides an overview of the key technologies and concepts involved in full stack Front-End CSS

Madhan HK

CRYSTALLIZE TECHNOLOGIES

CSS Introduction

CSS is the language we use to style a Web page.

What is CSS?

- 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.

CSS Example

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

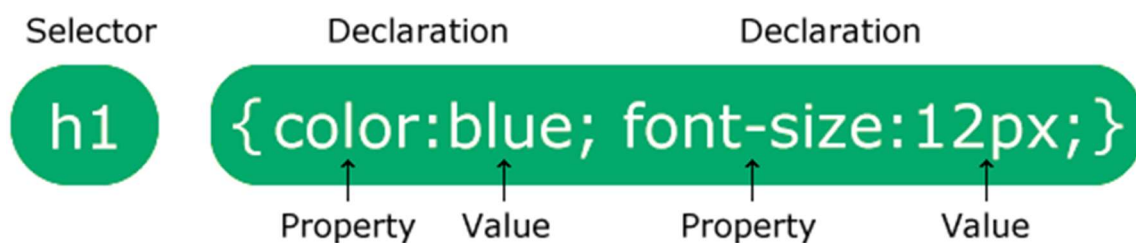
```
h1 {  
  color: white;  
  text-align: center;  
}
```

```
p {  
  font-family: verdana;
```

```
font-size: 20px;  
}
```

A CSS rule consists of a selector and a declaration block.

CSS Syntax



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.

Multiple CSS declarations are separated with semicolons, and declaration blocks are surrounded by curly braces.

Example

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

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

Example Explained

- **p** is a selector in CSS (it points to the HTML element you want to style: <p>).
- **color** is a property, and **red** is the property value
- **text-align** is a property, and **center** is the property value

CSS Selectors

A CSS selector selects the HTML element(s) you want to style.

The CSS element Selector

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

Example

Here, all <p> elements on the page will be center-aligned, with a red text color:

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

The CSS id Selector

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

The id of an element is 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.

Example

The CSS rule below will be applied to the HTML element with id="para1":

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

The CSS class Selector

The class selector selects HTML elements with a specific class attribute.

To select elements with a specific class, write a period (.) character, followed by the class name.

Example

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

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

Example

In this example only <p> elements with class="center" will be red and center-aligned:

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

HTML elements can also refer to more than one class.

Example

In this example the `<p>` element will be styled according to `class="center"` and to `class="large"`:

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

Note: A class name cannot start with a number!

The CSS Universal Selector

The universal selector (*) selects all HTML elements on the page.

Example

The CSS rule below will affect every HTML element on the page:

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

The CSS Grouping Selector

The grouping selector selects all the HTML elements with the same style definitions.

Look at the following CSS code (the `h1`, `h2`, and `p` elements have the same style definitions):

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

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

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

It will be better to group the selectors, to minimize the code.

To group selectors, separate each selector with a comma.

Example

In this example we have grouped the selectors from the code above:

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

Three Ways to Insert CSS

There are three ways of inserting a style sheet:

- External CSS
- Internal CSS
- Inline CSS

External CSS

With an external style sheet, you can change the look of an entire website by changing just one file!

Each HTML page must include a reference to the external style sheet file inside the <link> element, inside the head section.

Example

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

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="mystyle.css">
</head>
<body>

<h1>This is a heading</h1>
<p>This is a paragraph.</p>

</body>
</html>
```

An external style sheet can be written in any text editor, and must be saved with a .css extension.

The external .css file should not contain any HTML tags.

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

"mystyle.css"

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

h1 {
```



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

Internal CSS

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

The internal style is defined inside the `<style>` element, inside the head section.

Example

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

```
<!DOCTYPE html>
<html>
<head>
<style>
body {
  background-color: linen;
}

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

<h1>This is a heading</h1>
<p>This is a paragraph.</p>
```

```
</body>
```

```
</html>
```

Inline CSS

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:

```
<!DOCTYPE html>
```

```
<html>
```

```
<body>
```

```
<h1 style="color:blue;text-align:center;">This is a heading</h1>
```

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

```
</body>
```

```
</html>
```

CSS Comments

CSS comments are not displayed in the browser, but they can help document your source code.

CSS Comments

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

Comments are ignored by browsers.

A CSS comment is placed inside the `<style>` element, and starts with `/*` and ends with `*/`:

Example

```
/* This is a single-line comment */
```

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

You can add comments wherever you want in the code:

Example

```
p {  
  color: red; /* Set text color to red */  
}
```

Even in the middle of a code line:

Example

```
p {  
  color: /*red*/blue;  
}
```

Comments can also span multiple lines:

Example

```
/* This is  
a multi-line  
comment */
```

```
p {  
  color: red;
```

}

CSS RGB Colors

An RGB color value represents RED, GREEN, and BLUE light sources.

RGB Value

In CSS, a color can be specified as an RGB value, using this formula:

rgb(red, green, blue)

Each parameter (red, green, and blue) defines the intensity of the color between 0 and 255.

For example, `rgb(255, 0, 0)` is displayed as red, because red is set to its highest value (255) and the others are set to 0.

To display black, set all color parameters to 0, like this: `rgb(0, 0, 0)`.

To display white, set all color parameters to 255, like this: `rgb(255, 255, 255)`.

RGBA Value

RGBA color values are an extension of RGB color values with an alpha channel - which specifies the opacity for a color.

An RGBA color value is specified with:

rgba(red, green, blue, alpha)

The alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent at all):

CSS HEX Colors

A hexadecimal color is specified with: #RRGGBB, where the RR (red), GG (green) and BB (blue) hexadecimal integers specify the components of the color.

HEX Value

In CSS, a color can be specified using a hexadecimal value in the form:

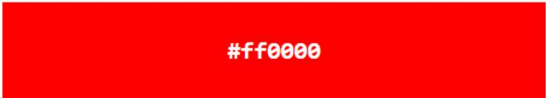


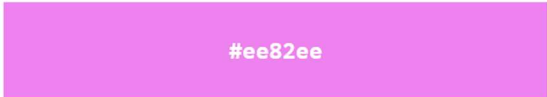

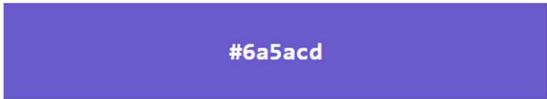
#rrggbb

Where rr (red), gg (green) and bb (blue) are hexadecimal values between 00 and ff (same as decimal 0-255).

For example, #ff0000 is displayed as red, because red is set to its highest value (ff) and the others are set to the lowest value (00).

To display black, set all values to 00, like this: #000000.

To display white, set all values to ff, like this: #ffffff.

 #ff0000	 #0000ff
 #3cb371	 #ee82ee
 #ffa500	 #6a5acd

CSS Backgrounds

The CSS background properties are used to add background effects for elements.

- background-color
- background-image
- background-repeat
- background-attachment
- background-position
- background (shorthand property)

CSS 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

CSS Margins

The CSS **margin** properties are used to create space around elements, outside of any defined borders.

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

Margin - Individual Sides

CSS has properties for specifying the margin for each side of an element:

- **margin-top**
- **margin-right**
- **margin-bottom**
- **margin-left**

If the **margin** property has four values:

- **margin: 25px 50px 75px 100px;**
 - top margin is 25px
 - right margin is 50px
 - bottom margin is 75px
 - left margin is 100px

Example

Use the margin shorthand property with four values:

```
p {  
  margin: 25px 50px 75px 100px;  
}
```

If the **margin** property has three values:

- **margin: 25px 50px 75px;**
 - top margin is 25px
 - right and left margins are 50px
 - bottom margin is 75px

Example

Use the margin shorthand property with three values:

```
p {  
  margin: 25px 50px 75px;  
}
```

If the **margin** property has two values:

- **margin: 25px 50px;**
 - top and bottom margins are 25px
 - right and left margins are 50px

Example

Use the margin shorthand property with two values:

```
p {  
  margin: 25px 50px;  
}
```


If the **margin** property has one value:

- **margin: 25px;**
 - all four margins are 25px

Example

Use the margin shorthand property with one value:

```
p {  
  margin: 25px;  
}
```

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**

If the **padding** property has four values:

- **padding: 25px 50px 75px 100px;**
 - top padding is 25px
 - right padding is 50px
 - bottom padding is 75px
 - left padding is 100px

Example

Use the padding shorthand property with four values:

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

If the **padding** property has three values:

- **padding: 25px 50px 75px;**
 - top padding is 25px
 - right and left paddings are 50px
 - bottom padding is 75px

Example

Use the padding shorthand property with three values:

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

If the **padding** property has two values:

- **padding: 25px 50px;**
 - top and bottom paddings are 25px
 - right and left paddings are 50px

Example

Use the padding shorthand property with two values:

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

If the **padding** property has one value:

- **padding: 25px;**
 - all four paddings are 25px

Example

Use the padding shorthand property with one value:

```
div {  
  padding: 25px;  
}
```

The CSS **height** and **width** properties are used to set the height and width of an element.

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

This element has a height of 50 pixels and a width of 100%.

CSS Setting height and width

The **height** and **width** properties are used to set the height and width of an element.

The height and width properties do not include padding, borders, or margins. It sets the height/width of the area inside the padding, border, and margin of the element.

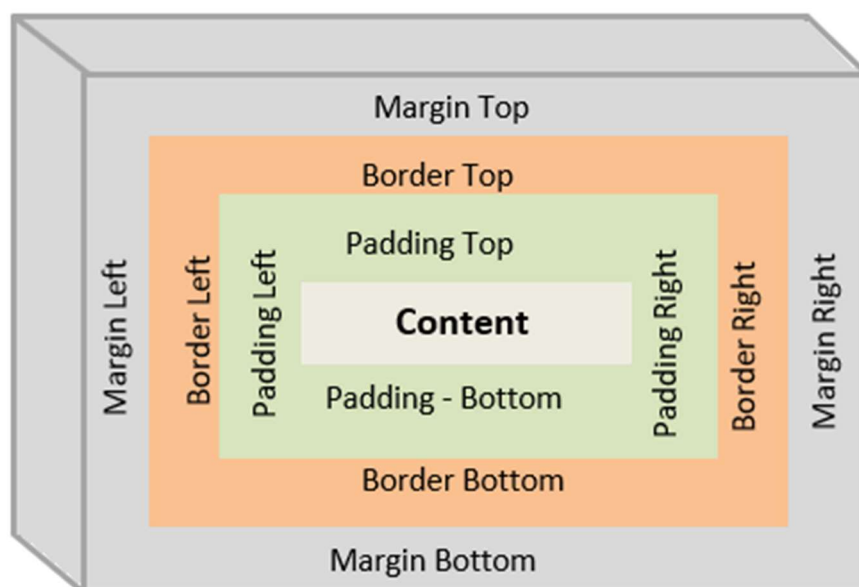
Set the height and width of a <div> element:

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

The CSS Box Model

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: content, padding, borders and margins. 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

How To Add 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 ``).

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

Font Awesome Icons

To use the Font Awesome icons, go to fontawesome.com, sign in, and get a code to add in the `<head>` section of your HTML page:

```
<script src="https://kit.fontawesome.com/yourcode.js" crossorigin="anonymous"></script>
```

The `display` property is the most important CSS property for controlling layout.

The display Property

The `display` property is used to specify how an element is shown on a web page.

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`.

The **display** property is used to change the default display behavior of HTML elements.

The **position** property specifies the type of positioning method used for an element (static, relative, fixed, absolute or sticky).

The position Property

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

There are five different position values:

- **static**
- **relative**
- **fixed**
- **absolute**
- **sticky**

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. They also work differently depending on the position value.

What is Responsive Web Design?

Responsive web design makes your web page look good on all devices.

Responsive web design uses only HTML and CSS.

Responsive web design is not a program or a JavaScript.

Designing For The Best Experience For All Users

Web pages can be viewed using many different devices: desktops, tablets, and phones. Your web page should look good, and be easy to use, regardless of the device.

What is The Viewport?

The viewport is the user's visible area of a web page.

The viewport varies with the device, and will be smaller on a mobile phone than on a computer screen.

Before tablets and mobile phones, web pages were designed only for computer screens, and it was common for web pages to have a static design and a fixed size.

Then, when we started surfing the internet using tablets and mobile phones, fixed size web pages were too large to fit the viewport. To fix this, browsers on those devices scaled down the entire web page to fit the screen.

This was not perfect!! But a quick fix.

Setting The Viewport

HTML5 introduced a method to let web designers take control over the viewport, through the `<meta>` tag.

You should include the following `<meta>` viewport element in all your web pages:

```
<meta name="viewport" content="width=device-width, initial-scale=1.0">
```

This gives the browser instructions on how to control the page's dimensions and scaling.

The `width=device-width` part sets the width of the page to follow the screen-width of the device (which will vary depending on the device).

The `initial-scale=1.0` part sets the initial zoom level when the page is first loaded by the browser.



CRYSTALL
TECHNOLOGIES