



PRIME INTUIT

Finishing School

Cascading Stylesheets (CSS)



What is CSS

CSS (Cascading Style Sheets) is the code that styles web content. Like HTML, CSS is not a programming language. It's not a markup language either. **CSS is a style sheet language.**

CSS is what you use to selectively style HTML elements. For example, this CSS selects paragraph text, setting the color to red:

CSS can be added to HTML documents in 3 ways:

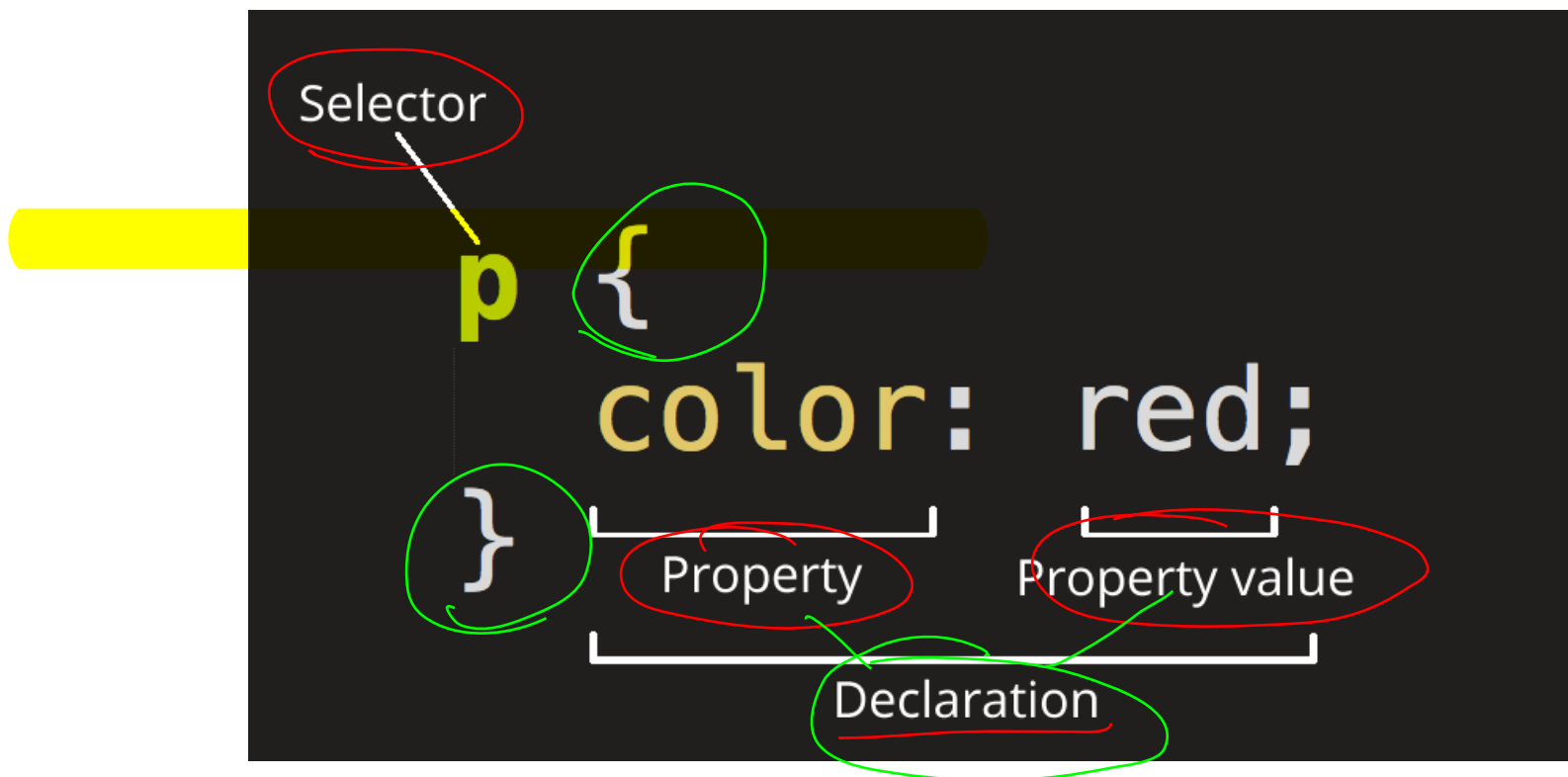
1. **Inline** - by using the style attribute inside HTML elements.
2. **Internal** - by using a <style> element in the <head> section.
3. **External** - by using a <link> element to link to an external CSS file.



CSS ruleset

```
P {  
  color: red;  
}
```

Selector {
 property: prop_value;
} declaration





CSS ruleset

Selector

This is the **HTML element name** at the start of the ruleset. It **defines the element(s) to be styled** (in this **example, <p> elements**). To style a different element, **change the selector**.

Declaration

This is a **single rule like color: red;**. It **specifies** which of the **element's properties** you want to style.

Properties

These are **ways** in which you can **style an HTML element**. (In this example, **color is a property** of the **<p> elements**.) In CSS, you choose which properties you want to affect in the rule.

Property value

To the **right of the property—after the colon—**there is the **property value**. This chooses **one out of many possible appearances** for a given property. (For example, there are **many color values** in **addition to red**.)



CSS ruleset

Other key parts of the syntax:

Apart from the **selector**, each **ruleset** must be wrapped in curly braces. (**{}**)

Within each **declaration**, you must use a **colon (:)** to **separate the property from its value** or values.

Within **each ruleset**, you must use a **semicolon (;)** to **separate each declaration** from the next one.



PRIME INTUIT

Finishing School

CSS ruleset

To modify multiple property values in one ruleset, write them separated by semicolons

```
p {  
  color: red;  
  width: 500px;  
  border: 1px solid black;  
}
```



Applying CSS to HTML

To make the code work, we still need to apply this CSS (above) to your HTML document. Otherwise, the styling won't change the appearance of the HTML.

```
1 <!DOCTYPE html>
2 <html lang="en" dir="ltr">
3   <head>
4     <meta charset="utf-8">
5     <title>Practice CSS</title>
6     <link rel="stylesheet" href="D:\ATOM\CSS\PI_B3_CSS_Level1.css">
7   </head>
```

To link an external CSS file to your HTML file.

Open your HTML file and insert `<link>` tag inside your `<head>` tag

The `href` element in your link tag should be assigned the file path of the CSS file.



Applying color to HTML elements using CSS

To make the code work, we still need to apply this CSS (above) to your HTML document. Otherwise, the styling won't change the appearance of the HTML.

```
1 <!DOCTYPE html>
2 <html lang="en" dir="ltr">
3   <head>
4     <meta charset="utf-8">
5     <title>Practice CSS</title>
6     <link rel="stylesheet" href="D:\ATOM\CSS\PI_B3_CSS_Level1.css">
7   </head>
```

To link an external CSS file to your HTML file.

Open your HTML file and insert `<link>` tag inside your `<head>` tag
The `href` element in your link tag should be assigned the file path of the CSS file.



Try it ! - HTML

```
<!DOCTYPE html>
<html lang="en" dir="ltr">
  <head>
    <meta charset="utf-8">
    <title>Practice CSS</title>
    <link rel="stylesheet" href="D:\ATOM\CSS\PI_B3_CSS_Level1.css">
  </head>
  <body>
    <h2>Prime Intuit Finishing School</h2>
    <p>We at Prime Intuite are very passionate about shaping young minds</p>
    <h3>Our Courses Include</h3>
    <ol>
      <li>Personality development</li>
      <li>Statistics</li>
      <li>Python Basics</li>
      <li>Python Libraries like Numpy, Pandas</li>
      <li>Visualization using Python</li>
      <li>Machine Learning</li>
      <li>NLP</li>
      <li>HTML, CSS, JS, Angular</li>
      <li>SQL Level One</li>
      <li>Tableau</li>
    </ol>
```



Try it ! - HTML

<h4> Additional topics like: </h4>

Advance Excel

Cloud Concepts

Data warehouse

SDLC

Basic JAVA

Guest lectures from Industry experts

<p>

<div class="prompt">

We have left no stone unturned to give you the best platform, to be job ready for a great carrer in IT industry</p>

</div>

< /p>

</body>

</html>



Try it ! CSS

/* syntax for CSS is :

selector{

property: value;

} */

body{

background: rgb(184, 186, 177)

}

h2{

color: purple;

}

ol{

color: rgb(197, 245, 66);

}

p{

color: #1dc487;

}

h3{

color: rgba(181, 42, 90, 0.5);

}

ul{

border-color: red;

border-width: 2px;

border-style: solid;

}

.prompt{

color: rgb(42, 151, 181);

background-color: rgb(195, 230, 23);

}

#spn1{

background-color: lightgrey;

}



Prime Intuit Finishing School

We at Prime Intuite are very passionate about shaping young minds

Our Courses Include

1. Personality development
2. Statistics
3. Python Basics
4. Python Libraries like Numpy, Pandas
5. Visualization using Python
6. Machine Learning
7. NLP
8. HTML, CSS, JS, Angular
9. SQL Level One
10. Tableau

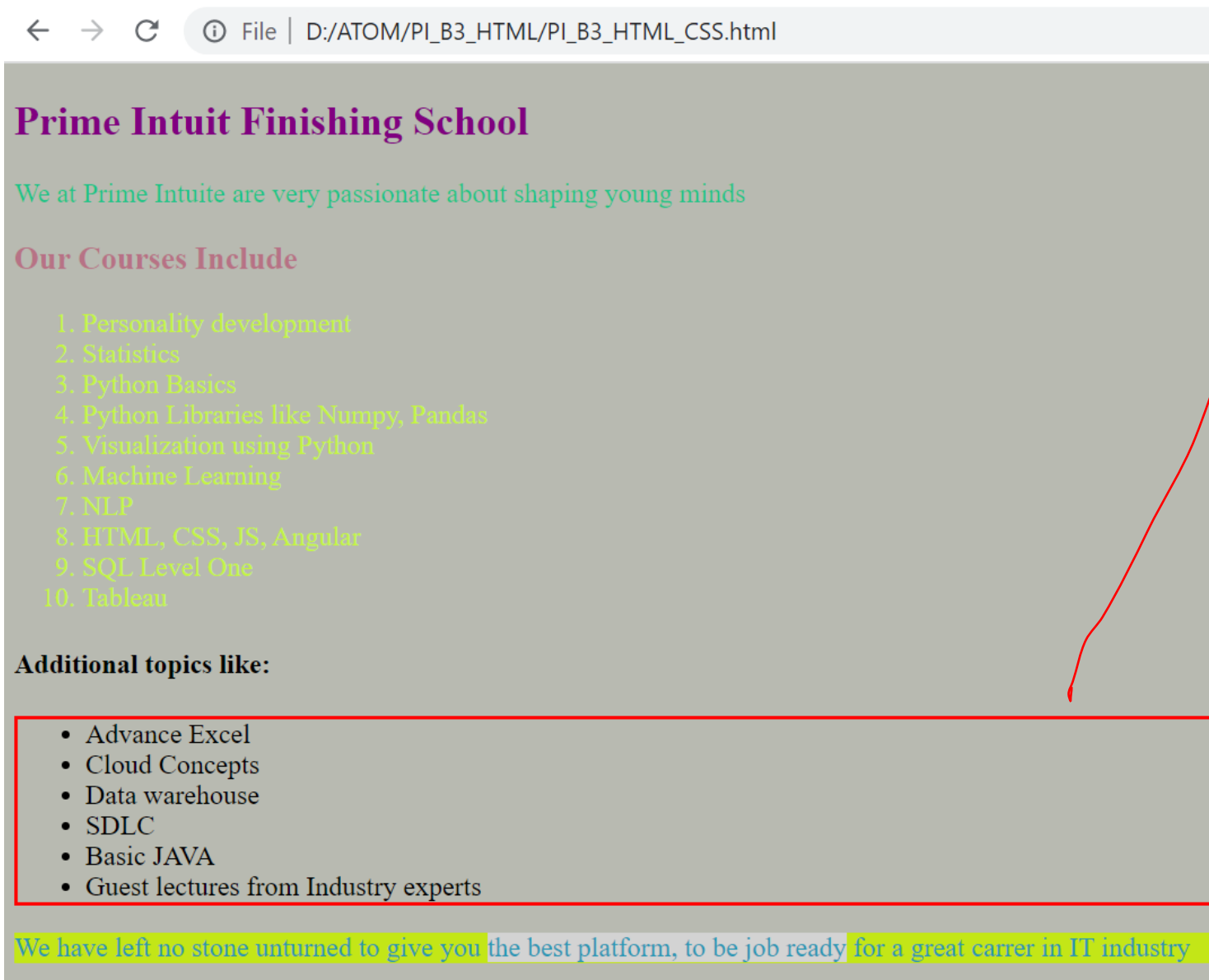
Additional topics like:

- Advance Excel
- Cloud Concepts
- Data warehouse
- SDLC
- Basic JAVA
- Guest lectures from Industry experts

We have left no stone unturned to give you the best platform, to be job ready for a great carrer in IT industry



Your HTML Page after applying CSS





Selecting multiple elements

We can also select multiple elements and apply a single ruleset to all of them. Separate multiple selectors by commas.

```
p, li, h1 {  
  color: red;  
}
```

CSS class selector

Selecting all HTML elements with `class="center"` will be red and center-aligned:

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

Selecting only specific HTML elements should be affected by a class, example `<p>`:

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

CSS Universal Selector

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

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


CSS Grouping Selector

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

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

Summary Selectors

Selector	Description
# id	Select the element with id
.class	Select all elements with class
element.class	Select only element with specified class
*	Select all elements
Element	Selects the specified element
Elements, elements	Selects multiple elements

Ways to Insert CSS

- There are three ways of inserting a style sheet:
- External CSS
- Internal CSS
- Inline CSS

External CSS

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

```
<head>
```

```
<link rel="stylesheet" href="mystyle.css">
```

```
</head>
```

Internal CSS

Internal: 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.

```
<head>
<style>
body {
  background-color: linen; → skin color
}

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

Inline CSS

Inline: 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.

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

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

Cascading Order

Cascade, Specificity, and Inheritance

Stylesheets **cascade** — at a very simple level, this means that the **order of CSS rules matters**; when **two rules apply** that have **equal specificity**, the **one that comes last in the CSS** is the one that **will be used**.

Specificity is **how the browser decides which rule applies if multiple rules have different selectors, but could still apply to the same element**. It is basically a measure of how specific a selector's selection will be:

- **An element selector is less specific** — it will **select all elements** of that type that **appear on a page** — so will get a **lower score**.
- **A class selector is more specific** — it will select **only the elements on a page that have a specific class attribute value** — so will get a **higher score**.

Inheritance — some **CSS property values set on parent elements** are **inherited by their child elements**, and some aren't.

Cascading Order

Cascade, Specificity, and Inheritance

```
h4 {  
  font-size: 18px;  
  color: red;  
}  
h4 {  
  font-size: 14px;  
  color: blue;  
}
```

1st

but

prevails

```
.SRB {  
  font-size: 16px;  
  color: green;  
  background-color: lightgrey;  
}  
h5 {  
  font-size: 14px;  
  color: blue;  
}
```

day

specific

prevails

specific

```
<body>  
  <h1>Little by Little every day, What is ment to be yours will flow into your life, only if you take acti:  
  <h2>You are better then this!</h2>  
  <h3>No pain No gain</h3>  
  <h4>Text to understand Cascade / conflicting rules </h4>  
  <h5 class="SRB">Text to understand Specificity</h5>  
  <h6>Text to understand inheritance</h6>  
  <p>Brave does not always mean being loud and strong, sometimes its just a tiny voice from inside, telli  
  >This is an additional paragraph</p>
```


Cascading Order

Cascade, Specificity, and Inheritance

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

whole

```
span {  
  color: black;  
}
```

prevail

```
<body>  
  <h1>Little by Little every day, What is ment to be yours will flow into your life, only if you take act  
  <h2>You are better then this!</h2>  
  <h3>No pain No gain</h3>  
  <h4>Text to understand Cascade / conflicting rules </h4>  
  <h5 class="SRB">Text to understand Specificity</h5>  
  <h6>Taxt to understand inheritance</h6>  
  <p>Brave does not always mean being loud and strong, sometimes <span>its just a tiny voice </span>from  
>This is an additional paragraph</p>
```

Cascading Order

Cascade, Specificity, and Inheritance

```
.pydt {  
  color: purple;  
  border: 2px solid #ccc;  
  padding: 1em;  
}  
  
.sequ {  
  color: black;  
  font-weight: bold;  
}
```

list
li also
falls
ul styles

```
<h3>Data Types in Python</h3>  
<ul class="pydt">  
  <li>Boolean</li>  
  <li>Numbers  
    <ul>  
      <li>Integer</li>  
      <li>Float</li>  
      <li>Complex Numbers</li>  
    </ul>  
  </li>  
  <li>Sequence  
    <ul class="sequ">  
      <li>Sets  
        <ul>  
          <li>Lists</li>  
          <li>Tuples</li>  
          <li>Strings</li>  
        </ul>  
      </li>  
      <li>Dictionary</li>  
    </ul>  
  </li>  
</ul>
```

The box model

Everything in CSS has a box around it, and understanding these boxes is key to being able to create layouts with CSS, or to align items with other items.

Block and inline boxes

Characteristics refer to how the box behaves in terms of page flow and in relation to other boxes on the page. Boxes also have an **inner display type** and an **outer display type**.

A box with outer display type as **Block** :

- will break into a new
- will extend in the inline direction to fill the space available in its container. In most cases this means that the box will become as wide as its container, filling up 100% of the space available.
- width and height properties are supported.
- Padding, margin and border will cause other elements to be pushed away from the box.

Some HTML elements, such as `<h1>` and `<p>`, use block as their outer display type by default.

Block and inline boxes

A box with outer display type as **Inline** :

- will not break into a new
- Vertical padding, margins, and borders will apply but will not cause other inline boxes to move away from the box.
- ~~width and height~~ properties are ~~not supported~~.
- Horizontal padding, margins, and borders will apply and will cause other inline boxes to move away from the box.

Some HTML elements, such as `<a>`, ``, `` and `` use inline as their outer display type by default.

`Visit W3Schools.com!`

→ create hyperlink

Inner and Outer Display types

Inner Display types: dictates how elements inside that box are laid out. By default, the elements inside a box are laid out in normal flow, which means that they behave just like any other block and inline elements.

Outer Display types: dictates whether the box is block or inline.

Examples of different display types

```
p,  
ul {  
  border: 2px solid rebeccapurple;  
  padding: .5em;  
}  
  
.block,  
li {  
  border: 2px solid blue;  
  padding: .5em;  
}  
  
ul {  
  display: flex;  
  list-style: none;  
}  
  
.block {  
  display: block;  
}
```

```
<p>I am a paragraph. A short one.</p>  
<ul>  
  <li>Item One</li>  
  <li>Item Two</li>  
  <li>Item Three</li>  
</ul>  
<p>I am another paragraph. Some of the <span class="block">words</span>  
have been wrapped in a <span>span element</span>.</p>
```

Examples of different display types

```
p,  
ul {  
  border: 2px solid rebeccapurple;  
  padding: .5em;  
}  
  
.block,  
li {  
  border: 2px solid blue;  
  padding: .5em;  
}  
  
ul {  
  display: flex;  
  list-style: none;  
}  
  
.block {  
  display: block;  
}
```

```
<p>I am a paragraph. A short one.</p>  
<ul>  
  <li>Item One</li>  
  <li>Item Two</li>  
  <li>Item Three</li>  
</ul>  
<p>I am another paragraph. Some of the <span class="block">words</span>  
have been wrapped in a <span>span element</span>.</p>
```


Examples of different display types

```
p,  
ul {  
  border: 2px solid rebeccapurple;  
}  
  
span,  
li {  
  border: 2px solid blue;  
}  
  
ul {  
  display: inline-flex;  
  list-style: none;  
  padding: 0;  
}  
  
.inline {  
  display: inline;  
}
```

```
✓ <p>  
  I am a paragraph. Some of the  
  <span>words</span> have been wrapped in a  
  <span>span element</span>.  
</p>  
✓ <ul>  
  <li>Item One</li>  
  <li>Item Two</li>  
  <li>Item Three</li>  
</ul>  
<p class="inline">I am a paragraph. A short one.</p>  
<p class="inline">I am another paragraph. Also a short one.</p>
```

CSS Box Model

The CSS box model as a whole applies to block boxes. Inline boxes use just *some* of the behavior defined in the box model. The model defines how the different parts of a box — margin, border, padding, and content — work together to create a box that you can see on a page. To add some additional complexity, there is a standard and an alternate box model.

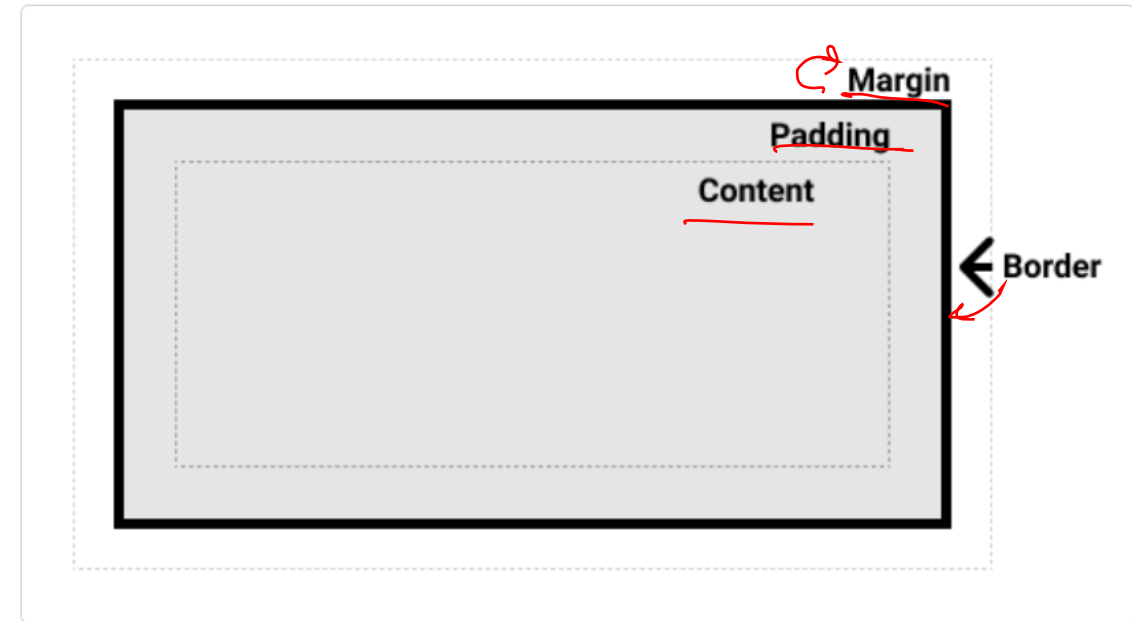
Parts of a box

Content box: The area where your content is displayed, which can be sized using properties like width and height.

Padding box: The padding sits around the content as white space; its size can be controlled using padding and related properties.

Border box: The border box wraps the content and any padding. Its size and style can be controlled using border and related properties.

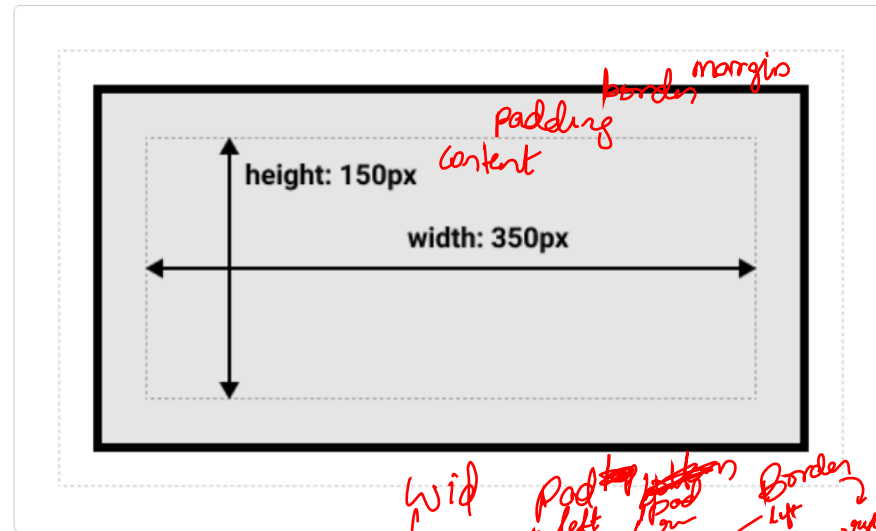
Margin box: The margin is the outermost layer, wrapping the content, padding, and border as whitespace between this box and other elements. Its size can be controlled using margin and related properties.



The Standard Box Model

In the standard box model, if you give a box a width and a height attribute, this defines the width and height of the content box. Any padding and border is then added to that width and height to get the total size taken up by the box.

```
.box {  
  width: 350px;  
  height: 150px;  
  margin: 10px;  
  padding: 25px;  
  border: 5px solid black;  
}
```



The actual space taken up by the box will be 410px wide ($350 + 25 + 25 + 5 + 5$) and 210px high ($150 + 25 + 25 + 5 + 5$).

The Alternative Box Model

In the Alternative box model, Any width is the width of the visible box on the page, therefore the content area width is that width minus the width for the padding and border.

```
.box {  
  width: 350px;  
  height: 150px;  
  margin: 10px;  
  padding: 25px;  
  border: 5px solid black;  
}
```

width = 350px. height = 150px



```
.box {  
  box-sizing: border-box;  
}
```

```
html {  
  box-sizing: border-box;  
}  
*, *::before, *::after {  
  box-sizing: inherit;  
}
```

By default, browsers use the standard box model. If you want to turn on the alternative model for an element, you do so by setting `box-sizing: border-box` on it. By doing this, you are telling the browser to use the border box

Comparing both box models

```
.box {  
  border: 5px solid rebeccapurple;  
  background-color: lightgray;  
  padding: 40px;  
  margin: 40px;  
  width: 300px;  
  height: 150px;  
}  
  
.alternate {  
  box-sizing: border-box;  
}
```

Both have a class of .box, which gives them the same width, height, margin, border, and padding. The only difference is that the second box has been set to use the alternative box model.

```
<div class="box">I use the standard box model.</div>  
<div class="box alternate">I use the alternate box model.</div>
```

Margins

The margin is an invisible space around your box. It pushes other elements away from the box. Margins can have positive or negative values. Setting a negative margin on one side of your box can cause it to overlap other things on the page.

We can control all margins of an element at once using the margin property, or each side individually using the equivalent longhand properties:

margin-top

margin-right

margin-bottom

margin-left

```
.box {  
  margin-top: -40px;  
  margin-right: 30px;  
  margin-bottom: 40px;  
  margin-left: 4em;  
}
```

```
<div class="container">  
  <div class="box">Change my margin.</div>  
</div>
```

Borders

The border is drawn between the margin and the padding of a box. If you are using the standard box model, the size of the border is added to the width and height of the box. If you are using the alternative box model then the size of the border makes the content box smaller as it takes up some of that available width and height.

You can set the width, style, or color of all four borders at once using the border property.

To set the properties of each side individually, you can use:

border-top

border-right

border-bottom

border-left

Borders

To set the width, style, or color of all sides, use the following:

`border-width`

`border-style`

`border-color`

To set the width, style, or color of a single side, you can use one of the more granular longhand properties:

`border-top-width`

`border-top-style`

`border-top-color`

Borders

border-right-width

border-right-style

border-right-color

border-bottom-width

border-bottom-style

border-bottom-color

border-left-width

border-left-style

border-left-color

```
.container {  
  border-top: 5px dotted green;  
  border-right: 1px solid black;  
  border-bottom: 20px double rgb(23,45,145);  
}  
  
.box {  
  border: 1px solid #333333;  
  border-top-style: dotted;  
  border-right-width: 20px;  
  border-bottom-color: hotpink;  
}
```

```
<div class="container">  
  <div class="box">Change my borders.</div>  
</div>
```

Padding

The padding sits between the border and the content area. Unlike margins, you cannot have negative amounts of padding, so the value must be 0 or a positive value. Padding is typically used to push the content away from the border. Any background applied to your element will display behind the padding.

We can control the padding on all sides of an element using the padding property, or on each side individually using the equivalent longhand properties:

padding-top

padding-right

padding-bottom

padding-left

Padding

```
.box {  
  padding-top: 0;  
  padding-right: 30px;  
  padding-bottom: 40px;  
  padding-left: 4em;  
}  
  
.container {  
  padding: 20px;  
}
```

```
<div class="container">  
  <div class="box">Change my padding.</div>  
</div>
```

Backgrounds and borders

The CSS background property is a shorthand for a number of background longhand properties.

Background colors

The background-color property defines the background color on any element in CSS. The property accepts any valid <color>. A background-color extends underneath the content and padding box of the element.

```
.box {  
  background-color: #567895;  
}  
  
h2 {  
  background-color: black;  
  color: white;  
}  
  
span {  
  background-color: rgba(255,255,255,.5);  
}
```

```
<div class="box">  
  <h2>Background Colors</h2>  
  <p>Try changing the background <span>colors</span>.</p>  
</div>
```

Backgrounds and borders

Background images

The `background-image` property enables the display of an image in the background of an element.

```
.a {  
  background-image: url(balloons.jpg);  
}  
  
.b {  
  background-image: url(star.png);  
}
```

```
<div class="wrapper">  
  <div class="box a"></div>  
  <div class="box b"></div>  
</div>
```

Backgrounds and borders

Controlling background-repeat

The background-repeat property is used to control the tiling behavior of images. The available values are:

no-repeat — stop the background from repeating altogether.

repeat-x — repeat horizontally.

repeat-y — repeat vertically.

repeat — the default; repeat in both directions.

```
.box {  
  background-image: url(star.png);  
  background-repeat: no-repeat;  
}
```

```
<div class="box"></div>
```

Backgrounds and borders

Sizing the background-image

We can use the `background-size` property, which can take length or percentage values, to size the image to fit inside the background.

You can also use keywords:

cover — the browser will make the image just large enough so that it completely covers the box area while still retaining its aspect ratio. In this case, part of the image is likely to end up outside the box.

contain — the browser will make the image the right size to fit inside the box. In this case, you may end up with gaps on either side or on the top and bottom of the image, if the aspect ratio of the image is different from that of the box.

```
.box {  
  background-image: url(balloons.jpg);  
  background-repeat: no-repeat;  
  background-size: 100px 10em;  
}
```

```
<div class="box"></div>
```


Backgrounds and borders

Positioning the background image

The background-position property allows you to choose the position in which the background image appears on the box it is applied to. This uses a coordinate system in which the top-left-hand corner of the box is (0,0), and the box is positioned along the horizontal (x) and vertical (y) axes.

The most common background-position values take two individual values — a horizontal value followed by a vertical value.

You can use keywords such as top and right

```
.box {  
  background-image: url(star.png);  
  background-repeat: no-repeat;  
  background-position: top center;  
}
```

```
.box {  
  background-image: url(star.png);  
  background-repeat: no-repeat;  
  background-position: 20px 10%;  
}
```

Backgrounds and borders

Positioning the background image

The background-position property allows you to choose the position in which the background image appears on the box it is applied to. This uses a coordinate system in which the top-left-hand corner of the box is (0,0), and the box is positioned along the horizontal (x) and vertical (y) axes.

The most common background-position values take two individual values — a horizontal value followed by a vertical value.

You can use keywords such as top and right

```
.box {  
  background-image: url(star.png);  
  background-repeat: no-repeat;  
  background-position: top center;  
}
```

```
.box {  
  background-image: url(star.png);  
  background-repeat: no-repeat;  
  background-position: 20px 10%;  
}
```

```
.box {  
  background-image: url(star.png);  
  background-repeat: no-repeat;  
  background-position: 20px top;  
}
```

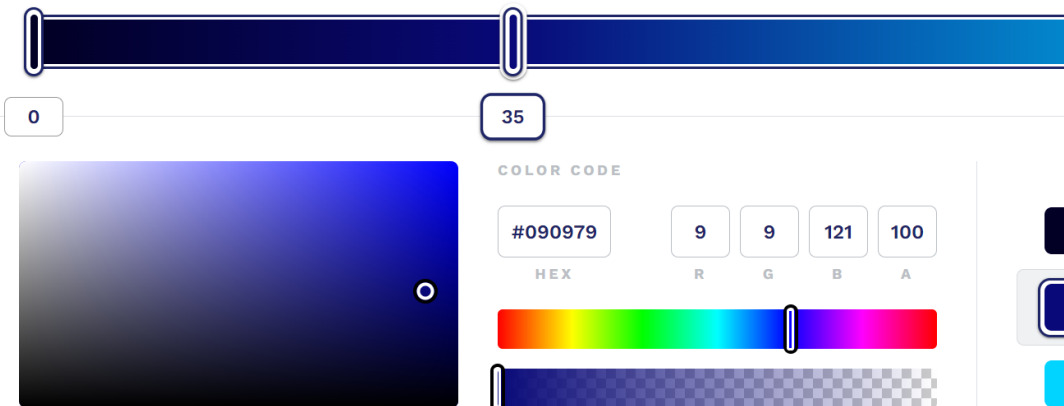
Backgrounds and borders

Gradient background

A gradient — when used for a background — acts just like an image and is also set by using the background-image property.

We can use one of the many CSS Gradient Generators available on the web.

Eg: <https://cssgradient.io/>



Copy code from CSS website from linear gradient / radial gradient

```
.a {  
  background-image: linear-gradient(105deg, rgba(0,249,255,1)  
39%, rgba(51,56,57,1) 96%);  
}  
  
.b {  
  background-image: radial-gradient(circle, rgba(0,249,255,1)  
39%, rgba(51,56,57,1) 96%);  
  background-size: 100px 50px;  
}
```

```
<div class="wrapper">  
  <div class="box a"></div>  
  <div class="box b"></div>  
</div>
```



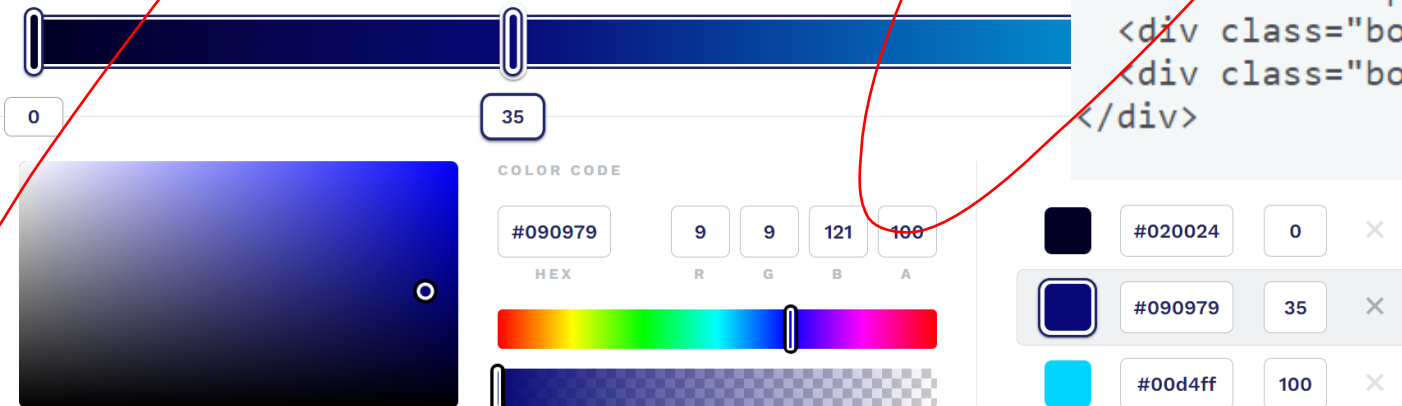
Backgrounds and borders

Gradient background

A gradient — when used for a background — acts just like an image and is also set by using the background-image property.

We can use one of the many CSS Gradient Generators available on the web.

Eg: <https://cssgradient.io/>



```
.a {  
  background-image: linear-gradient(105deg, rgba(0,249,255,1)  
39%, rgba(51,56,57,1) 96%);  
}  
  
.b {  
  background-image: radial-gradient(circle, rgba(0,249,255,1)  
39%, rgba(51,56,57,1) 96%);  
  background-size: 100px 50px;  
}
```

```
<div class="wrapper">  
  <div class="box a"></div>  
  <div class="box b"></div>  
</div>
```

Borders - Styling

*/*we can target one edge of the box*/*

```
.box {  
  border-top: 1px solid black;  
}
```

*/*The individual properties for these shorthands would be:*/*

```
.box {  
  border-width: 1px;  
  border-style: solid;  
  border-color: black;  
}
```

/ And for the longhands: */*

```
.box {  
  border-top-width: 1px;  
  border-top-style: solid;  
  border-top-color: black;  
}
```

/ Rounded corners */*

```
.box {  
  border-radius: 10px;  
}
```

Borders - Styling

```
/*Rounding corners on a box is achieved by using the border-radius property*/
```

```
/* make all four corners of a box have a 10px radius */
```

```
.box {  
  border-radius: 10px;  
}
```

```
/* make the top right corner have a horizontal radius of 1em, and a vertical radius of 10% */
```

```
.box {  
  border-top-right-radius: 1em 10%;  
}
```

```
.box {  
  border: 10px solid rebeccapurple;  
  border-radius: 1em;  
  border-top-right-radius: 10% 30%;  
}
```

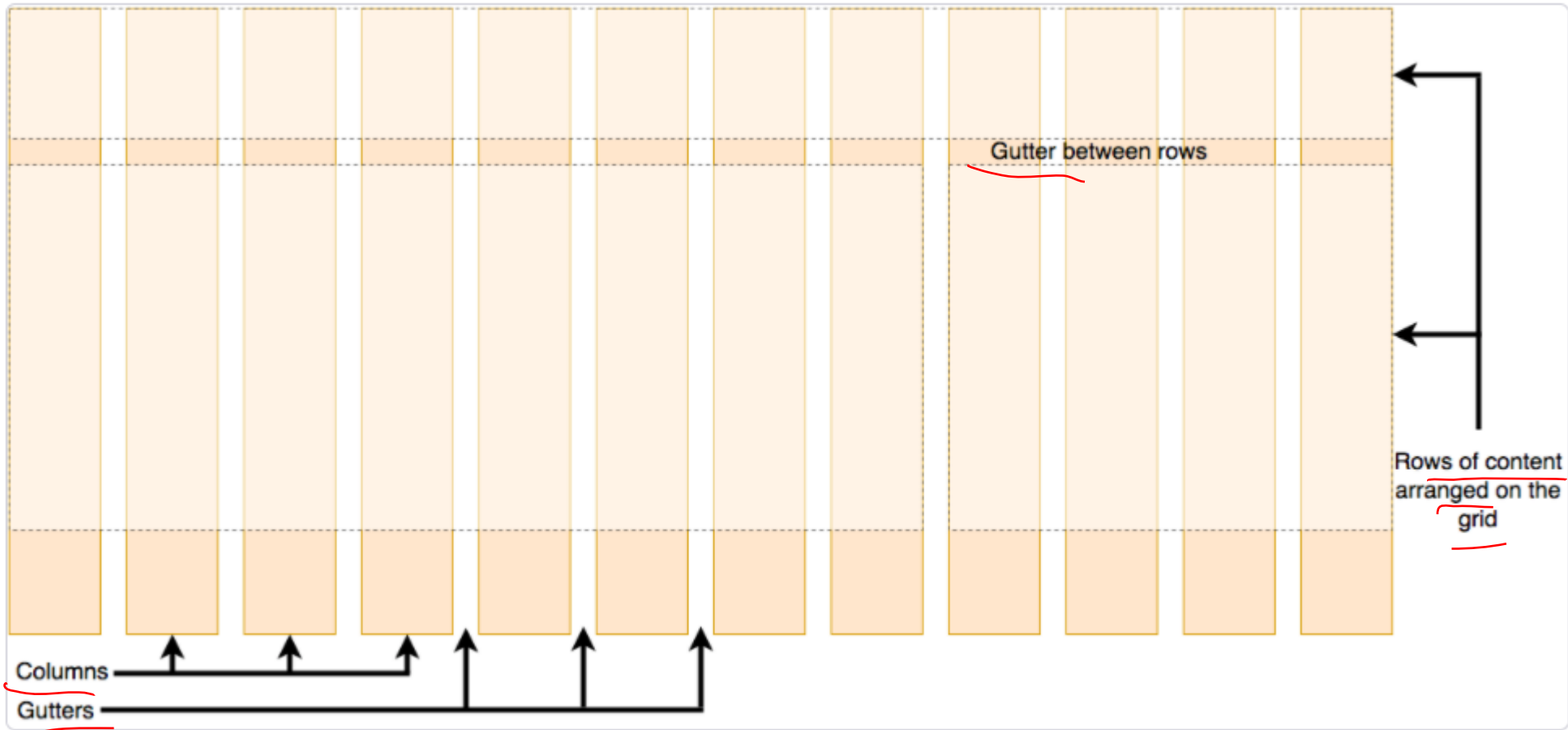
CSS Layout - Grids

CSS Grid Layout is a two-dimensional layout system for the web. It lets you lay content out in rows and columns.

A grid is a collection of horizontal and vertical lines creating a pattern against which we can line up our design elements. They help us to create layouts in which our elements won't jump around or change width as we move from page to page, providing greater consistency on our websites.

A grid will typically have columns, rows, and then gaps between each row and column. The gaps are commonly referred to as gutters.

CSS Layout - Grids



CSS Layout - Grids

Defining a grid:

To define a grid we use the `grid` value of the `display` property. As with Flexbox, this enables Grid Layout;

```
.container {  
  display: grid;  
}
```