CSS @ GLANCE

Structure or Syntax of CSS

- ✓ The syntax of CSS is slightly different from that of an HTML.
- ✓ CSS uses **(curly braces { }), (colons :)** and **(semicolon ;)**.

Syntax:

```
selector
{
        property : value;
        property : value;
        |                       |
        property : value;
}
```

Importance of semicolon

```
p{text-align:center background-color:tomato ;color:cyan}
```

• The code has a syntax error (missing semicolon after text-align:center).
• Because of the error, the browser ignores the entire line until the next semicolon.
• Only the last valid property works:
Color:cyan;
• Output: The <p> text becomes cyan color only , no background-color and align applied

Types of CSS

Type of CSS	Description	Where It Is Written	Syntax (Example)
1. Inline CSS	Used to style a **single HTML element** directly. It has the **highest priority**.	Inside the HTML tag using the style attribute.	Heading
2. Internal (Embedded) CSS	Used to style elements **within the same HTML page**.	Inside the <style> tag in the <head> section of the HTML document.	<head><style> p { color: red; font-size: 18px; } </style></head>
3. External CSS	Used to apply styles to **multiple web pages** using a separate .css file.	In an external stylesheet linked using the <link> tag in the <head> section.	<head><link rel="stylesheet" href="style.css"></head>*(style.css file contains CSS rules)*

Various CSS Selectors

Selector Type	Description	Syntax	Example
Element Selector	Selects all elements of a specific type/tag. Lowest specificity (except universal).	**element**	p { color: black; }
ID Selector	Selects a single element with a specific id attribute. Very high specificity.	**#id**	#header { color: red; }
Class Selector	Selects elements with a specific class. Medium specificity.	**.class**	.menu { font-size: 16px; }
Attribute Selector	Selects elements based on an attribute or attribute value.	**[attr], [attr=value]**	input[type="text"] { border: 1px solid #ccc; }
Universal Selector	Selects all elements. Lowest specificity.	*****	* { margin: 0; padding: 0; }
Descendant Selector	Selects elements that are descendants (any level) of a specified ancestor.	**ancestor descendant**	div p { color: blue; }
Child Selector	Selects elements that are direct children of a specified parent.	**parent > child**	ul > li { list-style: none; }
Grouping Selector	Groups multiple selectors and applies the same styles.	**selector1, selector2**	h1, h2, h3 { font-family: Arial; }

CSS Specificity (Priority Order)

When multiple CSS rules target the same element, the browser decides which one to apply based on **specificity** (priority):

Priority Level	Selector Type	Example
1 (Highest)	Inline Styles	Heading
2	IDs	#title { color: red; }
3	Classes, pseudo-classes, attribute selectors	.title { color: green; }, :hover, [type='text']
4 (Lowest)	Elements and pseudo-elements	h1 { color: blue; }, ::before

Text Properties

These properties are used to **style, format, and control the appearance** of text on a webpage, including alignment, spacing, transformation, decoration, font type, and size.

Text Properties

Property	Description	Syntax	Possible Values
color	Sets the color of the text	color: color;	Named colors (e.g., red), HEX (#ff0000), RGB (rgb(255,0,0)), RGBA
text-align	Aligns text inside an element	text-align: value;	left, right, center, justify, start, end
text-indent	Adds indentation to the first line	text-indent: length;	Any length (px, em, %)
text-transform	Controls text capitalization	text-transform: value;	none, capitalize, uppercase, lowercase
text-decoration	Adds decoration like underline or line-through	text-decoration: value;	none, underline, overline, line-through, underline overline
letter-spacing	Controls space between letters	letter-spacing: length;	Normal (normal), or custom spacing (px, em). Negative values are allowed.
word-spacing	Controls space between words	word-spacing: length;	Normal (normal), or custom spacing (px, em). Negative values are allowed.
line-height	Sets space between lines	line-height: value;	normal, number (e.g., 1.5), length (px, em), %. Negative values are **not** allowed.
text-shadow	Adds shadow to text	text-shadow: h-shadow v-shadow blur color;	none or values like 2px 2px 5px gray (can add multiple shadows separated by commas)

Font Properties

Property	Description	Syntax	Example	Possible Values
font-family	Specifies the font type	font-family: "FontName", fallback;	font-family: "Arial", sans-serif;	Any font name: "Arial", "Times New Roman", "Courier New", generic families: serif, sans-serif, monospace, cursive, fantasy
font-size	Sets the font size	font-size: size;	font-size: 20px;	Length units: px, em, rem, %; Keywords: xx-small, x-small, small, medium, large, x-large, xx-large, smaller, larger
font-style	Defines style of text	font-style: value;	font-style: italic;	normal, italic
font-weight	Sets font thickness	font-weight: value;	font-weight: bold;	normal, bold, bolder, lighter, 100, 200, 300, 400, 500, 600, 700, 800, 900
font-variant	Displays text in small-caps	font-variant: value;	font-variant: small-caps;	normal, small-caps
font (shorthand)	Sets all font properties in one line	font: [style] [variant] [weight] [size]/[line-height] [family];	font: italic small-caps bold 16px/1.5 "Times New Roman", serif;	Combines all above properties in one; values are the same as individual properties: <style> <variant> <weight> <size>/<line-height> <family>

Google Fonts

How to Embed Google Fonts

Method 1: Using <link> tag (Recommended)

Method 2: Using @import in CSS

CSS Borders

A **border** is a line that wraps around an HTML element's **content and padding**. It visually separates elements or highlights them on a web page.

Property	Description	Possible Values	Syntax
border-style	Defines the style of the border (Must to Define)	none, solid, dashed, dotted, double **Note:** Multiple values allow to get different border side wise	border-style: solid border-style: solid dashed double dotted
border-width	Sets the thickness of the border	length (px, em, rem)	border-width: 10px
border-color	Sets the color of the border	Named colors (red), hex (#FF0000), RGB (rgb(255,0,0))	border-color: tomato
border (shorthand)	Combines style, width, color in one line	border: 2px solid red;	border: 2px solid red;
border-top	Sets border properties for the top side	border-top: 3px dashed blue;	border-top: 3px dashed blue;
border-right	Sets border properties for the right side	border-right: 2px solid green;	border-right: 2px solid green;
border-bottom	Sets border properties for the bottom side	border-bottom: 4px dotted orange;	border-bottom: 4px dotted orange;
border-left	Sets border properties for the left side	border-left: 5px double purple;	border-left: 5px double purple;

Box Properties

Property	Description	Possible Values	Syntax / Examples
margin	Creates **space outside** the element (outside border). **Used separate elements** on the page.	• auto → browser calculates margin (useful for centering) • length → px, em, %, negative values allowed	**margin: 10px** **Shorthand examples:** • 4 values → margin: 25px 50px 75px 100px; → top=25px, right=50px, bottom=75px, left=100px • 3 values → margin: 25px 50px 75px; → top=25px, right/left=50px, bottom=75px • 2 values → margin: 25px 50px; → top/bottom=25px, right/left=50px • 1 value → margin: 25px; → all sides=25px
padding	Creates **space inside** the element (between content & border).	length → px, em, %	**padding:10px** **Shorthand examples:** • 4 values → padding: 25px 50px 75px 100px; → top=25px, right=50px, bottom=75px, left=100px • 3 values → padding: 25px 50px 75px; → top=25px, right/left=50px, bottom=75px • 2 values → padding: 25px 50px; → top/bottom=25px, right/left=50px • 1 value → padding: 25px; → all sides=25px
width	Specifies the **width** of an element's content area.	auto, length (px, em, %),	width : 50px
height	Specifies the **height** of an element's content area.	auto, length (px, em, %),	height:50px
box-sizing	Defines **how width and height are calculated** — whether padding and borders are included. Used to **control layout behavior** when adding padding/borders.	• content-box (default): width/height exclude padding & border. • border-box: width/height include padding & border.	**box-sizing : content-box** (height and width increase by increasing padding and border px) **box-sizing : border-box** (height and width will remain same)
box-shadow	Adds **shadow effects** around an element's frame. To **add depth**, **hover effects**, or **visual highlights**.	box-shadow: h-offset v-offset blur spread color;	**Syntax:** box-shadow: h-offset v-offset blur spread color; **Example:** box-shadow: 2px 2px 5px gray;

CSS background properties

Property	Description	Key Values / Notes	Syntax
background-color	Sets the background color of an element	Any valid color (red, #ffc0cb, rgb(255,0,0))	background-color: color;
background-image	Sets an image as the background	Use image URL;	background-image: url("image.jpg");
background-repeat	Controls how background image repeats	repeat, repeat-x, repeat-y, no-repeat default repeats if not specified	background-repeat: repeat;
background-position	Sets the starting position of a background image	top, bottom, left, right, center, x% y%	background-position: top;
background-size	Specifies the size of the background image	auto, cover, contain, width height	background-size: auto;
background-attachment	Sets whether background scrolls with content	scroll, fixed, local	background-attachment: scroll;
background-clip	Determines how far the background extends	border-box, padding-box, content-box	background-clip: border-box;
background (shorthand)	Combines color, image, repeat, position, size, attachment	Example: background: pink url("scenary.jfif") no-repeat top right cover fixed;	background: [color] [image] [repeat] [position] [size] [attachment];

Pseudo classes

Pseudo-class	What It Does	Syntax
:hover	Changes color when mouse hovers	Selector:hover{prop:value} p:hover{color:blue;}
:active	Changes color when clicked	a:active{prop:value}
:focus	Highlights input field when clicked	input:focus{prop:value}
:link / :visited	Colors for unvisited/visited links	a:link{prop:value}
:nth-child(n)	Targets the n^{th} list item	li:nth-child(2){prop:value}
:not(.special)	Styles all list items except .special	p:not(.classname){prop:value}

Pseudo Elements

Pseudo-element	Description	Syntax	Example	Result / Effect
::first-line	Styles the **first line** of a block-level element.	p::first-line { property: value; }	p::first-line { color: red; font-weight: bold; }	First line of the paragraph appears red and bold.
::first-letter	Styles the **first letter** of a block-level element.	p::first-letter { property: value; }	p::first-letter { font-size: 50px; color: red; }	First letter of the paragraph becomes large and red.
::before	Inserts content **before** an element's actual content.	selector::before { content: "..."; }	p::before { content: "Note: "; color: red; }	Adds "Note:" before each paragraph.
::after	Inserts content **after** an element's actual content.	selector::after { content: "..."; }	p::after { content: "✔"; color: green; }	Adds a green checkmark after the paragraph.
::marker	Styles the **marker (bullet or number)** of list items.	li::marker { property: value; }	li::marker { color: red; font-size: 20px; }	List bullets or numbers appear red and large.
::selection	Styles the **highlighted text** selected by the user.	::selection { property: value; }	::selection { background: blueviolet; color: white; }	Selected text appears white on a blue-violet background.

Display Property

Display Type	Description	Behavior / Use Case	Visual Behavior	Syntax
inline	Displays elements **in a line**, without starting on a new line.	Does **not accept width/height**. Common for , <a>, .	Elements sit **side by side** in a single line.	display: inline
block	Displays element as a **block**, starting on a new line.	Takes **full width** available and allows **width/height** to be set.	Each element appears **on a new line**.	display: block
inline-block	Combines features of **inline** and **block**.	Appears **inline**, but allows **width and height**.	Boxes are **side by side**, but **size-controllable**.	display: inline-block
none	**Hides** the element completely (removed from layout).	Element takes **no space** on the page.	The element is **invisible** and **does not occupy space**.	display: none
flex	Displays element as a **flex container**.	Allows flexible alignment and distribution of child elements.	Items are **arranged in a row** (or column) with flexible spacing.	display: flex
grid	Displays element as a **grid container**.	Divides layout into **rows and columns** for advanced control.	Elements are placed in a **grid layout** (rows and columns).	display:grid

Flexbox

Property	Description	Possible Values	Syntax / Example
display: flex	Defines a flex container to arrange items flexibly.	flex, inline-flex	div { display: flex; }
flex-direction	Defines the direction of flex items.	row (default), row-reverse, column, column-reverse	flex-direction: row;
justify-content	Aligns items horizontally (along the main axis).	flex-start, flex-end, center, space-between, space-around, space-evenly	justify-content: space-between;
align-items	Aligns items vertically (along the cross axis).	stretch (default), flex-start, flex-end, center	align-items: center;
flex-wrap	Determines whether flex items wrap onto multiple lines.	nowrap (default), wrap, wrap-reverse	flex-wrap: wrap;
gap	Defines the space between flex items.	Any CSS length unit (px, em, %)	gap: 15px;
order	Specifies the display order of flex items.	Integer values (0 default, can be positive or negative)	order: 2;
flex *(shorthand)*	Sets how an item grows, shrinks, and defines its base size.	flex: grow shrink basis; e.g., flex: 1 0 100px;	flex: 1; or flex: 1 0 200px;

Grid Layout Properties

Property	Description	Possible Values	Default Value	Syntax / Example
grid-template-columns	Defines the number and width of columns in a grid layout.	Fixed units (px, em, %)	none	grid-template-columns: 200px 1fr 2fr;
grid-template-rows	Defines the number and height of rows in a grid layout.	Fixed units (px, em, %),	none	grid-template-rows: 100px auto 100px;
gap (grid-gap)	Defines the space between rows and columns.	Any CSS length (px, em, %)	0	gap: 10px; or gap: 20px 40px;
grid-column	Specifies the horizontal position of an item within the grid columns (start / end lines).	<start-line> / <end-line>	auto	grid-column: 1 / 3; (spans across 2 columns)
grid-row	Specifies the vertical position of an item within the grid rows (start / end lines).	<start-line> / <end-line>	auto	grid-row: 1 / 2; (spans across 1 row)

CSS Positioning Properties

Property	Description	Possible Values	Default Value	Syntax / Example
position	Defines how an element is positioned in the document.	static, relative, absolute, fixed, sticky	static	position: absolute;
top	Distance between element and top edge of container.	Any CSS length (px, %, auto)	auto	top: 20px;
right	Distance between element and right edge of container.	Any CSS length (px, %, auto)	auto	right: 10px;
bottom	Distance between element and bottom edge of container.	Any CSS length (px, %, auto)	auto	bottom: 15px;
left	Distance between element and left edge of container.	Any CSS length (px, %, auto)	auto	left: 30px;
z-index	Controls stack order of overlapping elements.	Integer (auto, positive or negative values)	auto	z-index: 2; (Higher = on top)

- ✓ **static: Default, element follows normal document flow.**
- ✓ **relative: Moved relative to its normal position. (Shift top/bottom.. from it's aligned position)**
- ✓ **absolute: Positioned relative to the nearest positioned ancestor**
- ✓ **fixed: Stays fixed in place even when scrolling.**
- ✓ **z-index: Controls which element appears on top of others.**

CSS Multiple Column

Property	Syntax	Example	What it does
column-count	column-count:number;	column-count:4;	Specifies the number of columns an element should be divided into
column-gap	column-gap:number;	column-gap:20px;	Specifies the gap between the columns
column-rule-style	column-rule-style:value	column-rule-style:dashed;	Specifies the style of the rule between columns. (**none, dotted, dashed, solid, double) default: none**
column-rule-color	column-rule-color:color	column-rule-color:rgb(219, 12, 47);	Specifies the color of the rule between columns
column-rule-width	column-rule-width:value	column-rule-width:7px;	Specifies the width of the rule between columns
Shorthand column-rule	Column-rule: width style color	Column-rule:1px solid cyan	Specify width,style and color of rule

CSS style images

Property	Syntax	Example	What it does
border-radius	border-radius:radius;	border-radius: 20px;	Sets the corner rounding for **all four corners**.
border-top-left-radius	border-top-left-radius: radius;	border-top-left-radius: 15px;	Rounds only the **top-left** corner.
border-top-right-radius	border-top-right-radius;	border-top-right-radius: **15px**;	Rounds only the **top-right** corner.
border-bottom-right-radius	border-bottom-right-radius: radius;	border-bottom-right-radius: 15px;	Rounds only the **bottom-right** corner.
border-bottom-left-radius	border-bottom-left-radius: radius;	border-bottom-left-radius: 15px;	Rounds only the **bottom-left** corner.
padding	padding:value	padding:5px	The border property and padding property are used to make a thumbnail image.
opacity	opacity:value	opacity:0.3	To make an image transparent, we have to use the opacity property. The value of this property lies between **(transparent) 0.0 to 1.0 (Solid).**
Responsive	max-width: %; height:auto	max-width:100%; height:auto	The image becomes flexible — it scales down when the screen is smaller, but won't stretch beyond its natural size.

2D Transform

Property	Syntax	Example	What it does
translate(x, y)	transform: translate(x, y);	transform: translate(50px, 30px);	Moves an element 50px right and 30px down.
translateX(n)	transform: translateX(n);	transform: translateX(100px);	Moves an element 100px horizontally (right).
translateY(n)	transform: translateY(n);	transform: translateY(-50px);	Moves an element 50px upward.
rotate(angle)	transform: rotate(angle);	transform: rotate(45deg);	Rotates the element 45° clockwise.
scale(x, y)	transform: scale(x, y);	transform: scale(1.5, 0.8);	Increases width 1.5× and decreases height to 0.8×.
scaleX(n)	transform: scaleX(n);	transform: scaleX(2);	Doubles the element's width.
scaleY(n)	transform: scaleY(n);	transform: scaleY(0.5);	Reduces the element's height by half.
skew(x, y)	transform: skew(x, y);	transform: skew(20deg, 10deg);	Slants an element 20° horizontally and 10° vertically.
skewX(angle)	transform: skewX(angle);	transform: skewX(20deg);	Skews an element horizontally by 20°.
skewY(angle)	transform: skewY(angle);	transform: skewY(20deg);	Skews an element vertically by 20°.

Note:

- The transform: scaleX(-1) property is used to flip the image horizontally.
- The transform: scaleY(-1) - mirror image vertically.
- The transform: scale(-1) property create a mirror image vertically as well as horizontally.

Use for multiple transform actions

transform: rotate(30deg) scale(1.5) translate(100px, 100px) skew(20deg,20deg);

CSS Transitions Properties

Property	Syntax	Example	What it does
transition-property	transition-property: all | property | property1, property2, ...;	transition-property: background-color, transform;	Specifies which CSS properties will be animated during the transition.
transition-duration	transition-duration: time; (Must to mention)	transition-duration: 0.5s;	Specifies how long the transition takes to complete.
transition-delay	transition-delay: time;	transition-delay: 0.3s;	Specifies how long to wait before starting the transition.
transition-timing-function	transition-timing-function: linear | ease | ease-in | ease-out | ease-in-out;	transition-timing-function: ease-in-out;	Controls the speed curve of the transition — how the animation progresses over time.
transition (shorthand)	transition: property duration timing-function delay;	transition: background-color 0.5s ease-in-out 0.2s;	Sets all transition properties in a single line (shorthand form).

Note: If the duration part is not specified, the transition will have no effect (default duration is 0s).

You can apply *multiple transitions* to a single element by separating them with commas. Each transition can have its own **property**, **duration**, and **delay** — for example,
transition: width 2s 2s, height 2s 5s; means the width starts changing after 2 seconds, and the height starts after 5 seconds, both smoothly over 2 seconds each.

Transition Shorthand Property Breakdown

Property	Required in shorthand?	Default value
transition-property	No	all
transition-duration	Yes	0s
transition-timing-function	No	ease
transition-delay	No	0s

CSS Animation

The @keyframes Rule

When you specify CSS styles inside the @keyframes rule, the animation will gradually change from the current style to the new style at certain times.
To get an animation to work, you must bind the animation to an element.

Note: In CSS keyframe, **from** = **0%** (start) and **to** = **100%** (end); using **percentages** lets you define multiple stages within the animation.

@keyframes animation-name { from { ... } to { ... } }
or @keyframes animation-name { 0% { ... } 100% { ... } }

Inside {…}, write the **CSS properties you want to animate**, such as transform, opacity, background-color, width, etc.

CSS Animation Properties

Property	Syntax	Example	What it does
animation-name	animation-name: keyframename;	animation-name: moveBox;	Specifies the name of the @keyframes to link to the element.
animation-duration	animation-duration: time;	animation-duration: 2s;	Specifies the time it takes for one animation cycle to complete.
animation-delay	animation-delay: time;	animation-delay: 1s;	Specifies the delay before the animation starts.
animation-timing-function	linear | ease | ease-in | ease-out | ease-in-out	animation-timing-function: ease-in-out;	Controls the speed curve of the animation over time.
animation-iteration-count	number | infinite	animation-iteration-count: infinite;	Specifies how many times the animation should repeat.
animation-direction	normal | reverse | alternate | alternate-reverse	animation-direction: alternate;	Specifies whether animation should run forward, backward, or alternate directions.
animation (shorthand)	animation: name duration timing-function delay iteration-count direction;	animation: example 3s ease-in-out 1s infinite alternate	Sets all animation properties in a single line (shorthand form).

CSS Gradient

Linear gradient

type	Syntax	Example	What it does
Linear Gradient (Default – Top to Bottom)	**background-image:** linear-gradient(color1, color2);	background-image: linear-gradient(purple, yellow);	Creates a smooth color transition from top to bottom (default direction).
Bottom to Top	background-image: linear-gradient(to top, color1, color2);	background-image: linear-gradient(to top, purple, yellow);	Creates a gradient starting from bottom to top.
Left to Right	background-image: linear-gradient(to right, color1, color2);	background-image: linear-gradient(to right, purple, yellow);	Creates a gradient moving from left to right.
Right to Left	background-image: linear-gradient(to left, color1, color2);	background-image: linear-gradient(to left, purple, yellow);	Creates a gradient moving from right to left.
Diagonal Gradient (to top left)	background-image: linear-gradient(to top left, color1, color2);	background-image: linear-gradient(to top left, purple, yellow);	Creates a diagonal gradient moving from bottom right to top left.
Diagonal Gradient (to bottom right)	background-image: linear-gradient(to bottom right, color1, color2);	background-image: linear-gradient(to bottom right, purple, yellow);	Creates a diagonal gradient from top left to bottom right.
Angle-based Gradient	background-image: linear-gradient(angle, color1, color2);	background-image: linear-gradient(90deg, pink, lightblue);	Uses degrees (0°, 90°, 180°,-90° etc.) to control gradient direction.
Evenly spaced multiple colors	linear-gradient(color1, color2, color3, ...)	linear-gradient(red, yellow, green);	Colors are evenly spaced between start and end points.
Non-evenly spaced colors	linear-gradient(color1 %, color2 %, ...)	linear-gradient(pink 20%, lightgreen 30%, lightblue 50%);	Uses percentage values to control how far each color extends.
Rainbow Gradient Example	---	linear-gradient(to right, red, orange, yellow, green, blue, indigo, violet);	Creates a rainbow-style horizontal gradient.

Radial gradient

type	Syntax	Example	What it does
Basic Radial Gradient	background-image: radial-gradient(color1, color2, color3);	background-image: radial-gradient(red, yellow, blue);	Creates a circular gradient radiating outward from the center.
Different Color Stops	background-image: radial-gradient(color1 %, color2 %, color3 %);	background-image: radial-gradient(yellow 10%, red 25%, pink 40%);	Defines uneven spacing for color transitions.
Ellipse (default)	background-image: radial-gradient(ellipse, color1, color2);	background-image: radial-gradient(purple, yellow, pink);	Creates an oval-shaped gradient that fills the container.
Circle Shape	background-image: radial-gradient(circle, color1, color2);	background-image: radial-gradient(circle, blue, yellow, pink);	Creates a perfectly circular gradient.

Conic Gradients

type	Syntax	Example	What it does
Basic Conic Gradient	background-image: conic-gradient(color1, color2, color3);	background-image: conic-gradient(red, yellow, blue);	Creates a circular gradient where colors rotate around the center point.
Using Degrees for Each Color	background-image: conic-gradient(color1 deg, color2 deg, ...);	background-image: conic-gradient(red 45deg, yellow 90deg, green 180deg);	Defines starting angles for each color transition.
Defined Start and End Degrees (Pie Chart Style)	background-image: conic-gradient(color1 start end, color2 start end, ...);	background-image: conic-gradient(red 0deg 90deg, yellow 90deg 180deg, blue 270deg);	Creates solid color segments similar to pie chart slices.
Rounded Pie Chart	border-radius: 50%;	background-image: conic-gradient(red, yellow, green, blue); border-radius: 50%;	Turns conic gradient into a circular pie chart.
With “from” Angle	background-image: conic-gradient(from angle, color1, color2, ...);	background-image: conic-gradient(from 90deg, red, yellow, green);	Rotates where the gradient starts based on the specified angle.

CSS Variable

CSS variables store reusable values that can be updated from one place.
They can be global (defined in :root for whole document) or local (defined inside a selector).
Values are accessed using the var() function, e.g. color: var(--b);.

- Case sensitive and local variable overrides global vaiables.

Syntax:

Define variable globally (in :root)
:root{--b: blue; }
Use of variable
P{ color:var(--b) }
All the elements of the document can use the variable.

Define variable locally (Inside selectors) and use it
P{ --b: blue; color:var(--b) }
It has a scope only for the p element.

CSS Media Queries

To make your webpage **responsive** (adapt its layout properly to different screen sizes like phones, tablets, and desktops), you **must** include this line **inside the <head> section** of your HTML document:

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

Attribute / Setting	Meaning
name="viewport"	Tells the browser that this meta tag controls the **viewport** (visible area of the web page).
content="width=device-width"	Sets the page width to match the **device's screen width** (so it doesn't zoom out to a fixed desktop width).
initial-scale=1.0	Sets the **initial zoom level** when the page is first loaded (1.0 = 100% zoom).

Category	Property / Syntax	Example Description / Usage

Basic Syntax	@media not|only mediatype and (mediafeature) and|or (mediafeature) {
	/* CSS rules go here */
	}
		@media screen and (max-width: 600px) { ... }

Media Types	all, screen	@media screen { ... }
Media feature	(orientation: portrait) Or (orientation: landscape) Or (max-width:500px) Or (min-width:500px)	@media (orientation: portrait) { ... } Adjusts styles based on device orientation.
Combining Conditions	and, or, not	@media screen and (min-width: 600px) and (orientation: landscape) Combines multiple media features for precise control.
Range Example	@media (min-width: 500px) and (max-width: 700px)	@media (min-width: 500px) and (max-width: 700px) { body { background: lightblue; } } Styles apply only between 500px and 700px viewport widths.
Negation Example	@media not (orientation: landscape)	@media not (orientation: landscape) { body { background: pink; } } Excludes landscape orientation (applies to portrait).
'only' Keyword Example	@media only screen and (max-width: 600px)	Ensures old browsers ignore unsupported queries.
Device Adaptation Example	@media screen and (max-width: 400px)	body { background: orange; color: white; } Styles for small screens like phones.
Tablet View Example	@media screen and (min-width: 401px) and (max-width: 1024px)	body { background: pink; color: blue; } Styles for tablets or medium devices.