Chapter 4

CSS stands for **Cascading Style Sheet**.

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;
}
```

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.	<h1 style="color:blue; font-size:25px;">Heading</h1>
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]**	[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)

Priority Level	Selector Type	Example
1 (Highest)	Inline Styles	<h1 style="color:pink;">Heading</h1>
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

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	Why to Use	Possible Values / Syntax
border-style	Defines the style of the border	To give different visual effects like solid, dashed, dotted, etc.	none, solid, dashed, dotted, double
border-width	Sets the thickness of the border	To adjust border size	length (px, em, rem)
border-color	Sets the color of the border	To visually match design or highlight elements	Named colors (red), hex (#FF0000), RGB (rgb(255,0,0))
border (shorthand)	Combines style, width, color in one line	To quickly define a complete border	border: 2px solid red;
border-top	Sets border properties for the top side	To style only the top edge	border-top: 3px dashed blue;
border-right	Sets border properties for the right side	To style only the right edge	border-right: 2px solid green;
border-bottom	Sets border properties for the bottom side	To style only the bottom edge	border-bottom: 4px dotted orange;
border-left	Sets border properties for the left side	To style only the left edge	border-left: 5px double purple;

Box Properties

Property	Description	Why to Use	Possible Values / Syntax / Examples
margin	Creates **space outside** the element (outside border).	To **separate elements** on the page.	**Values:** • auto → browser calculates margin (useful for centering) • length → px, em, %, negative values allowed **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).	To **add inner spacing** inside the element.	**Values:** • length → px, em, % **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.	To **control horizontal size** of the element.	**Values:** auto, length (px, em, %),
height	Specifies the **height** of an element's content area.	To **control vertical size** of the element.	**Values:** auto, length (px, em, %),
box-sizing	Defines **how width and height are calculated** whether padding and borders are included.	To **control layout behavior** when adding padding/borders.	**Values:** • content-box (default): width/height exclude padding & border. • border-box: width/height include padding & border.
box-shadow	Adds **shadow effects** around an element's frame.	To **add depth**, **hover effects**, or **visual highlights**.	**Syntax:** box-shadow: h-offset v-offset blur spread color; **Example:** box-shadow: 2px 2px 5px gray;

CSS background properties

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

Pseudo classes

Pseudo-class	What It Does
:hover	Changes color when mouse hovers
:active	Changes color when clicked
:focus	Highlights input field when clicked
:link / :visited	Colors for unvisited/visited links
:nth-child(2)	Targets the 2nd list item
:not(.special)	Styles all list items except .special

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

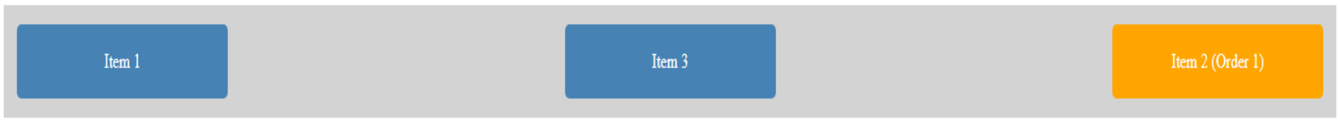
Display Type	Description	Behavior / Use Case	Visual Behavior
inline	Displays elements **in a line**, without starting on a new line.	Does **not accept width/height**. Common for <span>, <a>, <strong>.	Elements sit **side by side** in a single line.
block	Displays element as a **block**, starting on a new line.	Takes **full width** available and allows **width/height** to be set.	Each element appears **on a new line**.
inline-block	Combines features of **inline** and **block**.	Appears **inline**, but allows **width and height**.	Boxes are **side by side**, but **size-controllable**.
none	**Hides** the element completely (removed from layout).	Element takes **no space** on the page.	The element is **invisible** and **does not occupy space**.
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.
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).

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 Example

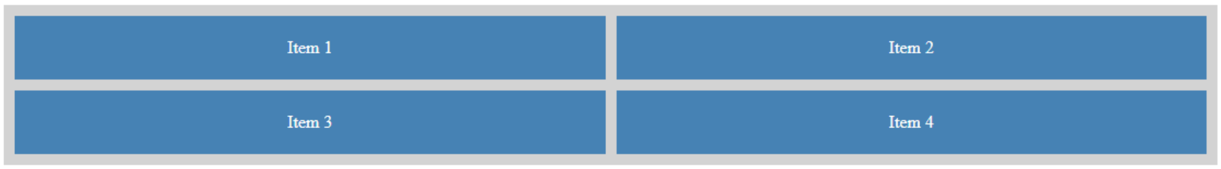
```
<head>
<style>
.container {
  display: flex;               /* Creates a flex container. Change values and check effects */
  flex-direction: row;          /* Arranges items in a row (default). Change values and check effects */
  justify-content: space-between; /* Spaces items horizontally. Change values and check effects */
  align-items: center;          /* Aligns items vertically in the center. Change values and check effects */
  flex-wrap: wrap;              /* Wraps items to next line if space runs out */
  gap: 15px;                   /* Adds space between items */
  background-color: lightgray;
  padding: 15px;
}
.item {
  background-color: steelblue;
  color: white;
  padding: 20px;
  text-align: center;
  border-radius: 5px;
  width: 200px;
}
.item:nth-child(2) {
  order: 1;            /* This will move the item to appear after others. Change values and check
effects(POSITIVE,NEGATIVE,0) */
/*  flex: 1 1 100px;  */      /* Remove comments from this line to check effect. flex-grow: 1, flex-shrink: 1,
flex-basis: 100px */
  background-color: orange;
}
</style>
</head>
<body>
<div class="container">
  <div class="item">Item 1</div>
  <div class="item">Item 2 (Order 1)</div>
  <div class="item">Item 3</div>
</div>
</body>
```



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	Defines the space between rows and columns.	Any CSS length (px, em, %)	0	gap: 10px; or gap: 20px 40px;

```
<head>
<style>
.grid1 {
  display: grid;
  grid-template-columns: repeat(2,1fr);
  grid-template-rows: auto;
  gap: 10px;
  background-color: lightgray;
  padding: 10px;
}
.item {
  background-color: steelblue;
  color: white;
  text-align: center;
  padding: 20px;
}
</style></head>
<body>
<div class="grid1">
  <div class="item">Item 1</div>
  <div class="item">Item 2</div>
  <div class="item">Item 3</div>
  <div class="item">Item 4</div>
</div>
</body>
```



CSS Positioning Properties

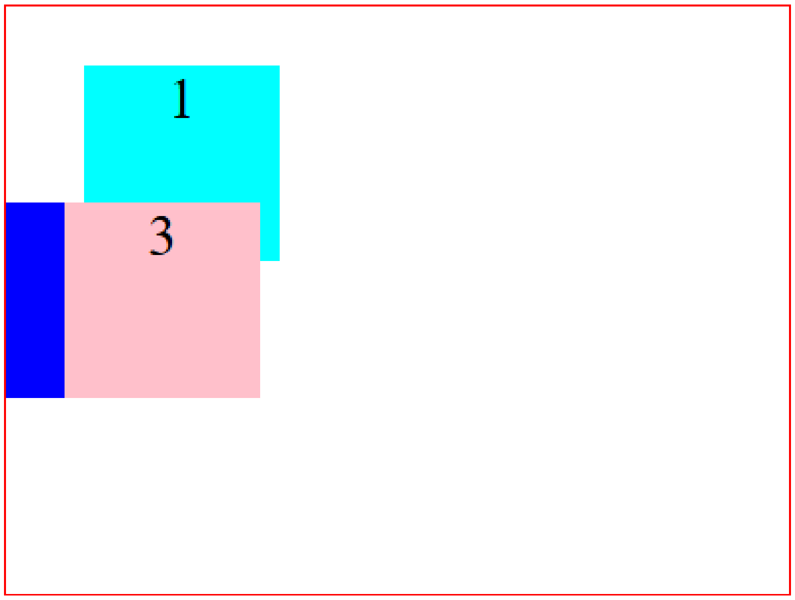
Property	Description	Possible Values	Default Value	Syntax / Example
position	Defines how an element is positioned in the document.	static, relative, absolute, fixed, sticky	static	position: absolute;
top	Distance between element and top edge of container.	Any CSS length (px, %, auto)	auto	top: 20px;
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.**
- ✓ **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.**

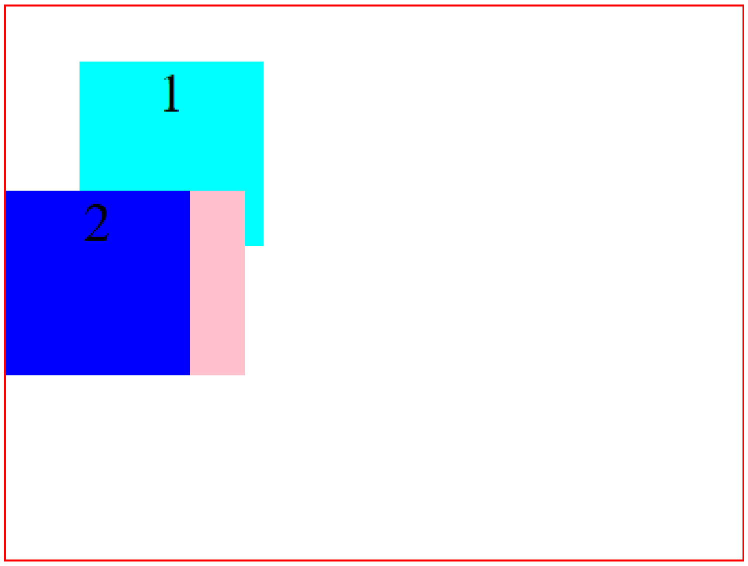
```
<head>
   <style>
.container{
   height: 300px;
   width: 400px;
   border: 1px solid red;
   position: relative;
}
.test {
   height: 100px;
   width : 100px;
   text-align: center;
```

```
    font-size:30px;
}
#test1{
    background-color: aqua;
    position: relative;
    top: 30px;
    left: 40px;
}
#test2{
    background-color: blue;
    position: absolute;
}
#test3{
    background-color: pink;
    position: absolute;
    left: 30px;
}
    </style>
</head>
<body>
    <div class="container">
        <div class="test" id="test1">1</div>
        <div class="test" id="test2">2</div>
        <div class="test" id="test3">3</div>
    </div>
</body>
```

- ✓ .container is **position: relative**, so absolutely positioned elements use it as their reference.
- ✓ #test1 is **relative** and moved **30px down** and **40px right** from its normal spot.
- ✓ #test2 is **absolute** with no top/left values, so it sits at the **top-left** of the container (0,0).
- ✓ #test3 is **absolute** with left: 30px, so it appears at the top, **30px from the left**, overlapping #test2



#test2 (blue box) has z-index: 1 → it appears in front of **#test3 (pink box) when they overlap.**



Adding z-index: 1 to #test2 makes box 2 appear on top of any other overlapping elements.

Chapter 5

CSS Multiple Column Properties

Property	Description
column-count	Specifies the number of columns an element should be divided into.
column-gap	Specifies the gap between the columns.
column-rule-style	Specifies the style of the rule between columns (none, dotted, dashed, solid, double).
column-rule-color	Specifies the color of the rule between columns.
column-rule-width	Specifies the width of the rule between columns.
Shorthand	column-rule: width style color; e.g., column-rule: 1px dotted red;

Image Properties

Property	Syntax	Example	Description
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: 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.
opacity	opacity:value	opacity:0.5	Sets the transparency level of an element, where 1 is fully opaque and 0 is fully transparent.

CSS Transform Functions

Transform Function	Example	Description / Effect
Translate (move element)		
translate(20px)	Moves element **20px right**	Equivalent to translateX(20px) — moves along **X-axis** only.
translate(20px, 20px)	Moves element **20px right and 20px down**	First value = **X**, second = **Y**.
translateX(20px)	Moves element **20px right**	Positive → right, negative → left.
translateY(20px)	Moves element **20px down**	Positive → down, negative → up.
Scale (resize element)		
scale(1)	No change	1 means 100% original size.
scale(2, 1.5)	Width ×2, Height ×1.5	First = **X scale**, second = **Y scale**.
scale(2)	Doubles size both horizontally and vertically	Equivalent to scale(2, 2).
scaleX(2)	Doubles width only	Height remains unchanged.
scaleY(2)	Doubles height only	Width remains unchanged.
scale(-1)	Flips element **horizontally and vertically**	Same as scale(-1, -1) → mirrored both axes.
scaleX(-1)	Flips **horizontally** (mirror image left–right)	Common trick to flip images or icons.
scaleY(-1)	Flips **vertically** (mirror image top–bottom)	Often used for reflection effects.
Rotate (turn element)		
rotate(45deg)	Rotates element **45° clockwise**	Rotation center is by default the element's center.
rotate(-45deg)	Rotates element **45° counterclockwise**	Negative values rotate opposite direction.
Skew (tilt element)		
skew(45deg)	Tilts element **45° along X-axis**	Equivalent to skewX(45deg).
skew(45deg, 20deg)	Tilts element **45° on X, 20° on Y**	Creates a combined shearing effect.
skewX(15deg)	Tilts only along **X-axis**	Top edge stays still, bottom edge moves.
skew(15deg, 0deg)	Tilts **15° along X**, none along Y	Same as skewX(15deg).
skewY(15deg)	Tilts only along **Y-axis**	Left edge stays still, right edge moves.

```
transform: rotate(45deg) scale(1.5) translate(50px, 50px) skew(15deg,15deg);
```

Transition Properties

Property	Syntax	Possible Values	Default Value	Required?
transition-property	transition-property: property-name;	- all → applies to all animatable properties - Single property: background-color, width, transform, etc. - Multiple properties separated by commas	all	No
transition-duration	transition-duration: time;	- Time in **seconds (s)** or **milliseconds (ms)** - e.g. 1s, 500ms	0s (no transition)	Yes
transition-timing-function	transition-timing-function: value;	- ease → starts slow, speeds up, slows down (default) - linear → constant speed - ease-in → starts slow - ease-out → ends slow - ease-in-out → starts & ends slow - steps(n) → jumps in n equal steps - cubic-bezier(x1, y1, x2, y2) → custom curve	ease	No
transition-delay	transition-delay: time;	- Time before transition starts - Can be positive or negative - e.g. 1s, 0.5s, -2s	0s	No

transition: property duration timing-function delay;

Create a CSS program that demonstrates the use of multiple transitions on a single element.
When the user hovers over the div, it should:

- **Increase in size,**
- **Change background color,**
- **Rotate 360 degrees,**
- **Become circular,**
  all with smooth timing, delay, and different transition durations.

```
<head>
<style>
div {
  width: 100px;
  height: 100px;
  background: rgb(142, 7, 233);

  /* Transition for multiple properties including transform */
  transition-property: width, height, transform, background-color,border-radius;
  transition-duration: 3s, 5s, 4s,6s,6s;
  transition-delay: 2s; /* Wait 2s before starting transition */
  transition-timing-function: ease-in-out;
}
div:hover {
  width: 150px;
  height: 150px;
  transform: rotate(360deg);
  background-color: yellow;
  border-radius:50%;
}
</style>
</head>
<body>
  <div></div>
</body>
```

Animation Properties

Property	Purpose	Possible Values / Examples
animation-name	Specifies the name of the @keyframes animation to apply.	Any valid animation name defined with @keyframes. Example: animation-name: bounce;
animation-duration	Defines how long one animation cycle takes to complete.	Time value in seconds (s) or milliseconds (ms). Examples: 2s, 500ms
animation-delay	Specifies the time to wait before starting the animation.	Time value (seconds or milliseconds). Can be negative. Examples: 1s, -2s
animation-iteration-count	Defines how many times the animation repeats.	A number or 'infinite'. Examples: 3, infinite
animation-direction	Defines whether the animation runs forward, backward, or alternates.	normal – runs forward (default) reverse – runs backward alternate – runs forward then backward alternate-reverse – runs backward then forward
animation-timing-function	Controls the speed curve of the animation.	linear – constant speed ease – starts slow, speeds up, slows down (default) ease-in – starts slow ease-out – ends slow ease-in-out – starts and ends slow

animation (shorthand)	Combines all animation properties in one line.	Syntax: animation: name duration timing-function delay iteration-count direction;
		Example: animation: bounce 2s ease-in-out 1s infinite alternate;

Animation in reverse direction with infinite iteration count

```
<html>
<head>
<style>
div {
width: 100px;
height: 100px;
background-color: red;
position: relative;
animation-name: example;
animation-duration: 4s;
animation-direction: reverse;
animation-iteration-count: infinite;
animation-delay:2s;
animation-timing-function:ease-in;
}
@keyframes example {
0%  {background-color:red; left:0px; top:0px;}
25% {background-color:yellow; left:200px; top:0px;}
50% {background-color:blue; left:200px; top:200px;}
75% {background-color:green; left:0px; top:200px;}
100% {background-color:red; left:0px; top:0px;}
}
</style>
</head>
<body>
<div></div>
</body>
</html>
```

Gradients

Gradient Type	Property / Syntax	Values / Options	Example	Description
Linear Gradient (Multiple Colors)	background-image: linear-gradient(direction, color1, color2, color3, ...);	Any number of colors	background-image: linear-gradient(to right, red, orange, yellow, green, blue);	Creates rainbow or multi-color transitions.
Linear Gradient (Angle)	background-image: linear-gradient(angle, color-stop1, color-stop2, ...);	0deg = to top,90deg = to right,180deg = to bottom,270deg = to left	background-image: linear-gradient(45deg, blue, pink);	Controls gradient direction precisely using angles.
Linear Gradient (Color Stops %)	background-image: linear-gradient(color1 stop%, color2 stop%);	Percentages for exact color positions	background-image: linear-gradient(red 20%, yellow 50%, green 80%,purple 100%);	Controls where each color begins and ends in the gradient.
Radial Gradient	background-image: radial-gradient(shape ,color-stop1, color-stop2, ...);	Shapes: circle, **ellipse** (**default**)	background-image: radial-gradient(circle, red, yellow, blue);	Creates circular or elliptical gradients radiating from a center.
Radial Gradient (Different Color Stops)	background-image: radial-gradient(color1 %, color2 %, color3 %);	Percent values for spread	background-image: radial-gradient(yellow 10%, red 25%, pink 50%);	Controls how quickly colors transition in circular gradient.
Conic Gradient	background-image: conic-gradient(color1, color2, ...);	Colors blend around a center point	background-image: conic-gradient(red, yellow, green, blue);	Colors rotate around the center like a color wheel. Default 0deg
Conic Gradient (from angle)	background-image: conic-gradient(from angle, color1, color2, ...);	Starting angle in degrees	background-image: conic-gradient(from 90deg, red, yellow, green);	Rotates the starting point of the conic gradient.
Conic Gradient (Color Segments)	background-image: conic-gradient(color1 startdeg enddeg, color2 startdeg enddeg, ...);	Degrees define color segment ranges	**background-image**: conic-gradient(from 90deg, red 0deg, red 45deg, green 45deg, green 90deg, blue 90deg, blue 180deg, yellow 180deg,	Creates pie-chart-like color sections.

Gradient Type	Property / Syntax	Values / Options	Example	Description

			yellow 315deg, pink 315deg);
			or
			background-image: conic-gradient(from 90deg, red 0deg 45deg, green 45deg 90deg, blue 90deg 180deg, yellow 180deg 315deg, pink 315deg);	

background-image: linear-gradient(to right top, purple, yellow);
background-image: radial-gradient(circle, blue, yellow, pink);
background-image: conic-gradient(from 90deg, red, yellow, green);

Variables

Category	Property / Syntax	Values / Example	Description / Usage
Definition	--variable-name: value;	--main-color: blue;	Defines a CSS variable (custom property). Must start with two dashes (--).
Usage	var(--variable-name)	color: var(--main-color);	Retrieves the value of the variable.
Global Variable	Declared inside :root	:root { --text-color: green; }	Can be accessed by all elements throughout the document.
Local Variable	Declared inside a selector	p { --border: solid; }	Accessible only within that selector's scope.
Case Sensitivity	Variable names are case-sensitive	--Color ≠ --color	Ensure consistent naming.
Overriding (Local > Global)	Local variable overrides global	:root { --b: blue; } p { --b: brown; color: var(--b); }	Local value (brown) will apply to <p> instead of global (blue).

Example:

```
:root{ --b : black; }
p{  --b: blue; color:var(--b); }
```

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	Possible Values	Example	Description / Usage
Basic Syntax	@media not|only mediatype and (mediafeature) and|or (mediafeature) { /* CSS rules go here */ }	-	@media screen and (max-width: 600px) { ... }	Defines conditional styles based on device features.
Media Types	all, screen	-	@media screen { ... }	Specifies the type of output device the styles apply to.
Orientation	(orientation: portrait)(orientation: landscape)	portrait / landscape	@media (orientation: portrait) { ... }	Adjusts styles based on device orientation.
Combining Conditions	and, or, not	logical operators	@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)	width range	@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)	not + feature	@media not (orientation: landscape) { body { background: pink; } }	Excludes landscape orientation (applies to portrait).

Category	**Property / Syntax**	**Possible Values**	**Example**	**Description / Usage**
'only' Keyword Example	@media only screen and (max-width: 600px)	only + mediatype	Ensures old browsers ignore unsupported queries.	
Device Adaptation Example	@media screen and (max-width: 400px)	mobile	body { background: orange; color: white; }	Styles for small screens like phones.
Tablet View Example	@media screen and (min-width: 401px) and (max-width: 1024px)	tablet	body { background: pink; color: blue; }	Styles for tablets or medium devices.

If you don't specify a media type like screen, the media query applies to all media types **(by default = all).**
If you do specify **screen, it applies** only **when the output device is a** screen **(computer, tablet, mobile, etc.) not printers or other media types.**