

Introduction to CSS



Agenda

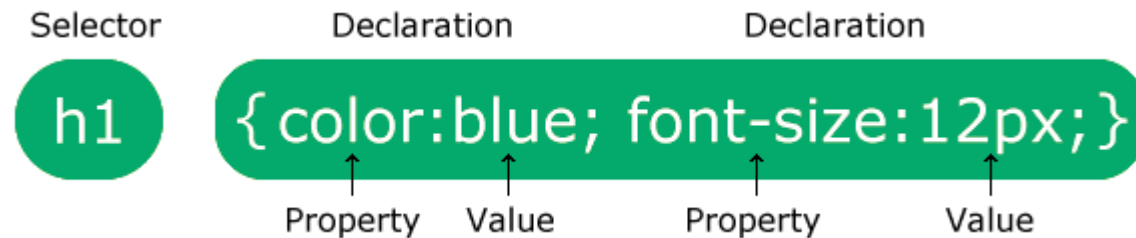
- What is CSS?
- CSS Syntax
- CSS Selector
- How to CSS
- CSS Functions
- CSS Units
- CSS References
- HTML Style

What is CSS?

- CSS is the language we use to style a web page
 - CSS stands for Cascading Style Sheets
 - CSS describes how HTML elements are to be displayed on screen or media
 - CSS can control the layout of web page

CSS Syntax

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



CSS Selector

- CSS selectors are to find or select the HTML elements to style
- CSS can divide to 5 categories
 - Simple selectors
 - select elements based on name, id, class
 - Combinator selectors
 - select elements based on a specific relationship between them
 - Pseudo-class selectors
 - select elements based on a certain state
 - Pseudo-elements selectors
 - select and style a part of an element
 - Attribute selector
 - select elements based on an attribute or attribute value

CSS Selector

- Simple Selector

Selector	Example	Example description
<u>#id</u>	#firstname	Selects the element with id="firstname"
<u>.class</u>	.intro	Selects all elements with class="intro"
<u>element.class</u>	p.intro	Selects only <p> elements with class="intro"
<u>*</u>	*	Selects all elements
<u>element</u>	p	Selects all <p> elements
<u>element,element,..</u>	div, p	Selects all <div> elements and all <p> elements

CSS Selector

- #id Selector

- syntax

```
#id {  
    css declarations;  
}
```

- example

```
#firstname {  
    background-color: yellow;  
}
```

CSS Selector

- .class Selector

- syntax

```
.class {  
  css declarations;  
}
```

- example

```
.intro {  
  background-color: yellow;  
}
```


CSS Selector

- element.class Selector
 - syntax

```
element.class {  
    css declarations;  
}
```

- example

```
p.intro {  
    background-color: yellow;  
}
```

CSS Selector

- * Selector

- syntax

```
* {  
    css declarations;  
}
```

- example

```
div * {  
    background-color: yellow;  
}
```

CSS Selector

- element Selector
 - syntax

```
element {  
    css declarations;  
}
```

- example

```
p {  
    background-color: yellow;  
}
```

CSS Selector

- element,element Selector
 - syntax

```
element, element {  
    css declarations;  
}
```

- example

```
h2, p {  
    background-color: yellow;  
}
```

How to CSS

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

How to CSS

- Inline CSS

```
<!DOCTYPE html>
<html>
<body style="background-color: powderblue;">
<h1 style="color: blue;">A Blue Heading</h1>
<p style="color: red;">A red paragraph.</p>
</body>
</html>
```

How to CSS

- Internal CSS

```
<!DOCTYPE html>
<html>
<head>
<style>
    body { background-color: powderblue;}
    h1 {color: blue;}
    p {color: red;}
</style>
</head>
<body>
<h1>A Blue Heading</h1>
<p>A red paragraph.</p>
</body>
</html>
```

How to CSS

- External CSS

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="style.css">
</head>
<body>
<h1>A Blue Heading</h1>
<p>A red paragraph.</p>
</body>
</html>
```

- style.css

```
body { background-color: powderblue;}
h1 {color: blue;}
p {color: red;}
```


How to CSS

- Cascading Priority
 - 1. inline style (inside an HTML element)
 - 2. external and internal style (in the head section)
 - 3. browser default

inline style has the highest priority,
and will override external and internal styles
and browser defaults

CSS Functions

- CSS functions are used as a value for various CSS properties

Function	Description
<u>attr()</u>	Returns the value of an attribute of the selected element
<u>calc()</u>	Allows you to perform calculations to determine CSS property values
<u>conic-gradient()</u>	Creates a conic gradient
<u>counter()</u>	Returns the current value of the named counter
<u>cubic-bezier()</u>	Defines a Cubic Bezier curve
<u>hsl()</u>	Defines colors using the Hue-Saturation-Lightness model (HSL)
<u>hsla()</u>	Defines colors using the Hue-Saturation-Lightness-Alpha model (HSLA)
<u>linear-gradient()</u>	Creates a linear gradient

CSS Functions

Function	Description
<u>max()</u>	Uses the largest value, from a comma-separated list of values, as the property value
<u>min()</u>	Uses the smallest value, from a comma-separated list of values, as the property value
<u>radial-gradient()</u>	Creates a radial gradient
<u>repeating-conic-gradient()</u>	Repeats a conic gradient
<u>repeating-linear-gradient()</u>	Repeats a linear gradient
<u>repeating-radial-gradient()</u>	Repeats a radial gradient
<u>rgb()</u>	Defines colors using the Red-Green-Blue model (RGB)
<u>rgba()</u>	Defines colors using the Red-Green-Blue-Alpha model (RGBA)
<u>var()</u>	Inserts the value of a custom property

CSS Units

- Absolute Length
 - The absolute length units are fixed and a length expressed in any of these will appear as exactly that size

Unit	Description
cm	centimeters
mm	millimeters
in	inches (1in = 96px = 2.54cm)
px *	pixels (1px = 1/96th of 1in)
pt	points (1pt = 1/72 of 1in)
pc	picas (1pc = 12 pt)

CSS Units

- Relative Length
 - Relative length units specify a length relative to another length property

Unit	Description
em	Relative to the font-size of the element (2em means 2 times the size of the current font)
ex	Relative to the x-height of the current font (rarely used)
ch	Relative to the width of the "0" (zero)
rem	Relative to font-size of the root element
vw	Relative to 1% of the width of the viewport*
vh	Relative to 1% of the height of the viewport*
vmin	Relative to 1% of viewport's* smaller dimension
vmax	Relative to 1% of viewport's* larger dimension
%	Relative to the parent element























CSS Reference

- initial
 - used to set a property to its default value
- inherit
 - used to set a property to its value from its parent
- unset
 - used to resets a property of an element (inherit + initial)
- revert
 - used to reset a property back to the browser specific style
- revert-layer
 - used to reset a property back to a previous layer
- all
 - shorthand property of an element's properties (all properties)

CSS Color

- Color Name

	IndianRed	#CD5C5C	rgb(205, 92, 92)
	LightCoral	#F08080	rgb(240, 128, 128)
	Salmon	#FA8072	rgb(250, 128, 114)
	DarkSalmon	#E9967A	rgb(233, 150, 122)
	LightSalmon	#FFA07A	rgb(255, 160, 122)
	Crimson	#DC143C	rgb(220, 20, 60)
	Red	#FF0000	rgb(255, 0, 0)
	FireBrick	#B22222	rgb(178, 34, 34)
	DarkRed	#8B0000	rgb(139, 0, 0)

	Pink	#FFC0CB	rgb(255, 192, 203)
	LightPink	#FFB6C1	rgb(255, 182, 193)
	HotPink	#FF69B4	rgb(255, 105, 180)
	DeepPink	#FF1493	rgb(255, 20, 147)
	MediumVioletRed	#C71585	rgb(199, 21, 133)
	PaleVioletRed	#DB7093	rgb(219, 112, 147)
	LightSalmon	#FFA07A	rgb(255, 160, 122)
	Coral	#FF7F50	rgb(255, 127, 80)
	Tomato	#FF6347	rgb(255, 99, 71)
	OrangeRed	#FF4500	rgb(255, 69, 0)
	DarkOrange	#FF8C00	rgb(255, 140, 0)
	Orange	#FFA500	rgb(255, 165, 0)

CSS Color

- Back Ground Color

```
<!DOCTYPE html>
<html>
<body>
<h1 style="background-color: blue;">A Blue Heading</h1>
<p style="background-color: red;">A red paragraph.</p>
</body>
</html>
```


CSS Color

- Text Color

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color: blue;">A Blue Heading</h1>
<p style="color: red;">A red paragraph.</p>
</body>
</html>
```

CSS Color

- Border Color

```
<!DOCTYPE html>
<html>
<body>
<h1 style="border: 2px solid blue;">A Blue Heading</h1>
<p style="border: 2px solid red;">A red paragraph.</p>
</body>
</html>
```

CSS Color

- Color Value

- HEX - #rrggbb, #rgb
- RGB – rgb(red, green, blue)
- RGBA – rgba(red, green, blue, alpha)
- HSL – hsl(hue, saturation, lightness)
- HSLA – hsla(hue, saturation, lightness, alpha)
 - Hue is a degree on the color wheel from 0 to 360. 0 is red, 120 is green, and 240 is blue
 - Saturation is a percentage value. 0% means a shade of gray, and 100% is the full color
 - Lightness is also a percentage value. 0% is black, and 100% is white
 - Alpha parameter is a number between 0.0 (fully transparent) and 1.0 (not transparent)

CSS Color

- Color Value

```
<!DOCTYPE html>
<html>
<body>
<h1 style="color: #FF0000;">A red Heading</h1>
<h1 style="color: #00F;">A blue Heading</h1>
<h1 style="color: rgb(255,0,0);">A red Heading.</h1>
<h1 style="color: hsl(240,100%,50%)">A blue
Heading</h1>
<h1 style="color: rgba(255,0,0,1.0);">A red
Heading.</h1>
<h1 style="color: hsla(240,100%,50%,1.0)">A blue
Heading</h1>
</body>
</html>
```

CSS Background

- background-color
 - opacity - specifies the opacity/transparency a value from 0.0 - 1.0

```
<!DOCTYPE html>
<html>
<body>
<h1 style="background-color: #FF0000;">A red
Heading</h1>
<h1 style="background-color: #0000FF; opacity: 0.5;">A
blue Heading</h1>
</body>
</html>
```

CSS Background

- background-image

```
<!DOCTYPE html>
<html>
<head>
<style>
    div {
        width: 100%; height: 100%;
        background-image: url('building.jpg');
    }
</style>
</head>
<body>
<div></div>
</body>
</html>
```

CSS Background

- background-image
 - background-repeat
 - repeat-x, repeat-y, no-repeat
 - background-position
 - top, right, bottom, left, center
 - percentage ex. 25% 75% (x% y%)
 - length values ex. 5cm 5cm (xpos ypos)
 - edge offsets values ex. left 25% top 75%
 - background-attachment
 - fixed, scroll
 - background-size
 - auto, cover, contain, or width height
- background shorthand
 - background-color > background-image > background-repeat > background-attachment > background-position

CSS Background

- background-repeat

```
<!DOCTYPE html>
<html>
<head>
<style>
    div {
        width: 100%; height: 100%;
        background-image: url('building.jpg');
        background-repeat: no-repeat;
        background-size: cover;
    }
</style>
</head>
<body>
<div></div>
</body>
</html>
```

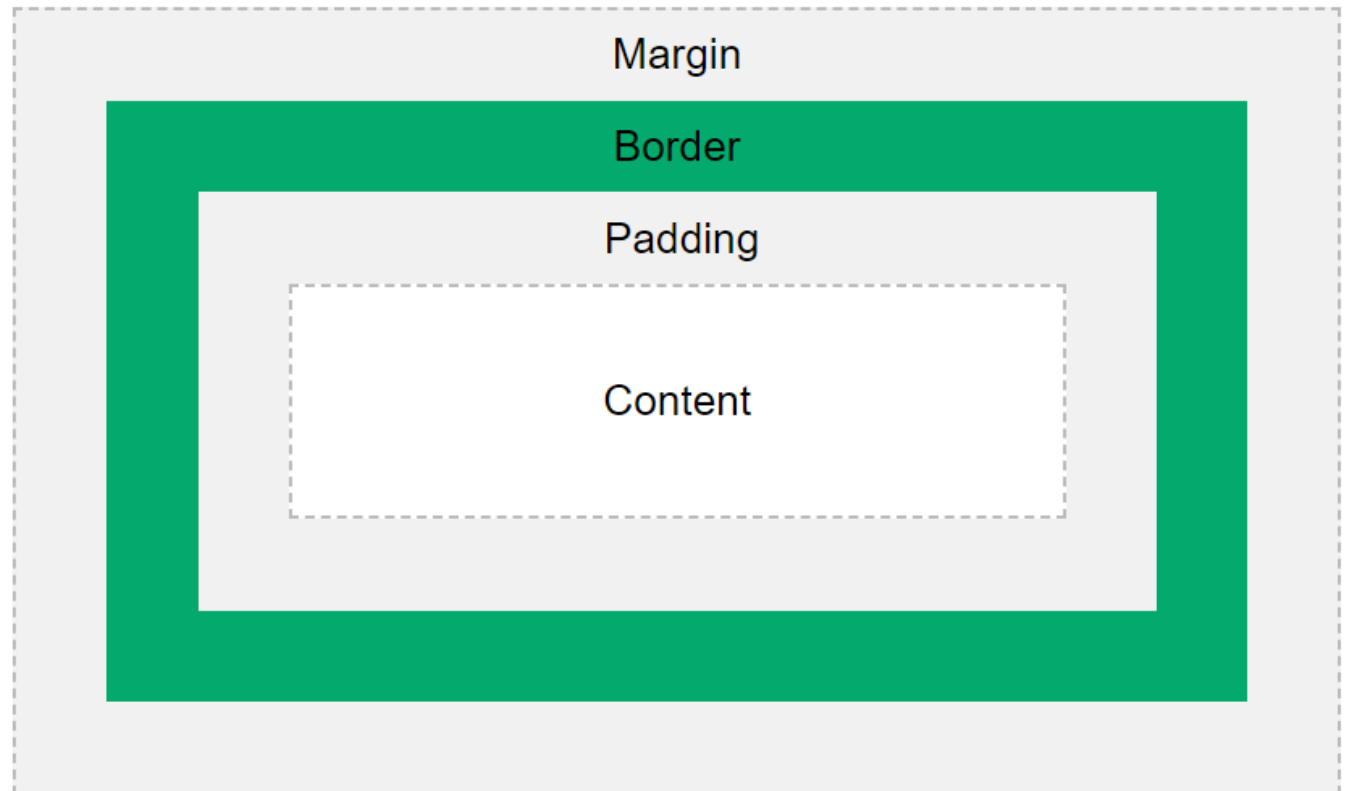

CSS Background

- background-position

```
<!DOCTYPE html>
<html>
<head>
<style>
    div {
        width: 100%; height: 100%;
        background-image: url('go.png');
        background-repeat: no-repeat;
        background-position: top right;
    }
</style>
</head>
<body>
<div></div>
</body>
</html>
```

CSS Box Model

- All HTML elements can be considered as boxes



CSS Border

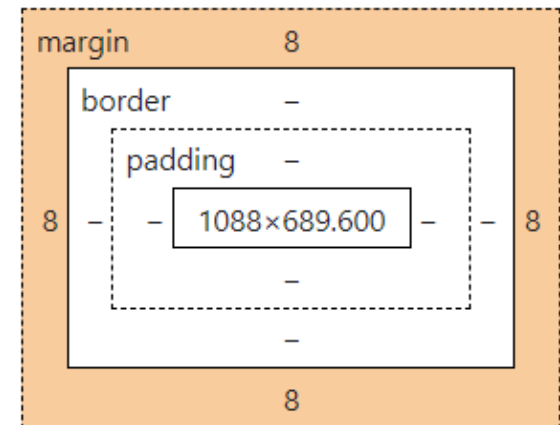
- To specify the style, width and color of an element's border
 - border-width
 - border-style
 - dotted, dashed, solid, double, groove, ridge, inset, outset, none, hidden
 - border-color
 - border-radius
 - border side
 - border-top, border-right, border-bottom, border-left
 - border shorthand
 - border-width > border-style > border-color

CSS Border

```
<!DOCTYPE html>
<html>
<body>
<p style="border-width: 2px; border-style: dotted; border-color: red;">Border dotted</p>
<p style="border: 2px dashed green;">Border dashed</p>
<p style="border: 2px solid blue;">Border solid</p>
<p style="border: 2px double black;">Border double</p>
<p style="border: 2px groove black;">Border groove</p>
<p style="border: 2px ridge black;">Border ridge</p>
<p style="border: 2px inset black;">Border inset</p>
<p style="border: 2px outset black;">Border outset</p>
<p style="border: 2px none black;">Border none</p>
<p style="border: 2px hidden black;">Border hidden</p>
<p style="border-bottom: 2px solid black;">Border side</p>
<p style="border: 2px solid black; border-radius: 10px;">Border radius</p>
</body>
</html>
```

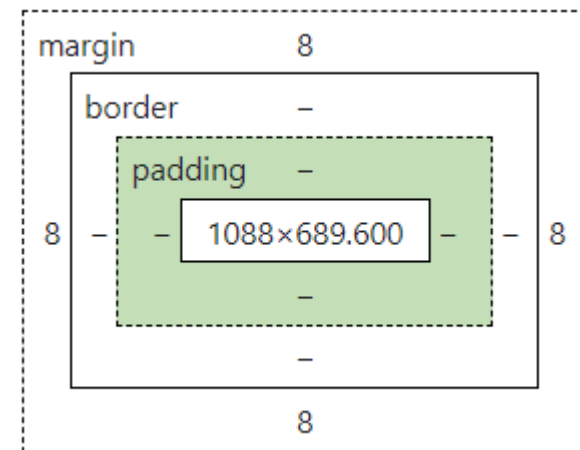
CSS Margin

- Margins are used to create space around elements outside of any defined borders
 - margin side
 - margin-top, margin-right, margin-bottom, margin-left
 - values: auto, length, %
 - margin shorthand
 - top > right > bottom > left



CSS Padding

- Padding is used to create space around an element's content inside of any defined borders
 - padding side
 - padding-top, padding-right, padding-bottom, padding-left
 - values: length, %
 - padding shorthand
 - top > right > bottom > left



CSS Width/Height

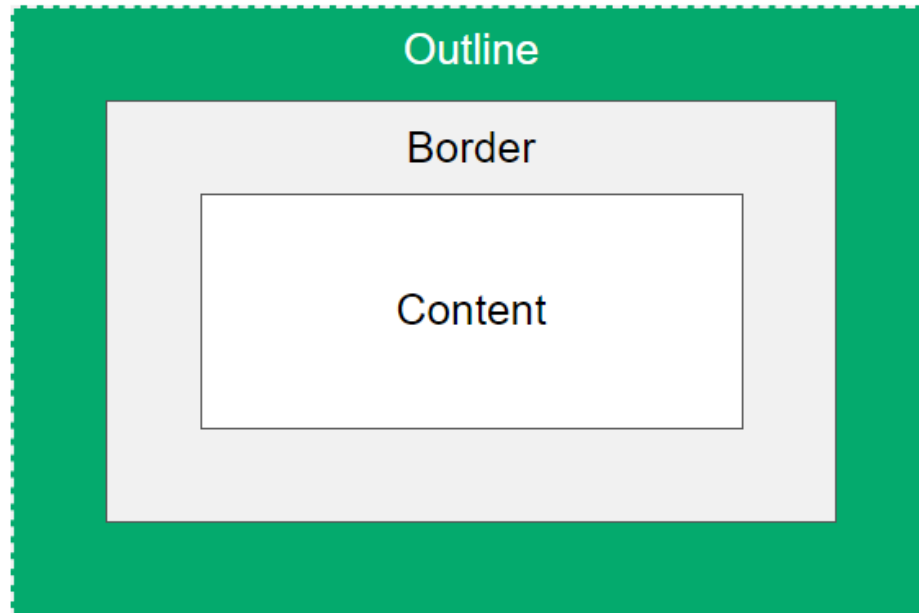
- Setting width/height
 - width
 - height
 - max-width, min-width
 - max-height, min-height
- Setting width/height values
 - auto – browser calculate width and height
 - length – defined in units, ex. cm, px, pt
 - % - defined in percent of containing block

CSS Outline

- An outline is a line drawn outside the element's border
 - outline-width
 - outline-style
 - dotted, dashed, solid, double, groove, ridge, inset, outset, none, hidden
 - outline-color
 - outline-offset
 - outline shorthand
 - outline-width > **outline-style** > outline-color

CSS Outline

- Outline is NOT a part of the element's dimension that may overlap other content



CSS Text

- A lot of properties for formatting text
 - color
 - text-align
 - center, left, right, justify
 - text-align-last
 - text-decoration-line
 - overline, underline, line-through
 - text-decoration-color
 - text-decoration-style
 - dotted, dashed, solid, double, wavy
 - text-decoration-thickness
 - text-decoration shorthand
 - **text-decoration-line** > text-decoration-color > text-decoration-style > text-decoration-thickness



CSS Text

- A lot of properties for formatting text
 - text-transformation
 - uppercase, lowercase, capitalize, none
 - text-indent
 - letter-spacing
 - line-height
 - word-spacing
 - white-space
 - normal, nowrap, pre, pre-wrap, pre-line, break-spaces
 - text-shadow
 - offset-x > offset-y > blur-radius > color
 - text-overflow
 - clip, ellipsis

CSS Text

```
<!DOCTYPE html>
<html>
<head>
<style>
  div { width: 200px; }
  .txt-left { text-align: left; color: red; }
  .txt-center { text-align: center; color: green; text-transform: capitalize; }
  .txt-right { text-align: right; color: blue; text-transform: uppercase; }
  .txt-line { text-decoration: underline black dotted; }
  .txt-shadow { text-shadow: 1px 1px blue; }
  .txt-space { white-space: nowrap; }
  .txt-over { overflow: hidden; text-overflow: ellipsis; }
</style>
</head>
<body>
<div>
  <div class="txt-left txt-line">text left under line</div>
  <div class="txt-center txt-decorate">text center</div>
  <div class="txt-right">text right</div>
  <div class="txt-shadow">text shadow</div>
  <div class="txt-space">This is simple text if it exceed width limit.</div>
  <div class="txt-space txt-over">This is simple text if it exceed width
limit.</div>
</div>
</body>
</html>
```

CSS Font

- Generic Font Families
 - Serif – fonts have a small stroke at the edges of each letter
 - Sans-serif – fonts have clean lines no small strokes attached
 - Monospace – fonts all the letters have the same fixed width
 - Cursive – fonts imitate human handwriting
 - Fantasy – fonts are decorative and playful

CSS Font

- Font Example

Font Family	Examples of Font
Serif	Times New Roman Georgia Garamond
Sans-serif	Arial Verdana Helvetica
Monospace	Courier New Luci da Consol e Monaco
Cursive	<i>Brush Script M7</i> <i>Lucida Handwriting</i>
Fantasy	Copperplate Papyrus

CSS Font

- Properties
 - font-family
 - font-size
 - font-style
 - normal, italic, oblique
 - font-weight
 - normal, bold, lighter, <number: 1 - 1000>
 - font-variant
 - normal, small-caps
 - font shorthand
 - font-style > font-variant > font-weight > font-size > font-family

CSS Font

```
<!DOCTYPE html>
<html>
<head>
<link rel="preconnect" href="https://fonts.googleapis.com">
<link rel="preconnect" href="https://fonts.gstatic.com" crossorigin>
<link href="https://fonts.googleapis.com/css2?family=Pacifico&display=swap"
rel="stylesheet">
<style>
  .fnt-family { font-family: 'Courier New', Courier, monospace; }
  .fnt-size { font-family: 'Times New Roman', Times, serif; font-size: 2em; }
  .fnt-style { font-style: italic; }
  .fnt-weight { font-weight: bold; }
  .fnt-hand { font: 20px Arial, Helvetica, sans-serif; }
  .fnt-api { font-family: 'Pacifico', cursive; }
</style>
</head>
<body>
<div>
  <div class="fnt-family">Font Family</div>
  <div class="fnt-size">Font Size</div>
  <div class="fnt-style">Font Style</div>
  <div class="fnt-weight">Font Weight</div>
  <div class="fnt-hand">This is a simple font</div>
  <div class="fnt-api">This is a simple font api</div>
</div>
</body>
</html>
```


CSS Icon

- Icon font - I

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" href="https://cdnjs.cloudflare.com/ajax/libs/font-
awesome/4.7.0/css/font-awesome.min.css">
</head>
<body>
<div>
  Basic icons<br/>
  <em class="fa fa-car"></em>
  <em class="fa fa-car" style="font-size:48px;"></em>
  <em class="fa fa-car" style="font-size:60px;color:red;"></em>
  <br/><br/>Animate icon<br/>
  <em class="fa fa-spinner fa-spin"></em>
  <em class="fa fa-circle-o-notch fa-spin"></em>
  <em class="fa fa-refresh fa-spin"></em>
  <em class="fa fa-cog fa-spin"></em>
  <br/><br/>Rotate icons<br/>
  <em class="fa fa-car fa-rotate-90"></em>
  <em class="fa fa-car fa-rotate-180"></em>
  <em class="fa fa-car fa-flip-horizontal"></em>
  <em class="fa fa-car fa-flip-vertical"></em>
```

CSS Icon

- Icon font - II

```
<br/><br/>Stack icons<br/>
<span class="fa-stack fa-lg">
  <em class="fa fa-circle-thin fa-stack-2x"></em>
  <em class="fa fa-twitter fa-stack-1x"></em>
</span>
<span class="fa-stack fa-lg">
  <em class="fa fa-circle fa-stack-2x"></em>
  <em class="fa fa-twitter fa-stack-1x fa-inverse"></em>
</span>
<span class="fa-stack fa-lg">
  <em class="fa fa-camera fa-stack-1x"></em>
  <em class="fa fa-ban fa-stack-2x text-danger" style="color:red;"></em>
</span>
</div>
</body>
</html>
```

CSS Display

- The **display** property specifies if/how an element is displayed
 - block – a block element always starts on a new line and takes up the full width
 - inline – an inline element does not start on a new line and only takes up as much width as necessary
 - inline-block - allows to set a width and height on the element, margin/padding are respected
 - none – to hide and show element without deleting and recreating



CSS Position

- The **position** property specifies the type of positioning method used for an element
 - static - default. elements are not affected by the top, bottom, left and right properties. It is always positioned according to the normal flow of the page
 - relative – an element is positioned relative to its normal position. Setting the top, bottom, left and right properties will cause it to be adjusted away from its normal position
 - fixed – an element is positioned relative to the viewport it always stays in the same place even if the page is scrolled
 - absolute – an element is positioned relative to the nearest positioned ancestor (like fixed)
 - sticky – an element is positioned based on the user's scroll position

CSS Z-Index

- The **z-index** property specified the stack order of an element which element should be placed in front of or behind the others

CSS Overflow

- The **overflow** property specified whether to clip the content or to add scrollbars when the content of an element is too big to fit in the specified area
 - visible – default. the overflow is not clipped, the content renders outside the element's box
 - hidden – the overflow is clipped and the rest of the content will be invisible
 - scroll – the overflow is clipped and a scrollbar is added to see the rest of the content
 - auto – like scroll but it adds scrollbars only when necessary
- **overflow-x** – specified what to do with the left/right edges of the content
- **overflow-y** – specified what to do with the top/bottom edges of the content

CSS Float

- The **float** property is used for positioning and formatting content
 - left – the element floats to the left of its container
 - right – the element floats to the right of its container
 - none – the element does not float. default.

CSS Combinator

- A CSS combinator can contain more than one simple selector
 - descendant selector (space)
 - child selector (>)
 - adjacent sibling selector (+)
 - general sibling selector (~)

CSS Combinator

```
<!DOCTYPE html>
<html>
<head>
  <style>
    div.vehicle p { background-color: cornsilk; }
    div.book > span { background-color: yellow; }
    div.drink + div { background-color: orange; }
    div.food ~ label { background-color: khaki; }
  </style>
</head>
<body>
  <div class="vehicle">
    <p>Car</p> <p>Motorcycle</p> <p>Bicycle</p>
  </div>
  <div class="book">
    <span>Java</span> <span>C++</span> <span>CS</span>
  </div>
  <div class="drink">Drink</div>
  <div>Coffee</div>
  <div>Tea</div>
  <div>Coca Cola</div>
  <div class="food">Food</div>
  <label>Salad</label>
  <label>Steak</label>
  <label>Spaghetti</label>
</body>
</html>
```

CSS Pseudo

- Pseudo class

Selector	Example	Example description
<u>:active</u>	a:active	Selects the active link
<u>:checked</u>	input:checked	Selects every checked <input> element
<u>:disabled</u>	input:disabled	Selects every disabled <input> element
<u>:empty</u>	p:empty	Selects every <p> element that has no children
<u>:enabled</u>	input:enabled	Selects every enabled <input> element
<u>:first-child</u>	p:first-child	Selects every <p> elements that is the first child of its parent
<u>:first-of-type</u>	p:first-of-type	Selects every <p> element that is the first <p> element of its parent
<u>:focus</u>	input:focus	Selects the <input> element that has focus
<u>:hover</u>	a:hover	Selects links on mouse over
<u>:in-range</u>	input:in-range	Selects <input> elements with a value within a specified range

CSS Pseudo

- Pseudo class

Selector	Example	Example description
<u>:invalid</u>	input:invalid	Selects all <input> elements with an invalid value
<u>:lang(<i>language</i>)</u>	p:lang(it)	Selects every <p> element with a lang attribute value starting with "it"
<u>:last-child</u>	p:last-child	Selects every <p> elements that is the last child of its parent
<u>:last-of-type</u>	p:last-of-type	Selects every <p> element that is the last <p> element of its parent
<u>:link</u>	a:link	Selects all unvisited links
<u>:not(selector)</u>	:not(p)	Selects every element that is not a <p> element
<u>:nth-child(n)</u>	p:nth-child(2)	Selects every <p> element that is the second child of its parent

CSS Pseudo

- Pseudo class

Selector	Example	Example description
<u>:nth-last-child(n)</u>	p:nth-last-child(2)	Selects every <p> element that is the second child of its parent, counting from the last child
<u>:nth-last-of-type(n)</u>	p:nth-last-of-type(2)	Selects every <p> element that is the second <p> element of its parent, counting from the last child
<u>:nth-of-type(n)</u>	p:nth-of-type(2)	Selects every <p> element that is the second <p> element of its parent
<u>:only-of-type</u>	p:only-of-type	Selects every <p> element that is the only <p> element of its parent
<u>:only-child</u>	p:only-child	Selects every <p> element that is the only child of its parent
<u>:optional</u>	input:optional	Selects <input> elements with no "required" attribute

CSS Pseudo

- Pseudo class

Selector	Example	Example description
:out-of-range	input:out-of-range	Selects <input> elements with a value outside a specified range
:read-only	input:read-only	Selects <input> elements with a "readonly" attribute specified
:read-write	input:read-write	Selects <input> elements with no "readonly" attribute
:required	input:required	Selects <input> elements with a "required" attribute specified
:root	root	Selects the document's root element
:target	#news:target	Selects the current active #news element (clicked on a URL containing that anchor name)
:valid	input:valid	Selects all <input> elements with a valid value
:visited	a:visited	Selects all visited links

CSS Pseudo

- Pseudo element

Selector	Example	Example description
<u>::after</u>	p::after	Insert content after every <p> element
<u>::before</u>	p::before	Insert content before every <p> element
<u>::first-letter</u>	p::first-letter	Selects the first letter of every <p> element
<u>::first-line</u>	p::first-line	Selects the first line of every <p> element
<u>::marker</u>	::marker	Selects the markers of list items
<u>::selection</u>	p::selection	Selects the portion of an element that is selected by a user

CSS Attribute Selector

- The attribute selector is used to select elements with a specified attribute
 - `[attribute]` – used to specified attribute
 - `[attribute="value"]` – used to specified attribute and value
 - `[attribute~="value"]` – used to select with an attribute value containing a specified word (space-separated)
 - `[attribute|="value"]` – used to select with an attribute whose value can be exactly specified value
 - `[attribute^="value"]` – used to select with an attribute whose value starts with specified value
 - `[attribute$="value"]` – used to select with an attribute whose value ends with a specified value
 - `[attribute*="value"]` – used to select with an attribute whose value contain a specified value

CSS Opacity/Transparency

- The **opacity** property specified the opacity/transparency of an element
 - value from 0.0 – 1.0 the lower value is more transparent

```
<!DOCTYPE html>
<html>
<head>
<style>
  img { opacity: 0.5; }
  img:hover { opacity: 1.0; }
</style>
</head>
<body>
  
  
  
</body>
</html>
```


CSS !important

- The **!important** rule in CSS is used to add more importance to a property/value than normal
- It will override ALL previous styling rules for that specific property on element
- Do not use it unless you absolutely have to

CSS Link

- Link states

- a:link – a normal, unvisited link
- a:visited – a link the user has visited
- a:hover – a link when user mouses over it
- a:active – a link the moment it is clicked

CSS Link

```
<!DOCTYPE html>
<html>
<head>
<style>
a.a1:link { background-color: yellow; }
a.a1:visited { background-color: cyan; }
a.a1:hover { background-color: lightgreen; }
a.a1:active { background-color: hotpink; }
a.a2:link, a.a2:visited {
  background-color: crimson;
  color: white;
  padding: 14px 25px;
  text-align: center;
  text-decoration: none;
  display: inline-block;
}
a.a2:hover, a.a2:active { background-color: red; }
</style></head>
<body>
  <a href="#0" class="a1">This is a link</a>
  <a href="#0" class="a2">This is a button</a>
</body>
</html>
```

CSS List

- 2 type of list
 - Unordered list () – the list items are marked as bullets
 - Ordered list () – the list items are marked with numbers or letters
- Properties
 - list-style-type – specified the type of list item marker
 - list-style-image – specified an image as the list item marker
 - list-style-position – inside, outside
 - list shorthand
 - list-style-type > list-style-position > list-style-image

CSS List

```
<!DOCTYPE html>
<html>
<head>
  <style>
    ol.vehicle { list-style-type: none; }
    ul.book { list-style-position: outside; }
    ul.drink { list-style: square inside url("sqpurple.gif"); }
  </style>
</head>
<body>
  <ol class="vehicle">
    <li>Car</li>
    <li>Motorcycle</li>
    <li>Bicycle</li>
  </ol>
  <ul class="book">
    <li>Java</li>
    <li>C++</li>
    <li>CS</li>
  </ul>
  <ul class="drink">
    <li>Coffee</li>
    <li>Tea</li>
    <li>Coca Cola</li>
  </ul>
</body>
</html>
```

CSS Table

- Table border
 - ex. border: 1px solid;
- Table size
 - ex. width: 100%; height: 100px;
- Table alignment
 - ex. text-align: center; vertical-align: bottom;
- Table hover
 - ex. tr:hover { background-color: coral; }
- Table striped
 - ex. tr:nth-child(even) { background-color: coral; }
- Table Color
 - ex. th { background-color: teal; color: white; }

CSS Table

- Table properties
 - border-collapse: separate;
 - separate – borders are separated, each cell will display its own borders
 - collapse – borders are collapsed into a single border
 - caption-side: bottom; (top/bottom)
 - Specified placement of a table caption
 - table-layout: fixed;
 - auto – the column width is set by the widest unbreakable content in the cells
 - fixed - the column widths area divided equally access the table

CSS Table

```
<!DOCTYPE html>
<html>
<head>
<style>
  table { caption-side: bottom; border-collapse: collapse; width: 250px; }
  table th { background-color: teal; color: white; }
  table, th, td { border: 2px solid black; }
  tbody tr:nth-child(even) { background-color: linen; }
  tbody tr:hover { background-color: coral; }
  tbody tr td:nth-child(2) { text-align: right; }
  tfoot tr td { text-align: right; background-color: lightgray; }
</style>
</head>
<body>
<table>
  <caption>Book Store</caption>
  <thead>
    <tr><th>Book</th><th>Price</th></tr>
  </thead>
  <tbody>
    <tr><td>Java</td><td>550</td></tr>
    <tr><td>C++</td><td>450</td></tr>
    <tr><td>C#</td><td>350</td></tr>
    <tr><td>Python</td><td>390</td></tr>
  </tbody>
  <tfoot><tr><td>Total</td><td>1740</td></tr></tfoot>
</table>
</body>
</html>
```


CSS Table

- Table responsive - I

```
<!DOCTYPE html>
<html>
<head>
<style>
  table { border-collapse: separate; table-layout: fixed; border-spacing: 0; }
  table th { background-color: teal; color: white; }
  table, th, td { border: 1px solid black; white-space: nowrap; }
  #bookpanel { overflow-x: auto; width: 395px; height: 107px; }
  #booktable thead th { position: sticky; top: 0; }
  .sticky-head { z-index: 1; }
  .sticky-col { position: sticky; background-color: white; }
  .first-col { width: 25px; min-width: 25px; max-width: 25px; left: 0px; }
  .second-col { width: 80px; min-width: 80px; max-width: 80px; left: 25px; }
</style>
</head>
<body>
<div id="bookpanel">
<table id="booktable">
  <thead>
    <tr>
      <th class="sticky-head first-col">No.</th>
      <th class="sticky-head second-col">Book</th>
      <th>Price</th> <th>Title</th> <th>Edition</th> <th>Author</th>
    <th>Publisher</th>
    </tr>
```

CSS Table

- Table responsive - II

```
</thead>
<tbody id="bookbody">
  <tr><td class="sticky-col first-col">1</td><td class="sticky-col second-
col">Java</td><td>550</td><td>Effective Java</td><td>5th Edition</td><td>Joshua
Bloch</td><td>O'Reilly</td></tr>
  <tr><td class="sticky-col first-col">2</td><td class="sticky-col second-
col">C++</td><td>450</td><td>C++ Pocket Reference</td><td>1th
Edition</td><td>Kyle Loudon</td><td>O'Reilly</td></tr>
  <tr><td class="sticky-col first-col">3</td><td class="sticky-col second-
col">C#</td><td>350</td><td>Head First C#</td><td>4th Edition</td><td>Andrew
Stellman</td><td>O'Reilly</td></tr>
  <tr><td class="sticky-col first-col">4</td><td class="sticky-col second-
col">Python</td><td>390</td><td>Learning Python</td><td>5th Edition</td><td>Mark
Lutz</td><td>O'Reilly</td></tr>
  <tr><td class="sticky-col first-col">5</td><td class="sticky-col second-
col">Type Script</td><td>490</td><td>TypeScript Cookbook</td><td>1th
Edition</td><td>Stefan Baumgartner</td><td>O'Reilly</td></tr>
</tbody>
</table>
</div>
</body>
</html>
```

CSS Border Image

- The CSS border-image property allows you to specify an image to be used as the border around an element

Property	Description
<u>border-image</u>	A shorthand property for setting all the border-image-* properties
<u>border-image-source</u>	Specifies the path to the image to be used as a border
<u>border-image-slice</u>	Specifies how to slice the border image. Default 100%
<u>border-image-width</u>	Specifies the widths of the border image. Default 1
<u>border-image-outset</u>	Specifies the amount by which the border image area extends beyond the border box. Default 0
<u>border-image-repeat</u>	Specifies whether the border image should be repeated, rounded or stretched. Default stretch.

CSS Border Image

```
<!DOCTYPE html>
<html>
<head>
<style>
p { border: 10px solid transparent; padding: 15px;}
#borderimg1 { border-image: url(border.png) 30 round; }
#borderimg2 { border-image: url(border.png) 50 round; }
#borderimg3 { border-image: url(border.png) 20% round; }
#borderimg4 { border-image: url(border.png) 30% round; }
#borderimg5 { border-image: url(border.png) 20 stretch; }
#borderimg6 { border-image: url(border.png) 30 stretch; }
#borderimg7 { border-image: url(border.png) 20 repeat; }
#borderimg8 { border-image: url(border.png) 40 repeat; }
</style>
</head>
<body>
<p id="borderimg1">border-image: url(border.png) 30 round;</p>
<p id="borderimg2">border-image: url(border.png) 50 round;</p>
<p id="borderimg3">border-image: url(border.png) 20% round;</p>
<p id="borderimg4">border-image: url(border.png) 30% round;</p>
<p id="borderimg5">border-image: url(border.png) 20 stretch;</p>
<p id="borderimg6">border-image: url(border.png) 30 stretch;</p>
<p id="borderimg7">border-image: url(border.png) 20 repeat;</p>
<p id="borderimg8">border-image: url(border.png) 40 repeat;</p>
</body>
</html>
```

CSS Shadow

- The CSS you can add shadow to text and element
 - text-shadow
 - horizontal > vertical > blur-effect > color
 - box-shadow
 - horizontal > vertical > blur-effect > spread-radius > color > inset-parameter
 - box-reflect
 - none, below, above, left, right, position offset gradient

CSS Shadow

- text-shadow

```
<!DOCTYPE html>
<html>
<head>
<style>
  #div1 { font-size: 50px; color: orange; text-shadow: 3px 3px 5px gray; }
  #div2 { font-size: 50px; color: orange;
          text-shadow: 3px 3px 5px black , 0 0 25px gold, 0 0 5px darkorange; }
  #div3 { font-size: 50px; color: orange;
          text-shadow: -1px 0 black, 0 1px black, 1px 0 black, 0 -1px black; }
</style>
</head>
<body>
  <div id="div1">Text Shadow</div><br/>
  <div id="div2">Multiple Shadow</div><br/>
  <div id="div3">Text Bolder</div><br/>
</body>
</html>
```

CSS Shadow

- box-shadow

```
<!DOCTYPE html>
<html>
<head>
<style>
  div.div-box { width: 200px; height: 100px; background-color: hotpink; padding: 5px;}
  #div4 { box-shadow: 5px 5px 5px 2px gray;
    border-style: ridge; border-width: 10px; border-color: hotpink; }
  #div5 { box-shadow: 5px 5px 5px gray; }
  #div6 { box-shadow: 5px 5px 5px lightgray inset; }
  #div7 { box-shadow: 0px 10px 5px -5px gray; }
  #div8 { box-shadow: 5px -5px 5px gray; }
  #boxtable { width: 400px; border-spacing: 5px 7px; }
  #boxtable td { height: 30px;
    background-color: hotpink; box-shadow: 5px 5px 5px gray; }
</style>
</head>
<body>
  <div id="div4" class="div-box">Box Shadow with Border</div><br/>
  <div id="div5" class="div-box">Box Shadow</div><br/>
  <div id="div6" class="div-box">Box Shadow Inset</div><br/>
  <div id="div7" class="div-box">Box Shadow Bottom</div><br/>
  <div id="div8" class="div-box">Box Shadow Top Right</div><br/>
  <table id="boxtable">
    <tr><td></td><td></td></tr>
    <tr><td></td><td></td></tr>
  </table>
</body>
</html>
```

CSS Shadow

- box-reflect

```
<!DOCTYPE html>
<html>
<head>
<style>
    #flower { -webkit-box-reflect: below; }
    #leaves { -webkit-box-reflect: below 10px linear-gradient(to bottom,
    rgba(0,0,0,0.00), rgba(0,0,0,0.5)); }
</style>
</head>
<body>
    
    
</body>
</html>
```


CSS Gradient

- The CSS gradient let you display smooth transitions between two or more colors
- 3 types of gradient
 - linear gradient – goes down/up/left/right/diagonally
 - radial gradient – defined by their center
 - conic gradient – rotated around a center

CSS Gradient

- Syntax
 - linear-gradient
 - background-image: linear-gradient (direction/angle, color-stop1, color-stop2, ...);
 - direction top to bottom as default
 - radial-gradient
 - background-image: radial-gradient(shape size at position, start-color, ..., last-color);
 - shape : circle or ellipse
 - size : closest-side, farthest-side, closest-corner, farthest-corner
 - by default shape is ellipse, size is farthest-corner and position is center
 - conic-gradient
 - background-image: conic-gradient([from angle] [at position,] color [degree], color [degree], ...);
 - by default angle is 0deg and position is center

CSS Gradient

- Functions

Function	Description
<u>conic-gradient()</u>	Creates a conic gradient. Define at least two colors (around a center point)
<u>linear-gradient()</u>	Creates a linear gradient. Define at least two colors (top to bottom)
<u>radial-gradient()</u>	Creates a radial gradient. Define at least two colors (center to edges)
<u>repeating-conic-gradient()</u>	Repeats a conic gradient
<u>repeating-linear-gradient()</u>	Repeats a linear gradient
<u>repeating-radial-gradient()</u>	Repeats a radial gradient

CSS Gradient

- linear-gradient

```
<!DOCTYPE html>
<html>
<head>
<style>
  div.div-box { width: 200px; height: 100px; }
  #div1 { background-image: linear-gradient(deeppink, white); }
  #div2 { background-image: linear-gradient(to right, deeppink, white); }
  #div3 { background-image: linear-gradient(to bottom right, deeppink, white); }
  #div4 { background-image: linear-gradient(180deg, deeppink, white); }
  #div5 { background-image: linear-gradient(to right, rgba(255,20,147,1),
  rgba(255,0,0,0)); }
  #div6 { background-image: repeating-linear-gradient(deeppink 10%, hotpink 20%, white
  30%); }
</style>
</head>
<body>
  <div id="div1" class="div-box">Linear by default top to bottom</div><br/>
  <div id="div2" class="div-box">Linear left to right</div><br/>
  <div id="div3" class="div-box">Linear diagonal top left to bottom right</div><br/>
  <div id="div4" class="div-box">Linear angle 180 degree</div><br/>
  <div id="div5" class="div-box">Linear using transparency</div><br/>
  <div id="div6" class="div-box">Linear repeating</div><br/>
</body>
</html>
```

CSS Gradient

- radial-gradient

```
<!DOCTYPE html>
<html>
<head>
<style>
  div.div-box { width: 200px; height: 100px; }
  #div1 { background-image: radial-gradient(deeppink, hotpink, white); }
  #div2 { background-image: radial-gradient(deeppink 10%, hotpink 20%, white 30%); }
  #div3 { background-image: radial-gradient(circle, deeppink, hotpink, white); }
  #div4 { background-image: radial-gradient(closest-side at 60% 50%, deeppink, hotpink,
white); }
  #div5 { background-image: repeating-radial-gradient(deeppink 10%, hotpink 20%, white
30%); }
</style>
</head>
<body>
  <div id="div1" class="div-box">Radial by default</div><br/>
  <div id="div2" class="div-box">Radial with space</div><br/>
  <div id="div3" class="div-box">Radial with circle shape</div><br/>
  <div id="div4" class="div-box">Radial with closest side</div><br/>
  <div id="div5" class="div-box">Radial repeating</div><br/>
</body>
</html>
```

CSS Gradient

- conic-gradient

```
<!DOCTYPE html>
<html>
<head>
<style>
  div.div-box { width: 200px; height: 100px; border-radius: 50%;}
  #div1 { background-image: conic-gradient(violet, deeppink, hotpink, white); }
  #div2 { background-image: conic-gradient(violet 0deg, violet 90deg, deeppink 90deg,
deeppink 180deg, crimson 180deg, crimson 270deg, orange 270deg); }
  #div3 { background-image: conic-gradient(from 90deg, violet, deeppink, hotpink,
white); }
  #div4 { background-image: conic-gradient(at 60% 40%, violet, deeppink, hotpink,
white); }
  #div5 { background-image: repeating-conic-gradient(deeppink 10%, white 20%); }
</style>
</head>
<body>
  <div id="div1" class="div-box">Conic by default</div><br/>
  <div id="div2" class="div-box">Conic with degree</div><br/>
  <div id="div3" class="div-box">Conic start with angle</div><br/>
  <div id="div4" class="div-box">Conic start with position</div><br/>
  <div id="div5" class="div-box">Conic repeating</div><br/>
</body>
</html>
```

CSS Transform

- The CSS transform allow you to move, scale and skew elements

Property	Description
<u>transform</u>	Applies a 2D or 3D transformation to an element
<u>transform-origin</u>	Allows you to change the position on transformed elements

CSS Transform

- Functions

Function	Description
<code>matrix(n,n,n,n,n,n)</code>	Defines a 2D transformation, using a matrix of six values : <code>matrix(scaleX(), skewY(), skewX(), scaleY(), translateX(), translateY())</code>
<code>translate(x,y)</code>	Defines a 2D translation, moving the element along the X- and the Y-axis
<code>translateX(n)</code>	Defines a 2D translation, moving the element along the X-axis
<code>translateY(n)</code>	Defines a 2D translation, moving the element along the Y-axis
<code>scale(x,y)</code>	Defines a 2D scale transformation, changing the elements width and height
<code>scaleX(n)</code>	Defines a 2D scale transformation, changing the element's width
<code>scaleY(n)</code>	Defines a 2D scale transformation, changing the element's height
<code>rotate(angle)</code>	Defines a 2D rotation, the angle is specified in the parameter
<code>skew(x-angle,y-angle)</code>	Defines a 2D skew transformation along the X- and the Y-axis
<code>skewX(angle)</code>	Defines a 2D skew transformation along the X-axis
<code>skewY(angle)</code>	Defines a 2D skew transformation along the Y-axis

CSS Transform

```
<!DOCTYPE html>
<html>
<head>
<link rel="stylesheet" type="text/css" href="star.css" />
<style>
#star-five {
    margin: 50px 0;
    position: relative;
    display: block;
    color: red; width: 0px; height: 0px;
    border-right: 100px solid transparent;
    border-bottom: 70px solid red;
    border-left: 100px solid transparent;
    transform: rotate(35deg);
}
#star-five:before {
    border-bottom: 80px solid red;
    border-left: 30px solid transparent;
    border-right: 30px solid transparent;
    position: absolute;
    height: 0; width: 0; top: -45px; left: -65px;
    display: block;
    content: '';
    transform: rotate(-35deg);
}
```

CSS Transform

```
#star-five:after {  
  position: absolute;  
  display: block;  
  color: red;  
  top: 3px;  
  left: -105px;  
  width: 0px;  
  height: 0px;  
  border-right: 100px solid transparent;  
  border-bottom: 70px solid red;  
  border-left: 100px solid transparent;  
  transform: rotate(-70deg);  
  content: '';  
}  
</style>  
</head>  
<body style="padding: 10px;">  
<div id="star-five"></div><br/>  
<div class="star"></div>  
<div class="star"></div>  
</body>  
</html>
```

CSS Transform

- star.css

```
.star {
  position: relative;
  display: inline-block;
  width: 0;
  height: 0;
  margin-left: .9em;
  margin-right: .9em;
  margin-bottom: 1.2em;
  border-right: .3em solid transparent;
  border-bottom: .7em solid #FC0;
  border-left: .3em solid transparent;
  font-size: 24px;
  &:before, &:after {
    content: '';
    display: block;
    width: 0;
    height: 0;
    position: absolute;
    top: .6em;
    left: -1em;
    border-right: 1em solid transparent;
    border-bottom: .7em solid #FC0;
    border-left: 1em solid transparent;
    transform: rotate(-35deg);
  }
  &:after {
    transform: rotate(35deg);
  }
}
```

CSS Transition

- The CSS transition allow you to change property values smoothly over a given duration
- A transition effect need two things
 - CSS property you want to add an effect to
 - Duration of the effect

CSS Transition

- Functions

Property	Description
<u>transition</u>	A shorthand property for setting the four transition properties into a single property
<u>transition-delay</u>	Specifies a delay (in seconds) for the transition effect
<u>transition-duration</u>	Specifies how many seconds or milliseconds a transition effect takes to complete
<u>transition-property</u>	Specifies the name of the CSS property the transition effect is for
<u>transition-timing-function</u>	Specifies the speed curve of the transition effect

CSS Transition

- transition-timing-function
 - ease - specifies a transition effect with a slow start, then fast, then end slowly (this is default)
 - linear - specifies a transition effect with the same speed from start to end
 - ease-in - specifies a transition effect with a slow start
 - ease-out - specifies a transition effect with a slow end
 - ease-in-out - specifies a transition effect with a slow start and end
 - cubic-bezier(x1,y1,x2,y2) - lets you define your own values in a cubic-bezier function. x1,y1,x2,y2 is numeric values must be a number from 0 to 1
- short hand
 - transition-property > transition-duration > transition-timing-function > transition-delay

CSS Transition

```
<!DOCTYPE html>
<html>
<head>
<style>
  div { width: 100px; height: 100px; background-color: orange; }
  div.progress { transition: width 2s; }
  div.progress:hover { width: 300px; }
  div.rotate { transition: width 2s, height 2s, transform 2s; }
  div.rotate:hover { width: 300px; height: 300px;
    transform: rotate(180deg);
  }
</style>
</head>
<body style="padding: 10px;">
  <div class="progress">Progress</div> <br/>
  <div class="rotate">Rotate</div>
</body>
</html>
```

CSS Animation

- The CSS allow animation of HTML elements without using java script

Property	Description
<u>@keyframes</u>	Specifies the animation code
<u>animation</u>	A shorthand property for setting all the animation properties
<u>animation-delay</u>	Specifies a delay for the start of an animation
<u>animation-direction</u>	Specifies whether an animation should be played forwards, backwards or in alternate cycles
<u>animation-duration</u>	Specifies how long time an animation should take to complete one cycle
<u>animation-fill-mode</u>	Specifies a style for the element when the animation is not playing (before it starts, after it ends, or both)
<u>animation-iteration-count</u>	Specifies the number of times an animation should be played.
<u>animation-name</u>	Specifies the name of the @keyframes animation
<u>animation-play-state</u>	Specifies whether the animation is running or paused
<u>animation-timing-function</u>	Specifies the speed curve of the animation

CSS Animation

- **@keyframes** rule hold what styles the element will have at a certain time. The animation will gradually change from the current style to the new style at certain time
- animation-direction
 - **normal** - The animation is played as normal (forwards). This is default
 - **reverse** - The animation is played in reverse direction (backwards)
 - **alternate** - The animation is played forwards first, then backwards
 - **alternate-reverse** - The animation is played backwards first, then forwards

CSS Animation

- animation-fill-mode
 - **none** - Default value. Animation will not apply any styles to the element before or after it is executing
 - **forwards** - The element will retain the style values that is set by the last keyframe (depends on animation-direction and animation-iteration-count)
 - **backwards** - The element will get the style values that is set by the first keyframe (depends on animation-direction), and retain this during the animation-delay period
 - **both** - The animation will follow the rules for both forwards and backwards, extending the animation properties in both directions

CSS Animation

- animation-play-state
 - **paused** – Specifies that the animation is paused
 - **running** – Specifies that the animation is running. This is default value.

CSS Animation

- animation-timing-function
 - **ease** - Specifies an animation with a slow start, then fast, then end slowly (this is default)
 - **linear** - Specifies an animation with the same speed from start to end
 - **ease-in** - Specifies an animation with a slow start
 - **ease-out** - Specifies an animation with a slow end
 - **ease-in-out** - Specifies an animation with a slow start and end
 - **cubic-bezier(x1,y1,x2,y2)** - Lets you define your own values in a cubic-bezier function. x1,y1,x2,y2 is numeric values must be a number from 0 to 1

CSS Animation

- short hand
 - animation-name > animation-duration > animation-timing-function > animation-delay > animation-iteration-count > animation-direction > animation-fill-mode > animation-play-state

CSS Animation

```
<!DOCTYPE html>
<html>
<head>
<style>
@keyframes myanimate {
  0%   {background-color:orange; left:0px; top:0px;}
  25%  {background-color:yellow; left:200px; top:0px;}
  50%  {background-color:red; left:200px; top:200px;}
  75%  {background-color:green; left:0px; top:200px;}
  100% {background-color:blue; left:0px; top:0px;}
}
div { width: 100px; height: 100px; position: relative;
      background-color: orange; border-radius: 50%;
      animation: myanimate 5s linear 1s infinite alternate;
}
</style>
</head>
<body>
  <div></div>
</body>
</html>
```

CSS Media Query

- The **@media** rule made it possible to define different style rule for different media type
- Media query can be used to check
 - width and height of the viewport
 - width and height of the device
 - orientation – is the tablet/phone in landscape or portrait mode
 - resolution

CSS Media Query

- Syntax

- **@media** not|only mediatype and (expressions) { css-code; }
- `<link rel="stylesheet" media="mediatype and|not|only (expressions)" href="media.css">`

- Types

Value	Description
all	Used for all media type devices
print	Used for printers
screen	Used for computer screens, tablets, smart-phones etc.
speech	Used for screen readers that "reads" the page out loud

CSS Media Query

```
<!DOCTYPE html>
<html>
<head>
<style>
div.column { box-sizing: border-box; float: left; width: 25%; padding: 20px; }
@media screen and (max-width: 992px) {
  div.column { width: 50%; }
}
@media screen and (max-width: 600px) {
  div.column { width: 100%; }
}
@media print { h1 { display: none; } }
</style>
</head>
<body>
  <h1>Responsive Column Layout</h1>
  <div class="row">
    <div class="column" style="background-color:#aaa;">
      <h2>Column 1</h2> <p>Some text..</p>
    </div>
    <div class="column" style="background-color:#bbb;">
      <h2>Column 2</h2> <p>Some text..</p>
    </div>
    <div class="column" style="background-color:#ccc;">
      <h2>Column 3</h2> <p>Some text..</p>
    </div>
    <div class="column" style="background-color:#ddd;">
      <h2>Column 4</h2> <p>Some text..</p>
    </div>
  </div>
</body>
</html>
```

CSS Flexbox

- The Flexbox make it easier to design flexible responsive layout structure without using float or positioning
- 4 layout modes
 - block – for sections in a webpage
 - inline – for text
 - table – for two-dimensional table data
 - positioned – for explicit position of an element

CSS Flexbox

- Flexbox consist of flex container and flex items
- Flex container have to setting **display: flex;**

Property	Description
<u>align-content</u>	Modifies the behavior of the flex-wrap property. It is similar to align-items, but instead of aligning flex items, it aligns flex lines
<u>align-items</u>	Vertically aligns the flex items when the items do not use all available space on the cross-axis
<u>flex-direction</u>	Specifies the direction of the flexible items inside a flex container
<u>flex-flow</u>	A shorthand property for flex-direction and flex-wrap
<u>flex-wrap</u>	Specifies whether the flex items should wrap or not, if there is not enough room for them on one flex line
<u>justify-content</u>	Horizontally aligns the flex items when the items do not use all available space on the main-axis

CSS Flexbox

- Flex container
 - flex-direction

Value	Description
row	Default value. The flexible items are displayed horizontally, as a row
row-reverse	Same as row, but in reverse order
column	The flexible items are displayed vertically, as a column
column-reverse	Same as column, but in reverse order

CSS Flexbox

- Flex container
 - flex-wrap

Value	Description
nowrap	Default value. Specifies that the flexible items will not wrap
wrap	Specifies that the flexible items will wrap if necessary
wrap-reverse	Specifies that the flexible items will wrap, if necessary, in reverse order

CSS Flexbox

- Flex container
 - justify-content

Value	Description
flex-start	Default value. Items are positioned at the beginning of the container
flex-end	Items are positioned at the end of the container
center	Items are positioned in the center of the container
space-between	Items will have space between them
space-around	Items will have space before, between, and after them
space-evenly	Items will have equal space around them

CSS Flexbox

- Flex container
 - align-items

Value	Description
normal	Default. Behaves like 'stretch' for flexbox and grid items, or 'start' for grid items with a defined block size.
stretch	Items are stretched to fit the container
center	Items are positioned at the center of the container
flex-start	Items are positioned at the beginning of the container
flex-end	Items are positioned at the end of the container
start	Items are positioned at the beginning of their individual grid cells, in the block direction
end	Items are positioned at the end of the their individual grid cells, in the block direction
baseline	Items are positioned at the baseline of the container

CSS Flexbox

- Flex container
 - align-content

Value	Description
stretch	Default value. Lines stretch to take up the remaining space
center	Lines are packed toward the center of the flex container
flex-start	Lines are packed toward the start of the flex container
flex-end	Lines are packed toward the end of the flex container
space-between	Lines are evenly distributed in the flex container
space-around	Lines are evenly distributed in the flex container, with half-size spaces on either end
space-evenly	Lines are evenly distributed in the flex container, with equal space around them

CSS Flexbox

- Flex items

Property	Description
<u>align-self</u>	Specifies the alignment for a flex item (overrides the flex container's align-items property)
<u>flex</u>	A shorthand property for the flex-grow, flex-shrink, and the flex-basis properties
<u>flex-basis</u>	Specifies the initial length of a flex item
<u>flex-grow</u>	Specifies how much a flex item will grow relative to the rest of the flex items inside the same container. Number default value 0
<u>flex-shrink</u>	Specifies how much a flex item will shrink relative to the rest of the flex items inside the same container. Number default value 1
<u>order</u>	Specifies the order of the flex items inside the same container. Number default value 0

CSS Flexbox

- Flex items
 - flex-basis

Value	Description
<i>number</i>	A length unit, or percentage, specifying the initial length of the flexible item(s)
auto	Default value. The length is equal to the length of the flexible item. If the item has no length specified, the length will be according to its content

CSS Flexbox

- Flex items
 - align-self

Value	Description
auto	Default. The element inherits its parent container's align-items property, or "stretch" if it has no parent container
stretch	The element is positioned to fit the container
center	The element is positioned at the center of the container
flex-start	The element is positioned at the beginning of the container
flex-end	The element is positioned at the end of the container
baseline	The element is positioned at the baseline of the container

CSS Flexbox

```
<!DOCTYPE html>
<html>
<head>
<style>
div.row { display: flex; flex-wrap: wrap; }
div.column { box-sizing: border-box; flex: 25%; padding: 20px; max-width: 25%;}
@media (max-width: 800px) {
  div.column { flex: 50%; max-width: 50%;}
}
@media (max-width: 600px) {
  div.column { flex: 100%; max-width: 100%;}
}
</style>
</head>
<body>
  <h1>Responsive Flexbox</h1>
  <div class="row">
    <div class="column" style="background-color:#aaa;">
      <h2>Column 1</h2> <p>Some text..</p>
    </div>
    <div class="column" style="background-color:#bbb;">
      <h2>Column 2</h2> <p>Some text..</p>
    </div>
    <div class="column" style="background-color:#ccc;">
      <h2>Column 3</h2> <p>Some text..</p>
    </div>
    <div class="column" style="background-color:#ddd;">
      <h2>Column 4</h2> <p>Some text..</p>
    </div>
  </div>
</body>
</html>
```

CSS Grid

- The CSS Grid layout offers a grid-based layout system, with rows and columns making it easier to design web pages without having to use floats and positioning
- A grid layout consists of a parent element with one or more child elements
- A grid container have to setting **display: grid;** or **display: inline-grid;**

CSS Grid

- Grid Properties

Property	Description
<u>column-gap</u>	Specifies the gap between the columns
<u>gap</u>	A shorthand property for the <i>row-gap</i> and the <i>column-gap</i> properties
<u>grid</u>	A shorthand property for the <i>grid-template-rows</i> , <i>grid-template-columns</i> , <i>grid-template-areas</i> , <i>grid-auto-rows</i> , <i>grid-auto-columns</i> , and the <i>grid-auto-flow</i> properties
<u>grid-area</u>	Either specifies a name for the grid item, or this property is a shorthand property for the <i>grid-row-start</i> , <i>grid-column-start</i> , <i>grid-row-end</i> , and <i>grid-column-end</i> properties
<u>grid-auto-columns</u>	Specifies a default column size
<u>grid-auto-flow</u>	Specifies how auto-placed items are inserted in the grid
<u>grid-auto-rows</u>	Specifies a default row size
<u>grid-column</u>	A shorthand property for the <i>grid-column-start</i> and the <i>grid-column-end</i> properties
<u>grid-column-end</u>	Specifies where to end the grid item

CSS Grid

- Grid Properties

Property	Description
grid-column-gap	Specifies the size of the gap between columns
grid-column-start	Specifies where to start the grid item
grid-gap	A shorthand property for the <i>grid-row-gap</i> and <i>grid-column-gap</i> properties
grid-row	A shorthand property for the <i>grid-row-start</i> and the <i>grid-row-end</i> properties
grid-row-end	Specifies where to end the grid item
grid-row-gap	Specifies the size of the gap between rows
grid-row-start	Specifies where to start the grid item
grid-template	A shorthand property for the <i>grid-template-rows</i> , <i>grid-template-columns</i> and <i>grid-areas</i> properties
grid-template-areas	Specifies how to display columns and rows, using named grid items
grid-template-columns	Specifies the size of the columns, and how many columns in a grid layout
grid-template-rows	Specifies the size of the rows in a grid layout
row-gap	Specifies the gap between the grid rows

CSS Grid

```
<!DOCTYPE html>
<html>
<head>
<style>
  div.grid-container { display: grid; grid-template-columns: auto auto auto; gap:
2px; }
  div.grid-item { padding: 20px; background-color:#eee; border: 1px solid #bbb;
  text-align: center; font-size: 40px; }
</style>
</head>
<body>
  <h1>Grid Layout</h1>
  <div class="grid-container">
    <div class="grid-item">1</div>
    <div class="grid-item">2</div>
    <div class="grid-item">3</div>
    <div class="grid-item">4</div>
    <div class="grid-item">5</div>
    <div class="grid-item">6</div>
    <div class="grid-item">7</div>
    <div class="grid-item">8</div>
    <div class="grid-item">9</div>
  </div>
</body>
</html>
```


CSS Vendor Prefix

- Browser vendors used to add prefixed to experimental or nonstandard CSS properties and JavaScript APIs.
- Browser prefixes are used to add new CSS features onto a site that the browsers will support those styles

CSS Vendor Prefix

- CSS prefixes
 - -ms- IE, Edge
 - -moz- Firefox
 - -o- Opera
 - -webkit- Chrome, Safari, iOS, Android

CSS Vendor Prefix

- CSS prefixes

```
<!DOCTYPE html>
<html>
<head>
<style>
  div { width: 100px; height: 100px; background-color: orange; }
  div.rotate {
    -webkit-transition: width 2s, height 2s, transform 2s;
    -moz-transition: width 2s, height 2s, transform 2s;
    -ms-transition: width 2s, height 2s, transform 2s;
    -o-transition: width 2s, height 2s, transform 2s;
    transition: width 2s, height 2s, transform 2s;
  }
  div.rotate:hover { width: 300px; height: 300px;
    -webkit-transform: rotate(180deg);
    -moz-transform: rotate(180deg);
    -ms-transform: rotate(180deg);
    -o-transform: rotate(180deg);
    transform: rotate(180deg);
  }
</style>
</head>
<body style="padding: 10px;">
  <div class="rotate">Rotate</div>
</body>
</html>
```

Reference

- <https://www.w3schools.com/css/default.asp>
- https://www.w3schools.com/html/html_css.asp
- <https://www.w3schools.com/cssref/index.php>
- https://www.w3schools.com/colors/colors_names.asp
- <https://htmlcolorcodes.com/color-names/>
- <https://developer.mozilla.org/en-US/docs/Web/CSS>
- <https://www.htmlcsscolor.com/>



Q & A