“ Learn Programming For Beginners – Free Full Course “

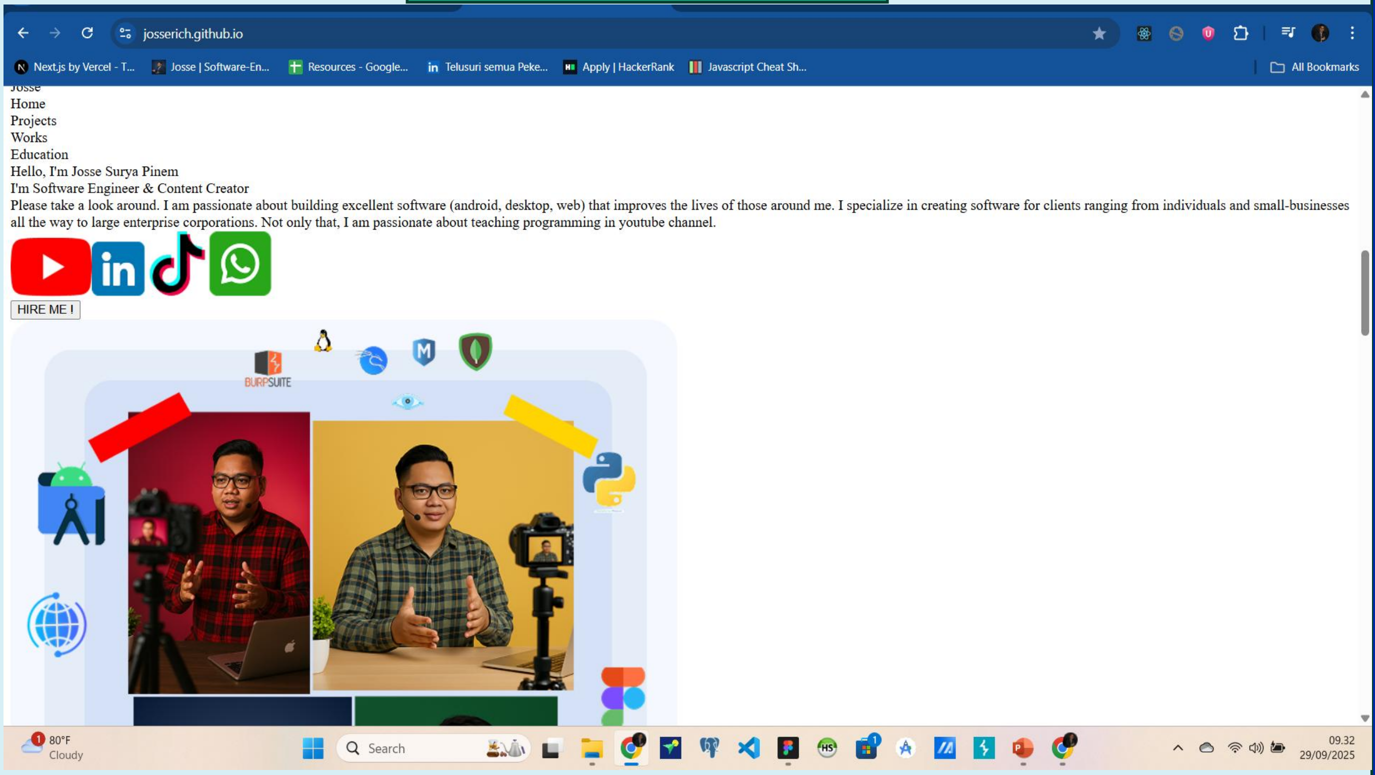
#2 – CSS

CSS stands for Cascading For Style

CSS is the language we use to style an HTML document.

CSS describes how HTML elements should be displayed.

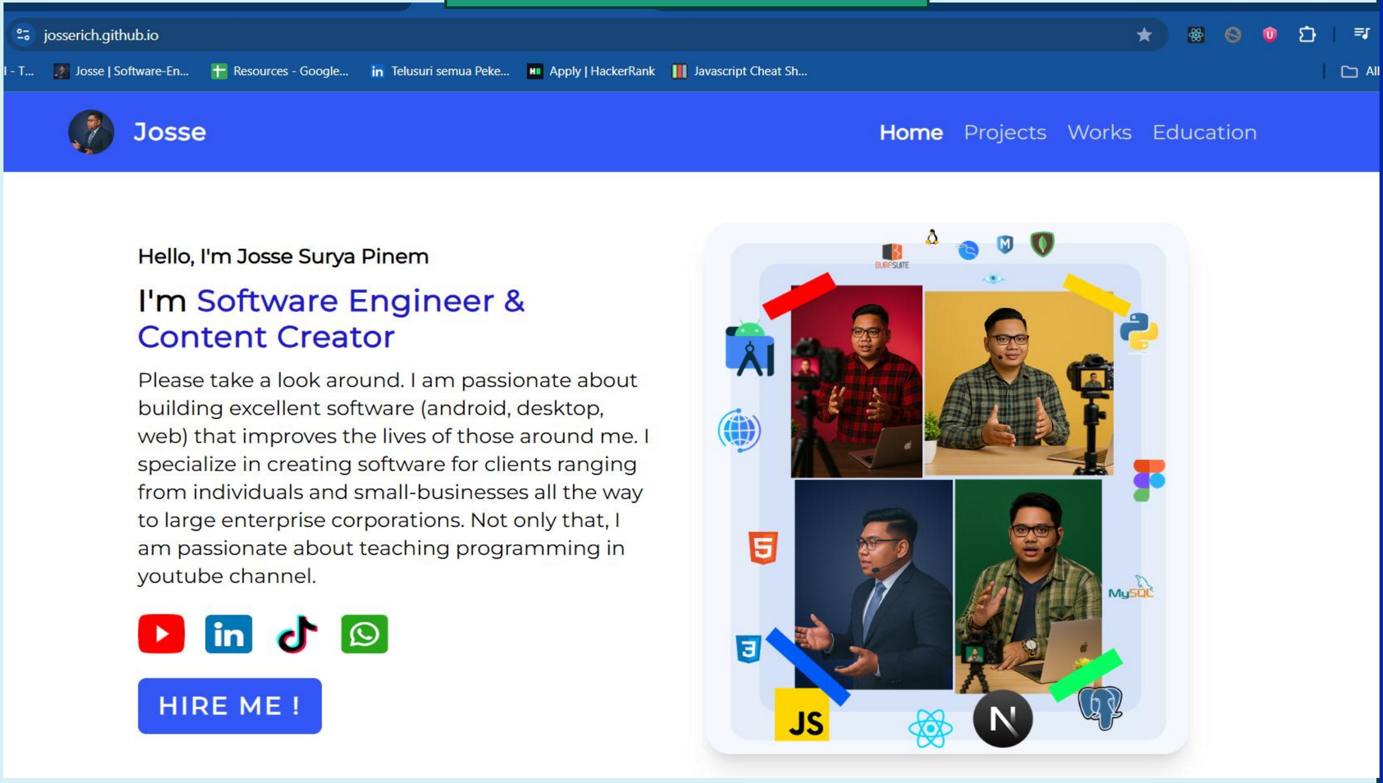
Without CSS


josserich.github.io

Josse
Home
Projects
Works
Education
Hello, I'm Josse Surya Pinem
I'm Software Engineer & Content Creator
Please take a look around. I am passionate about building excellent software (android, desktop, web) that improves the lives of those around me. I specialize in creating software for clients ranging from individuals and small-businesses all the way to large enterprise corporations. Not only that, I am passionate about teaching programming in youtube channel.

HIRE ME !

With CSS


josserich.github.io

Josse
Home Projects Works Education

Hello, I'm Josse Surya Pinem
I'm Software Engineer & Content Creator
Please take a look around. I am passionate about building excellent software (android, desktop, web) that improves the lives of those around me. I specialize in creating software for clients ranging from individuals and small-businesses all the way to large enterprise corporations. Not only that, I am passionate about teaching programming in youtube channel.

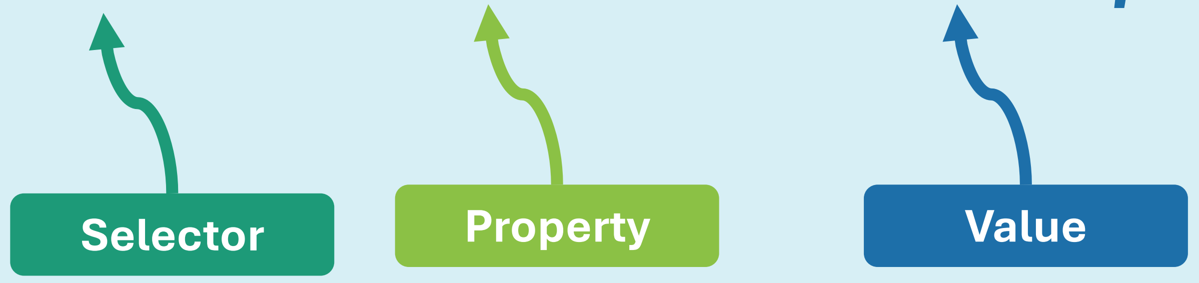
HIRE ME !

https://josserich.github.io

CSS - Syntax

```
Selector { property : value; }
```

```
h1 { color : blue; }
```



HOW TO ADD CSS

1.External CSS (recommended)

```
<head>
    <link rel="stylesheet" href="style.css">
</head>
```

2.Internal CSS

```
<head>
    <style>
        body {
            background-color: red;
        }
        h1 {
            color: blue;
        }
    </style>
</head>
```

3.Inline CSS

```
<h1
style="color:blue;text-
align:center;">
    Hello World
</h1>
```

CSS Selector

CSS selectors are used to "find" (or select) the HTML elements you want to style.

Simple Selectors - based on element HTML, id, class
example : element, # , . .

Combinator Selectors - based on relationship
example : descendant (space), child (>), next sibling (+), subsequent-sibling (~))

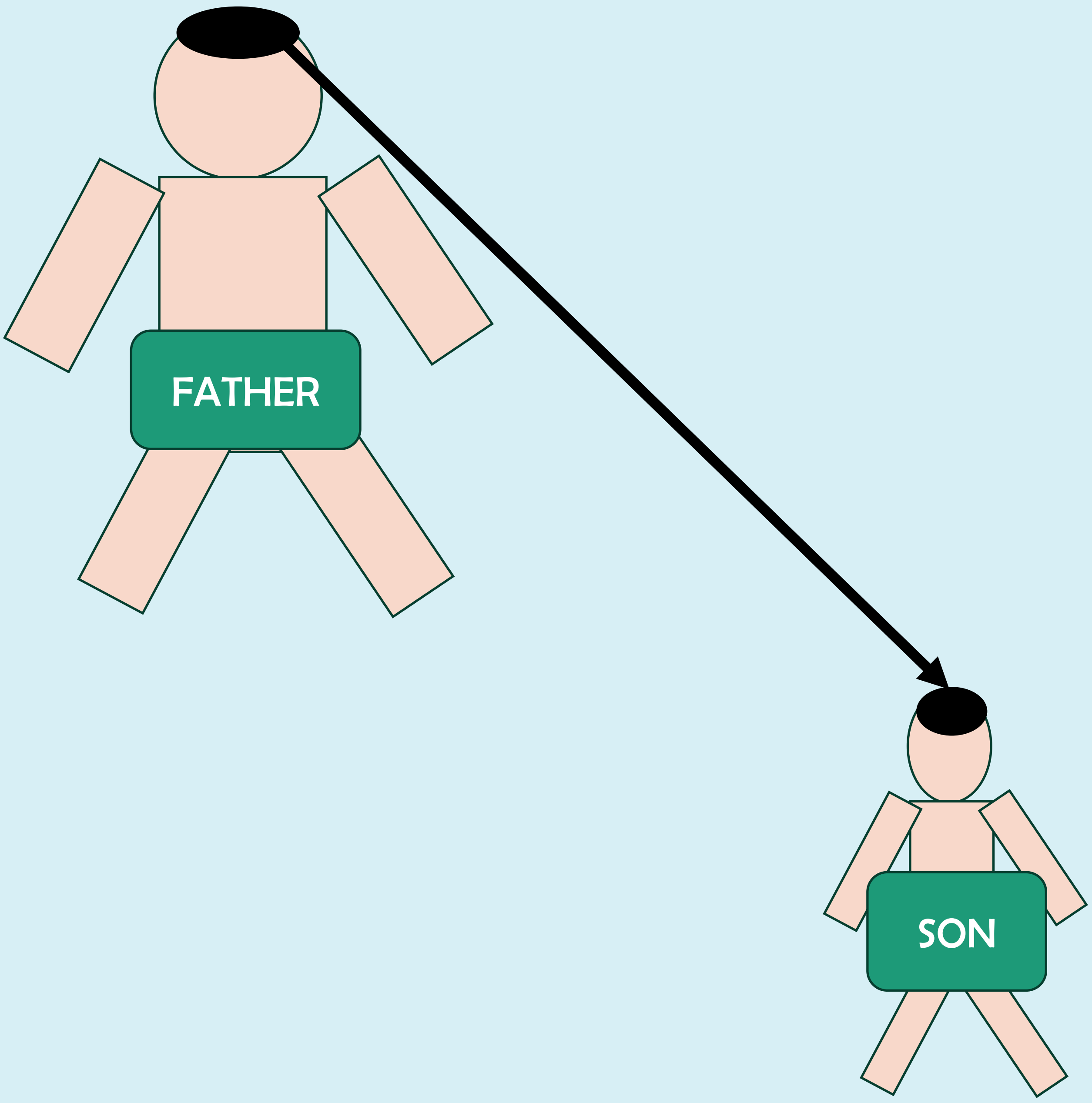
Pseudo-class Selectors - based on certain state
example :
:link, :visited, :hover, :active
input:focus
:first-of-type, :last-of-type
:first-child, :last-child, :nth-child(n), :nth-child(odd | even)

Psuedo-elements Selectors (::) – style a part of element
example :
::first-line
::first-letter
element::before { content:url(); }
element::after { content:url(); }
::backdrop
::selection

[Attribute] Selectors – based on attribute
[attribute],
[attribute="value"],
[atttribute$="value"]

INHERITANCE

an element inherits some values from the properties held by its parent element



Specificity

Each CSS declaration has a different weight. This weight determines how specifically an element can be selected by the selector.

page.html

```
<p>
    Lorem ipsum dolor sit amet
    consectetur adipisicing elit.
    Magnam  eaquedelectus at nostrum
    voluptates quae! Reiciendis
    accusamus illum voluptasut sit
    dolorum esse. Dolores nesciunt,
    neque illo quasi nisi iure.
</p>
```

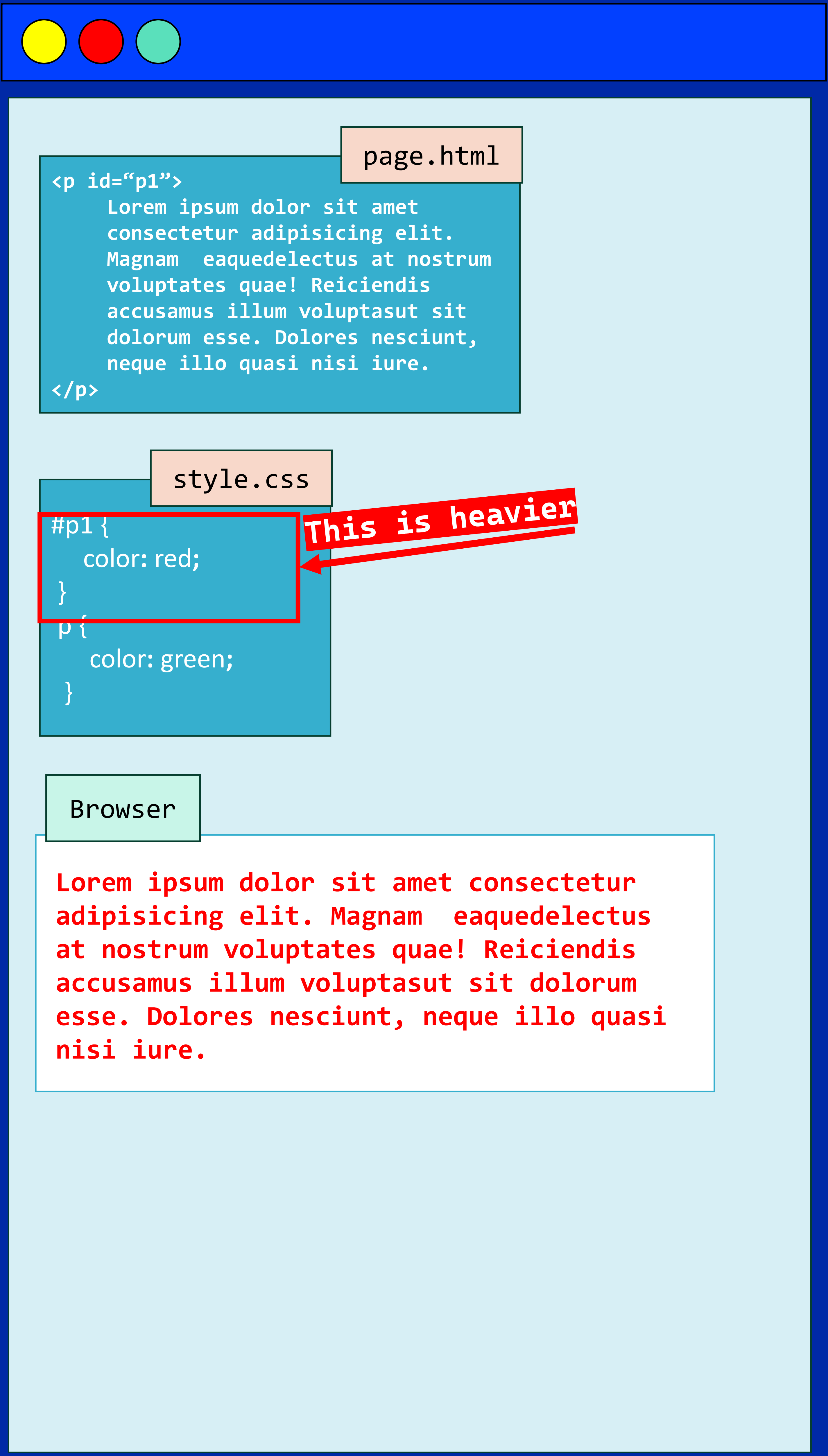
style.css

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

This is heavier

Browser

Lorem ipsum dolor sit amet consectetur adipisicing elit. Magnam eaquedelectus at nostrum voluptates quae! Reiciendis accusamus illum voluptasut sit dolorum esse. Dolores nesciunt, neque illo quasi nisi iure.



```html
<p id="p1">
     Lorem ipsum dolor sit amet
     consectetur adipisicing elit.
     Magnam  eaquedelectus at nostrum
     voluptates quae! Reiciendis
     accusamus illum voluptasut sit
     dolorum esse. Dolores nesciunt,
     neque illo quasi nisi iure.
</p>
```

```css
#p1 {
   color: red;
}
p {
   color: green;
}
```

Lorem ipsum dolor sit amet consectetur adipisicing elit. Magnam eaquedelectus at nostrum voluptates quae! Reiciendis accusamus illum voluptasut sit dolorum esse. Dolores nesciunt, neque illo quasi nisi iure.

Format Calculate Specificity

inline	id	class	element
0	0	0	0

Calculate Specificity CSS

	inline	id	class	element
#p1 = ✓	0	1	0	0

VS

	inline	id	class	element
p =	0	0	0	1

FONT

```
font-family :
 Arial, Helvetica,
sans-serif
font-size:
 [value]px|em|rem.
font-weight :
normal|bold|bolder|ligh
ter|100-900
font-variant :
 normal|small-caps
font-style :
 normal|italic|oblique
line-height :
 normal|px|em|%
```

Google Font

If you do not want to use any of the standard fonts in HTML, you can use Google Fonts.
Google Fonts are free to use, and have more than 1000 fonts to choose from = https://fonts.google.com/

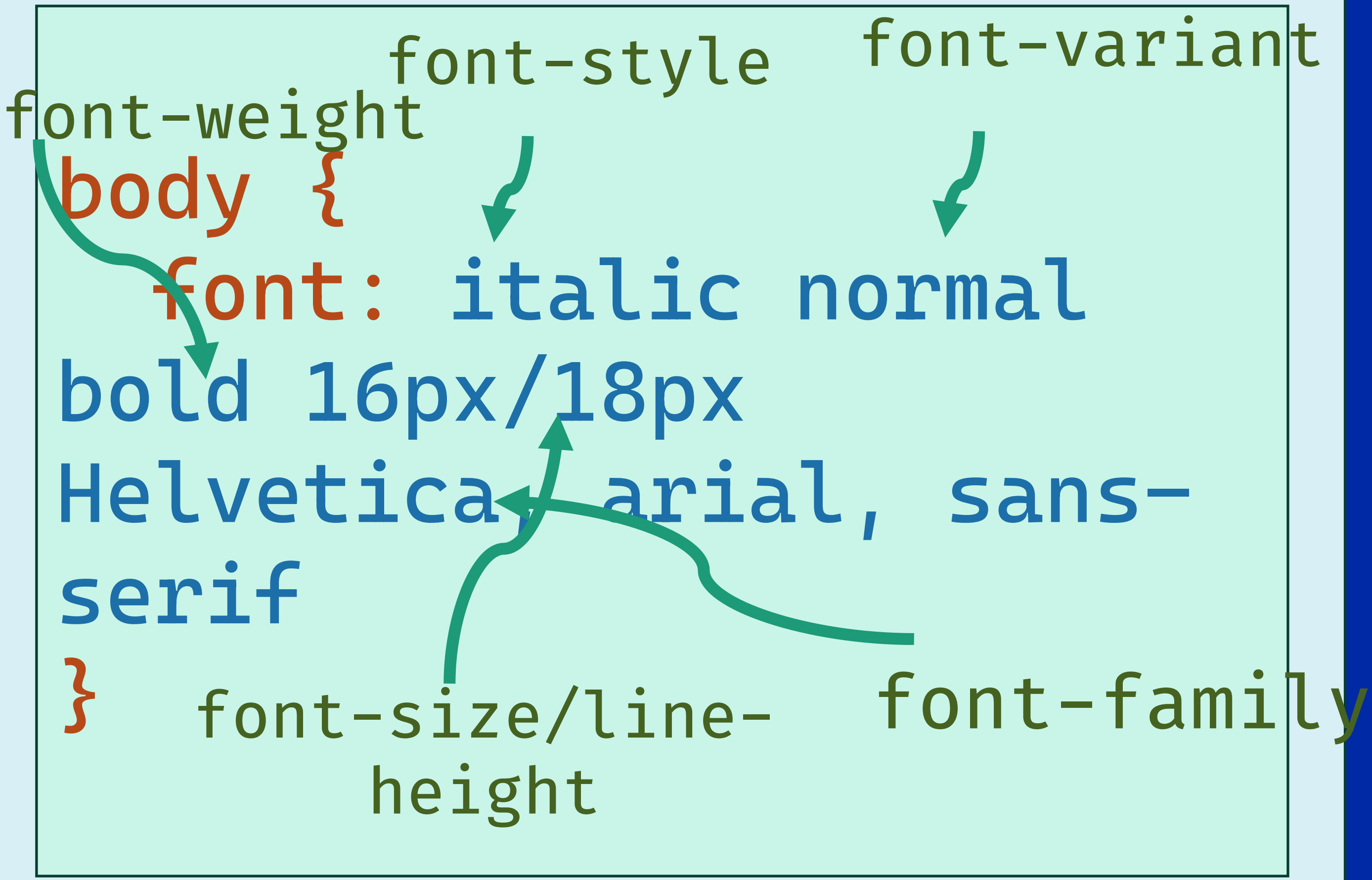
@font-face

```
@font-face {
  font-family: myFirstFont;
  src: url(font.TTF|OTF);
}
p{
 font-family: myFirstFont
}
```

Without Shorthand CSS Font

```
body {
  font-style: italic;
  font-variant: normal;
  font-weight: bold;
  font-size: 16px;
  line-height: 18px;
  font-family:
helvetica,arial;
}
```

With Shorthand CSS Font



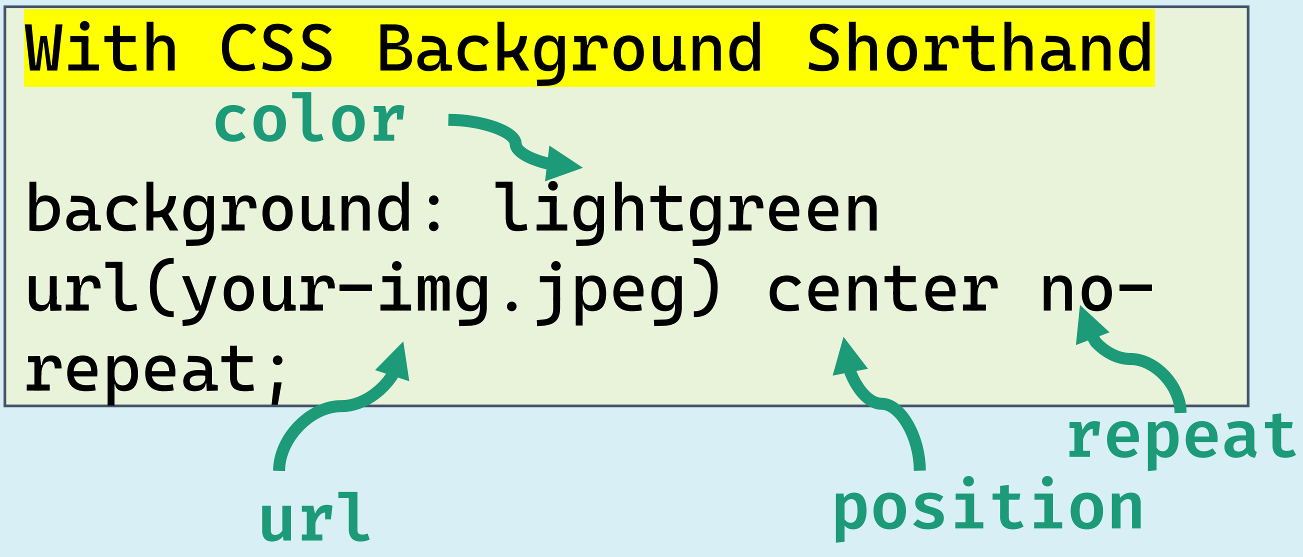
```
body {
  font: italic normal
bold 16px/18px
Helvetica, arial, sans-
serif
}
```

TEXT

```
color:
name|hexadecimal|rgb();
background-color:
name|hexadecimal|rgb();
text-align:
left|center|justify|right;
text-indent: [number]px;
text-decoration:
overline|line-
through|underline;
text-transform:
capitalize|uppercase|lowerca
se;
letter-spacing: [number]px;
word-spacing: [number]px;
text-shadow: h-shadow v-
shadow blur-radius
color|none|initial|inherit;
```

Background

```
background-color:
name|hexadecimal|rgb(255,255,
255);
background-image: url(your-
img.jpeg);
background-position:
left|top|right|bottom|center
background-repeat:
repeat|repeat-x|repeat-y|no-
repeat|space|round
```



With CSS Background Shorthand

```
background: lightgreen
url(your-img.jpeg) center no-
repeat;
```

Display

The display property is an important CSS property for controlling layout
Every HTML element has a default display value, depending on what type of element it is.
The default display value for most elements is block or inline.

block

It stretches out to the left and right as far as it can.
It makes a new line

```
Ex : <div>, <h1> - <h6> ,
<p> , <form>, <header>,
<footer>, <section>
```

inline

It doesn’t stretch out to the left and right as far as it can.
It doesn’t make a new line

```
Ex : <span>, <a>, <img>
```

Block VS Inline

Block

<p> this is an element block </p>

<p> this is an element block </p>

Inline

<span> this is an element inline </span> <span> this is an element inline </span>

Display Value

inline

it doesn’t make a new line.
It doesn’t stretch out to the left and right
we can’t set width and height except element image

Inline-block

There is no html element that defaults to inline-block.
It is similar with inline but it can apply width , height

block

It stretches out to the left and right as far as it can.
It makes a new line
it can set width and height

none

The element is completely hidden from the document flow (does not take up any space).

flex

Displays an element as a block-level flex container

grid

Displays an element as a block-level grid container

WIDTH & HEIGHT

auto – this is default,
length - Defines the height or width in px, cm, em, etc.
% - Defines the height or width in percent of the containing block
initial - Sets the height or width to its default value
inherit - The height or width will be inherited from its parent value

Overflow

The CSS overflow property controls what happens to content that is too big to fit into an 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 is hidden
scroll - Scrollbars are added. User must scroll to see all content
auto - Similar to scroll, but adds scrollbars only when necessary

Box Model

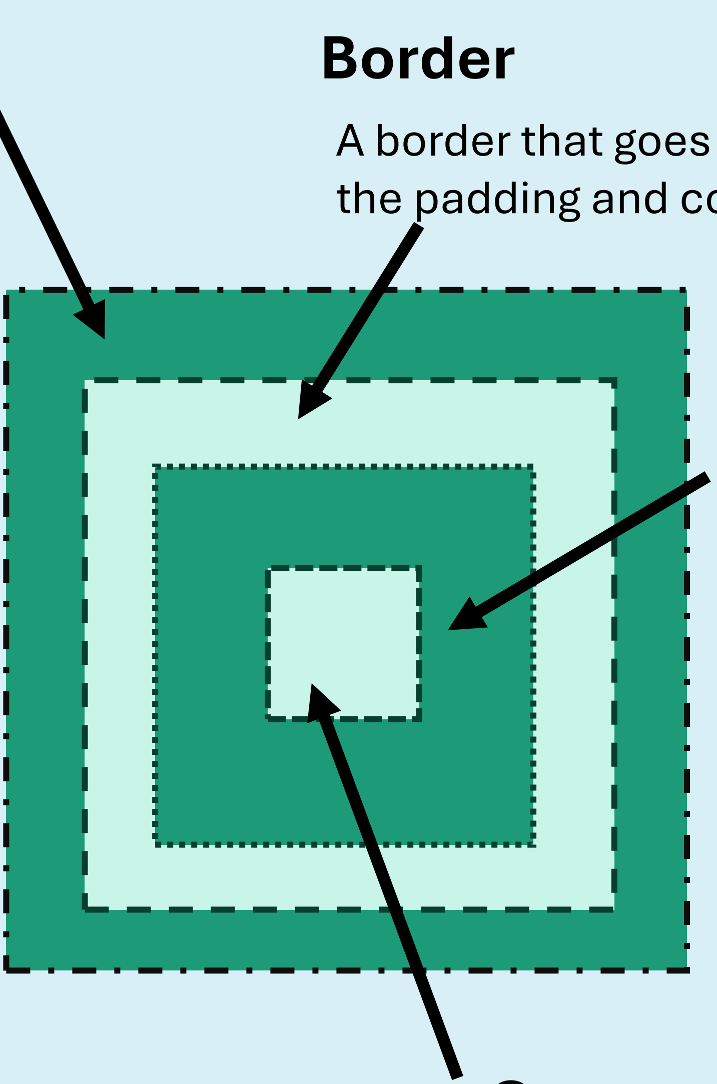
The CSS box model is essentially a box that wraps around every HTML element.

Margin

Clears an area outside the border. The margin is transparent

Border

A border that goes around the padding and content



Padding

Clears an area around the content. The padding is transparent

Content

Where text and images appear

Margin

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

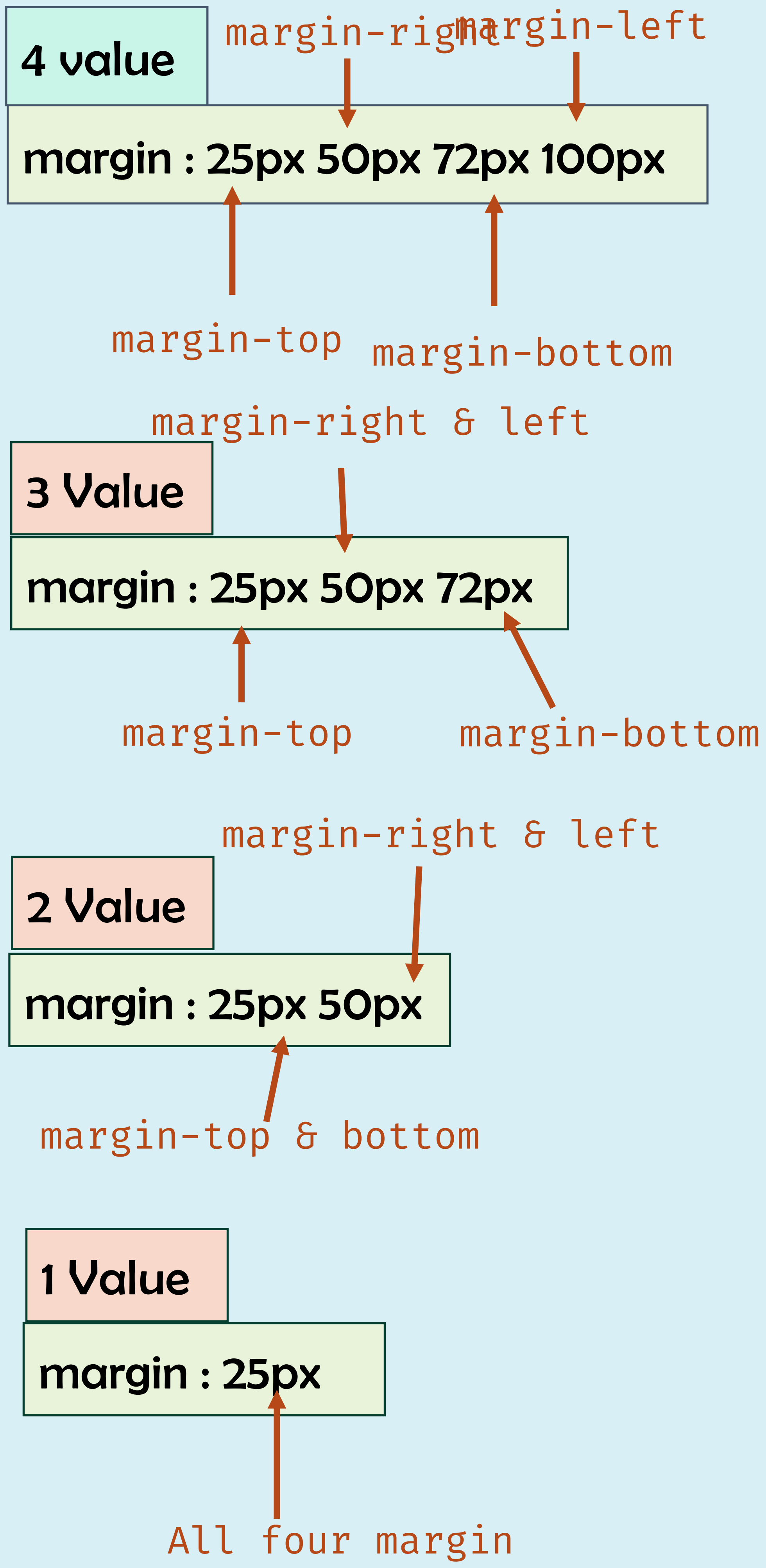
Properties Margin :
margin-top
margin-right
margin-left
margin-bottom

Value Margin :

auto - the browser calculates the margin

length - specifies a margin in px, pt, cm, etc.
% - specifies a margin in % of the width of the containing element
inherit - specifies that the margin should be inherited from the parent element

Margin – Shorthand Property


4 value

```
margin : 25px 50px 72px 100px
```

3 Value

```
margin : 25px 50px 72px
```

2 Value

```
margin : 25px 50px
```

1 Value

```
margin : 25px
```

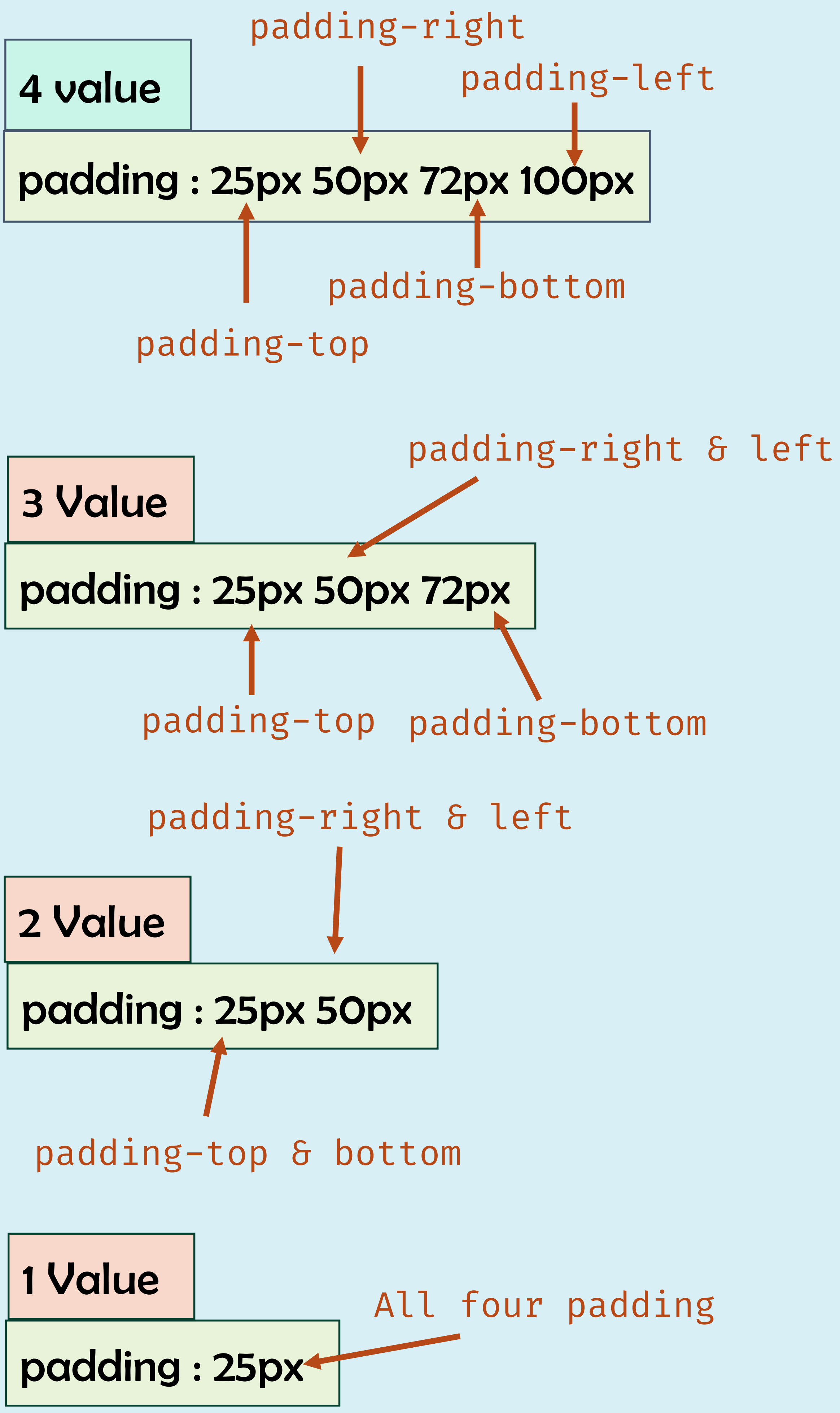
Padding

The CSS padding properties are used to generate space around an element's content, inside of any defined borders.

```
Properties Padding :
padding-top
padding-right
padding-left
padding-bottom
```

```
Value Padding :
length - specifies a margin in px, pt, cm, etc.
% - specifies a margin in % of the width of the containing element
inherit - specifies that the margin should be inherited from the parent element
```

Padding – Shorthand Property



4 value

```
padding : 25px 50px 72px 100px
```

3 Value

```
padding : 25px 50px 72px
```

2 Value

```
padding : 25px 50px
```

1 Value

```
padding : 25px
```

Padding Isn’t same with margin

Padding has no negative value

Padding has no auto value

Border

The CSS border properties allow you to specify the style, width, and color of an element's border.

```
border-style : dotted / dashed / solid
/ double / groove / ridge / inset /
outset / none / hidden
```

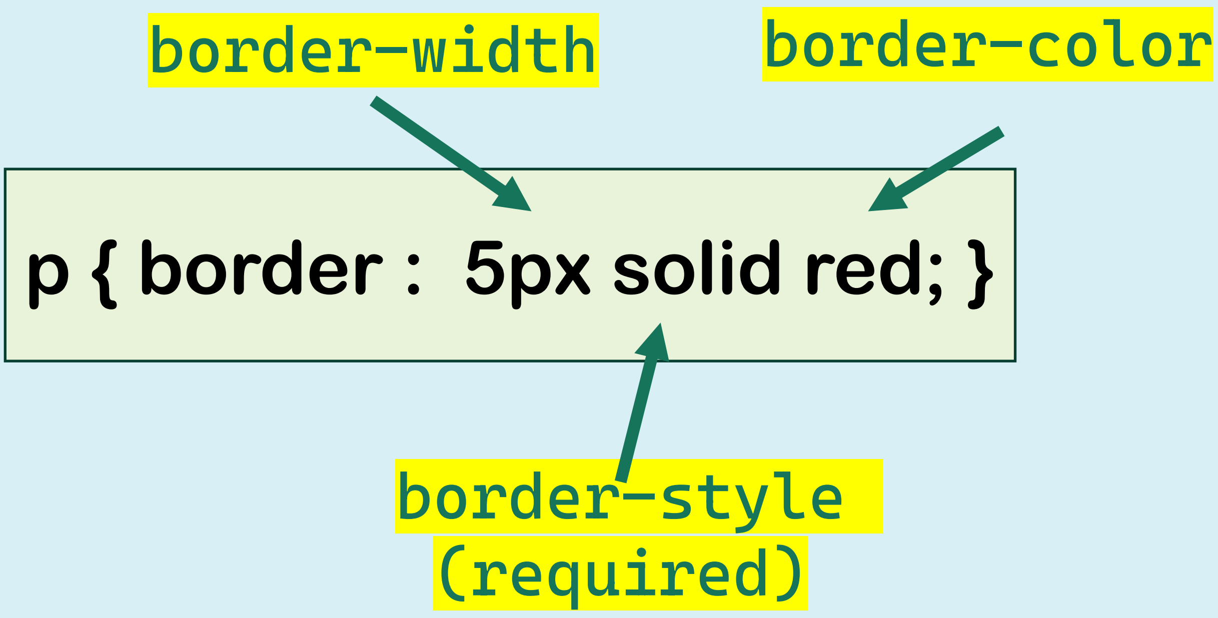
```
border-width : size (in, px, pt, cm,
em, etc) / thin / medium / thick
```

```
border-color : name / HEX / RGB / HSL
/ transparent
```

```
border-top-style / border-right-style /
border-bottom-style / border-left-style :
value
```

```
border-radius : [value]px
```

Border – Shorthand Property



```
p { border :  5px solid red; }
```

Box-sizing:border-box

The box-sizing property allows us to include the padding and border in an element's total width and height.
If you set box-sizing: border-box; on an element, padding and border are included in the width and height:

Example

div1 has width 300px , height 50px

div2 has width 300px, height 50px, padding 50px

```
Box-sizing:border-box;
```

Same size

Hooraay!

Box Shadow

The box-shadow property attaches one or more shadows to an element.

Syntax :

```
box-shadow: none|h-offset v-offset
blur spread
color |inset|initial|inherit;
```

CSS - Position

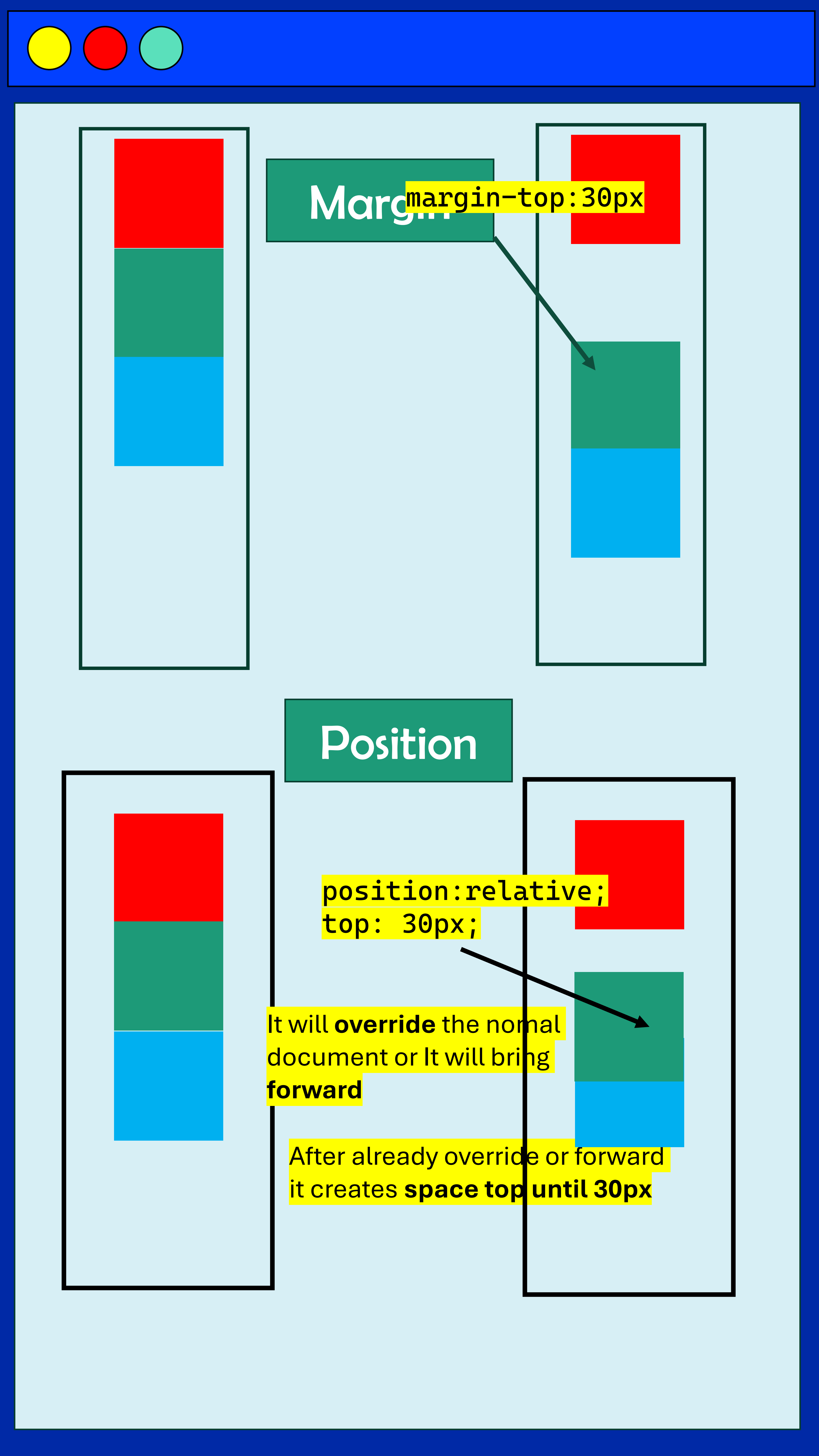
CSS positioning is about **controlling the placement of elements** within a web page
With CSS positioning, you can **override** the normal document flow.

Syntax

```
element{
 position : relative;
 top: [number]px
}
```

static – default
relative
fixed
absolute
sticky

top
bottom
left
right



Margin

```
margin-top:30px
```

Position

```
position:relative;
top: 30px;
```

It will **override** the normal document or It will bring **forward**

After already override or forward it creates **space top until 30px**

Position - Relative

An element with position: relative; is positioned relative to its normal position in the document flow.
Setting the top, right, bottom, and left properties will cause the element to be adjusted away from its normal position. Other content will not be adjusted to fit into any gap left by the element.

Position - Absolute

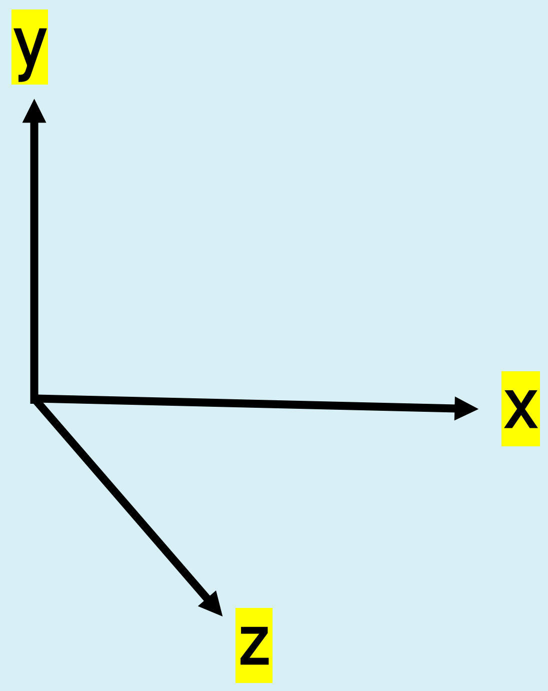
An element with position: absolute; is positioned relative to the nearest positioned ancestor (with position other than static).
However; if an absolute positioned element has no positioned ancestors, it uses the document body, and moves along with page scrolling.

Position - Sticky

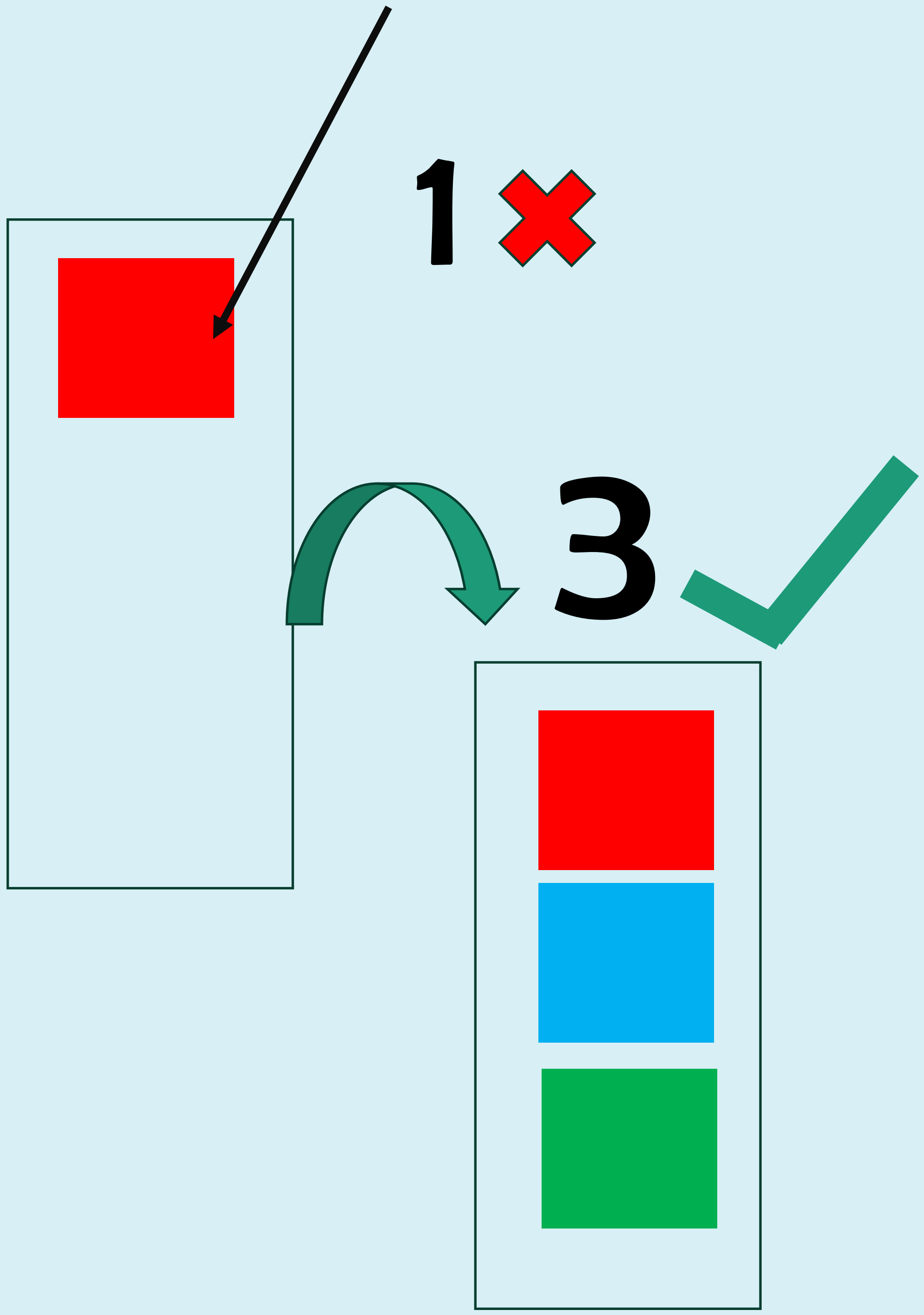
An element with position: sticky; toggles between a relative and fixed position, depending on the scroll position.
A sticky element is positioned relative until a certain scroll position is reached - then it "sticks" in that place (like position:fixed).

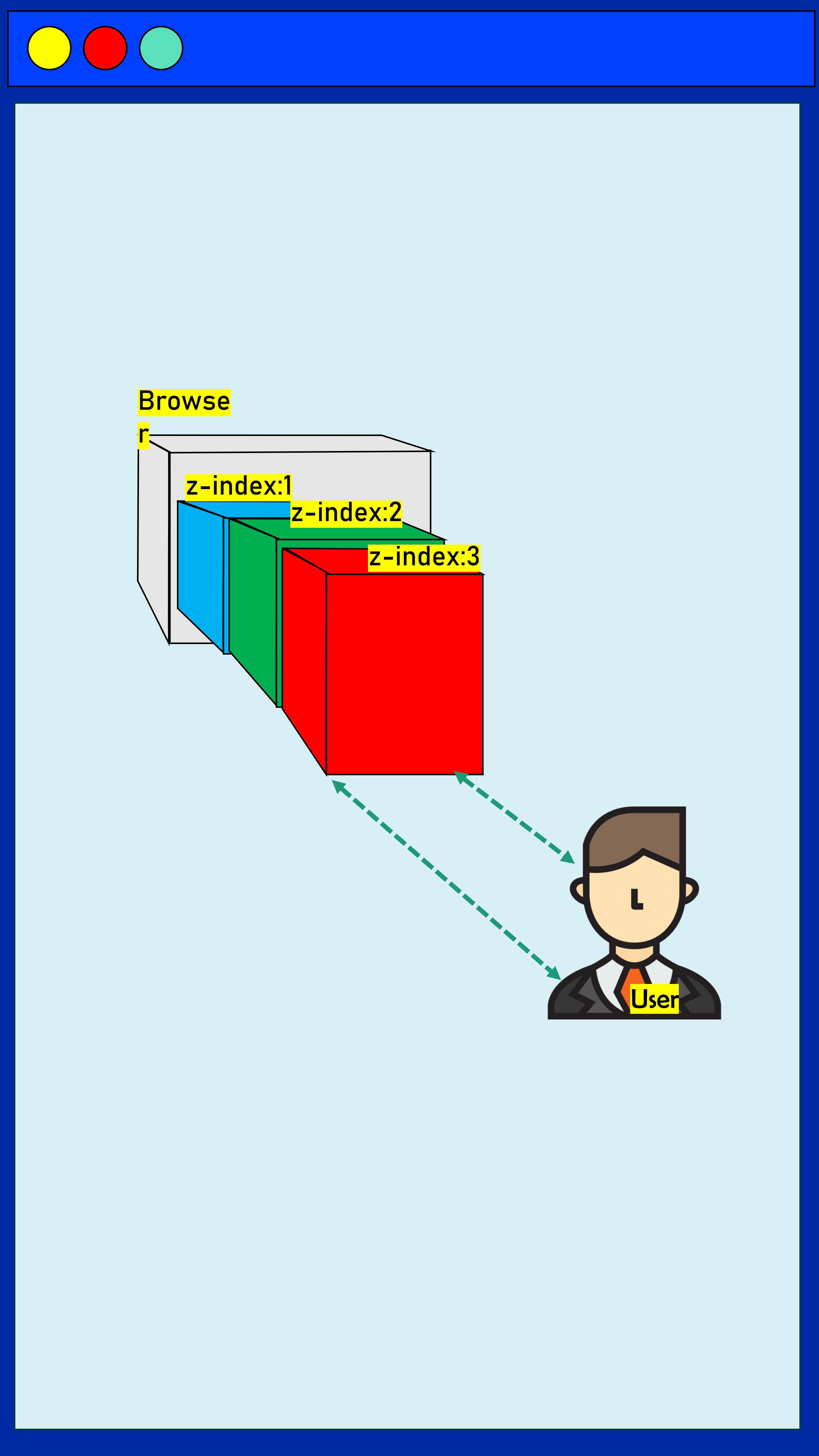
Z-index

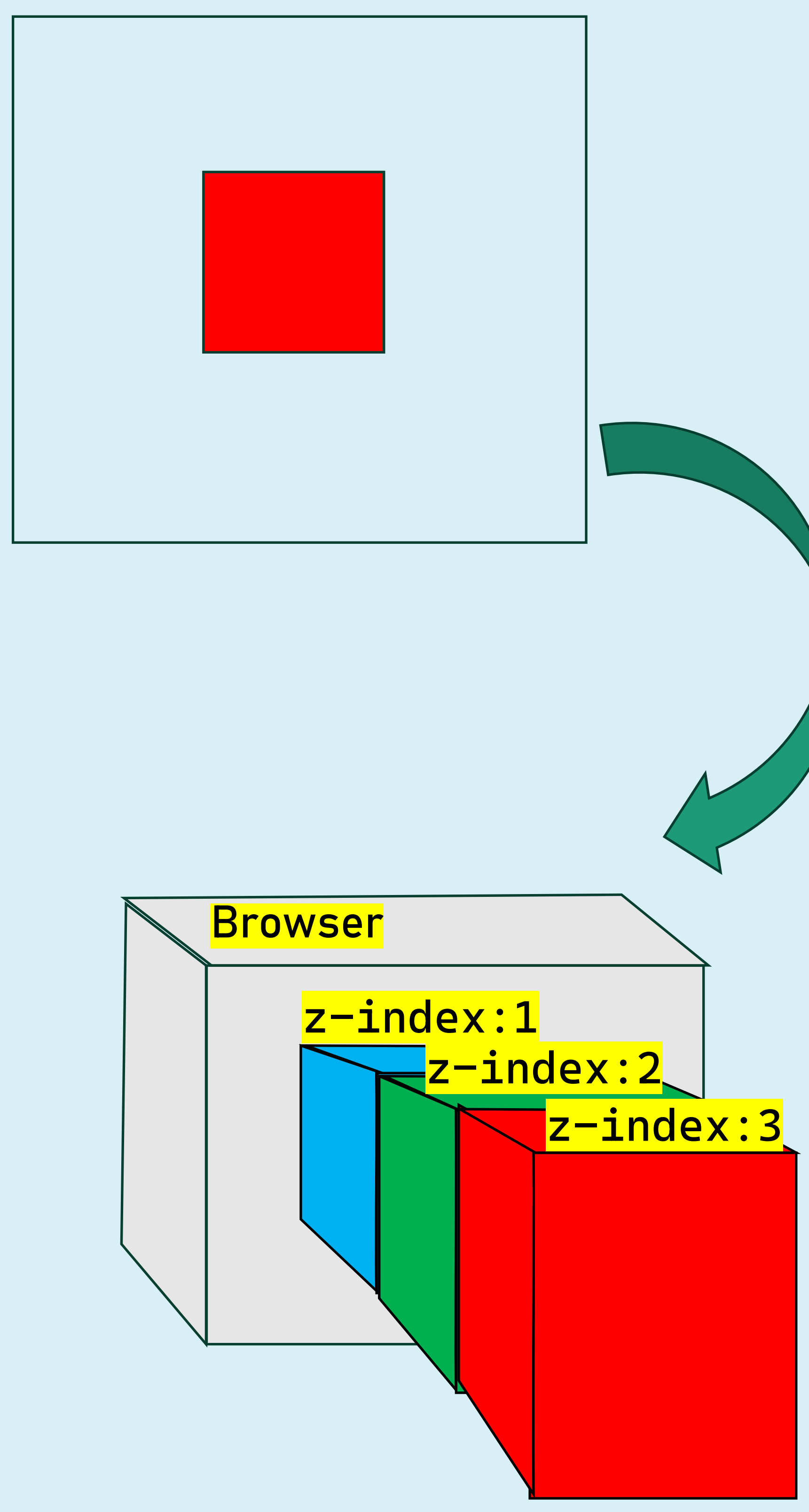
The z-index property specifies the stack order of positioned elements.
The stack order defines which **element should be placed in front or behind other elements.**
When elements are positioned, they can **overlap** other elements.



How many square element this ?




Browse
r
z-index:1
z-index:2
z-index:3
User


Browser
z-index:1
z-index:2
z-index:3

CSS - OPACITY

The opacity property sets the opacity level for an element.
The opacity-level **describes the transparency-level**, where 1 is not transparent at all, 0.5 is 50% see-through, and 0 is completely transparent.

```
opacity: number|initial|inherit;

number :  Specifies the opacity. From 0.0
(fully transparent) to 1.0 (fully opaque)
```

CSS - GRADIENT

The CSS gradient functions let you display smooth transitions between two or more colors within an element.

Linear Gradients : The color transition goes down, up, left, right, or diagonally

```
sytanx :
background-image: linear-
gradient(direction, color-stop1, color-stop2,
 ... );
```

Radial Gradients : The color transition goes out from a central point

```
sytanx :
background-image: radial-gradient(shape size
at position, start-color, ..., last-color)
parameter :
shape : ellipse(default)|circle.
size : farthest-corner(this is default)|closest-
side|farthest-side|closest-corner.
position :
center(default)|top|right|bottom|left|axis-x|axis-y
```

Conic Gradients : The color transition is rotated around a center point

```
syntax :
background-image: conic-
gradient([from angle]
[at position,] color [degree],
color [degree], ... );
```

Filter

The filter property defines **visual effects** (like blur and saturation) to an element (often <img>).

```
Syntax :
filter: none | blur() | brightness() |
contrast() | drop-shadow() | grayscale()
| hue-rotate() | invert() | opacity() |
saturate() | sepia() | url();
```

Transform

The transform property applies a **2D or 3D** transformation to an element. This property allows you to **rotate, scale, move, skew, etc**., elements.

2D:

```
transform :
translate(),
rotate(),
scaleX(),
scaleY(),
scale(), skewX(),
skew(), matrix()
```

3D:

```
transform :
rotateX(), rotateY(),
rotateZ()
```

Transition

CSS transitions allows you to change property values **smoothly**, over a given duration.

Property

The CSS transition property is a shorthand property for :

- transition-property (required)
- transition-duration (required)
- transition-timing-function
  - ease (default), linear, ease-in, ease-in-out, cubic-bezier
- transition-delay

CSS - Animation

An animation lets an element gradually change from one style to another.
You can change as many CSS properties you want, as many times as you want.

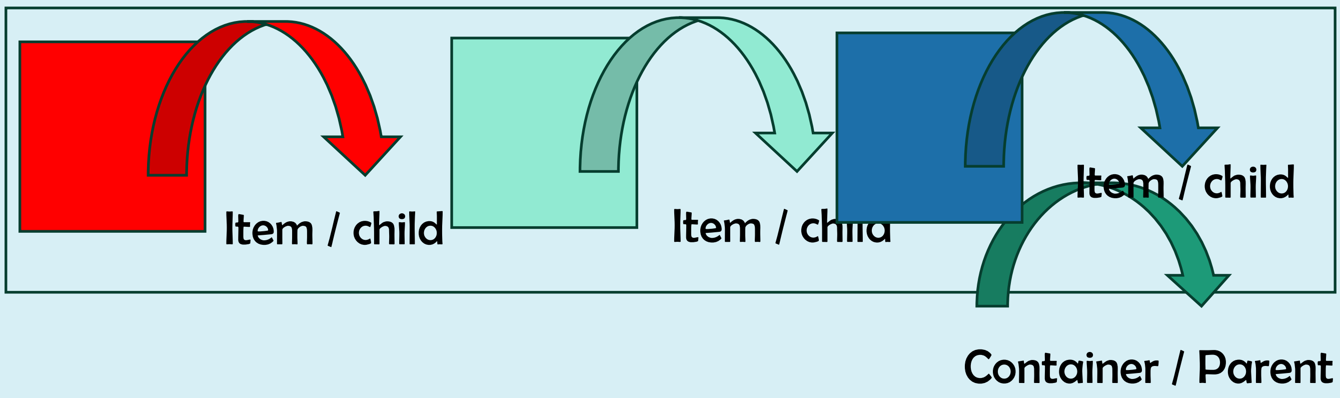
```
div {
   animation-name: [name];
   animation-duration:3s;
   animation-timing-function:
    linear|ease|ease-in|ease-in-out|ease-
    out|ease-in-out|step-start|step-
    end|steps(int,step-position)}cubic-
    Bezier(n,n,n,n);
   animation-delay: 2s;
   animation-iteration-count:number|infinite;
   animation-direction:
    reverse|alternate|alternate-reverse;
}

@keyframes [name] {
 from {
    sytanx-css
  }
 to {
    sytanx-css
  }
}
```

CSS Flexbox (Flexible Box Layout)

CSS Flexbox is short for the CSS Flexible Box Layout Module.
Flexbox is a layout **model for arranging items (horizontally or vertically) within a container**, in a flexible and responsive way
Flexbox makes it easy to design a flexible and responsive layout, without using float or positioning.
CSS Flexbox is used for **a one-dimensional layout, with rows OR columns.**

Css Flexbox Components



CSS – Flexbox Container Properties

Must set! & make sure container is bigger than item

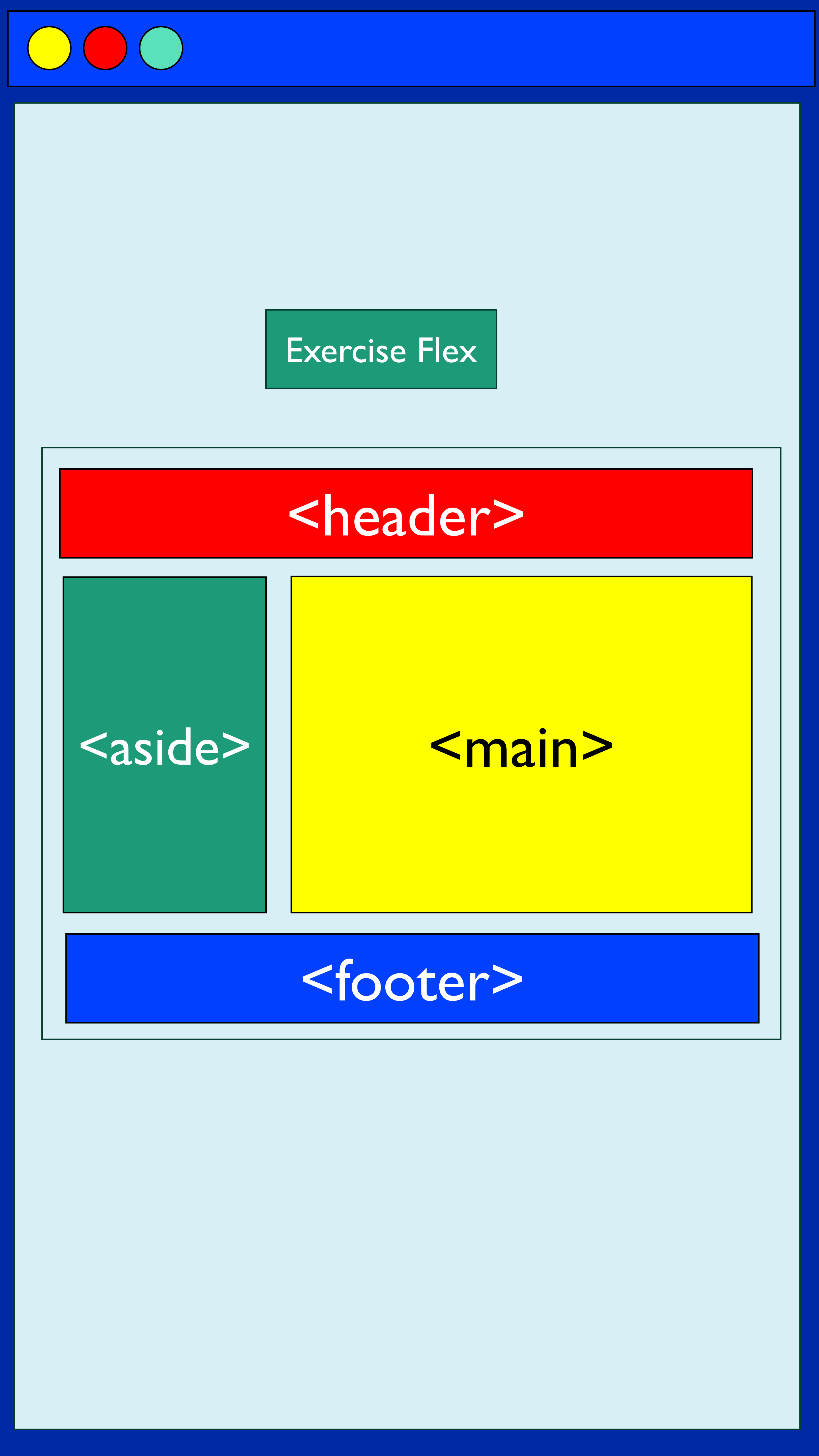
display: flex or inline-flex;
flex-direction: row(default)|column|row-reverse|column-reverse
flex-wrap: nowrap(default)|wrap|wrap-reverse
flex-flow: flex-direction flex-wrap

Horizontally
justify-content: center|flex-start(default)|flex-end|space-around|space-between|space-evenly
justify-items: center|flex-start|flex-end|stretch|baseline|normal(default)

Vertically
align-content: center|stretch(default)|flex-start|flex-end|space-around|space-between|space-evenly
align-items: center|flex-start|flex-end|stretch|baseline|normal(default)

CSS Flex Items

align-self: center|flex-start|flex-end|auto|stretch|baseline|initial|inherit;
justify-self: center|start|end|auto|normal|stretch
place-self: align-self justify-self
order: [number]:
flex-shrink: [number];
flex-grow: [number];
flex-basis: [number]px;


Exercise Flex
<header>
<aside>
<main>
<footer>

FLEX

index.html

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport"
content="width=device-width,
initial-scale=1.0" />
    <title>Document</title>
    <link rel="stylesheet"
href="style.css" />
  </head>
  <body>
    <header>Header</header>
    <div>
      <aside>Aside</aside>
      <main>Main</main>
    </div>
    <footer>Footer</footer>
  </body>
</html>
```

It doesn't have a meaning or
not semantic
it's not good for maintaining

GRID

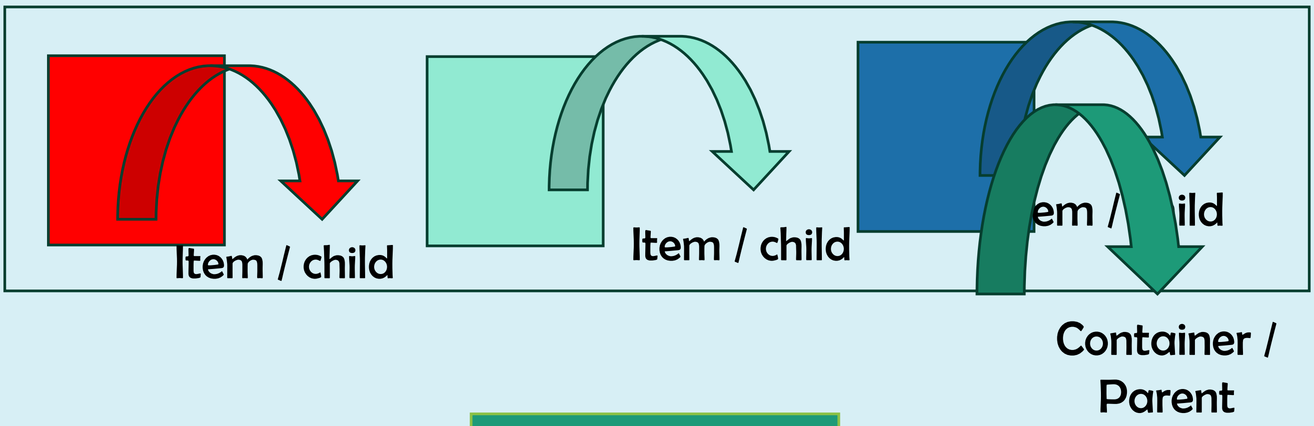
index.html

```html
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport"
content="width=device-width,
initial-scale=1.0" />
    <title>Document</title>
    <link rel="stylesheet"
href="style.css" />
  </head>
  <body>
    <header>Header</header>
    <aside>Aside</aside>
    <main>Main</main>
    <footer>Footer</footer>
  </body>
</html>
```

CSS Grid

The Grid Layout Module offers a grid-based layout system, with rows and columns.
The Grid Layout Module makes it easier to design a responsive layout structure,

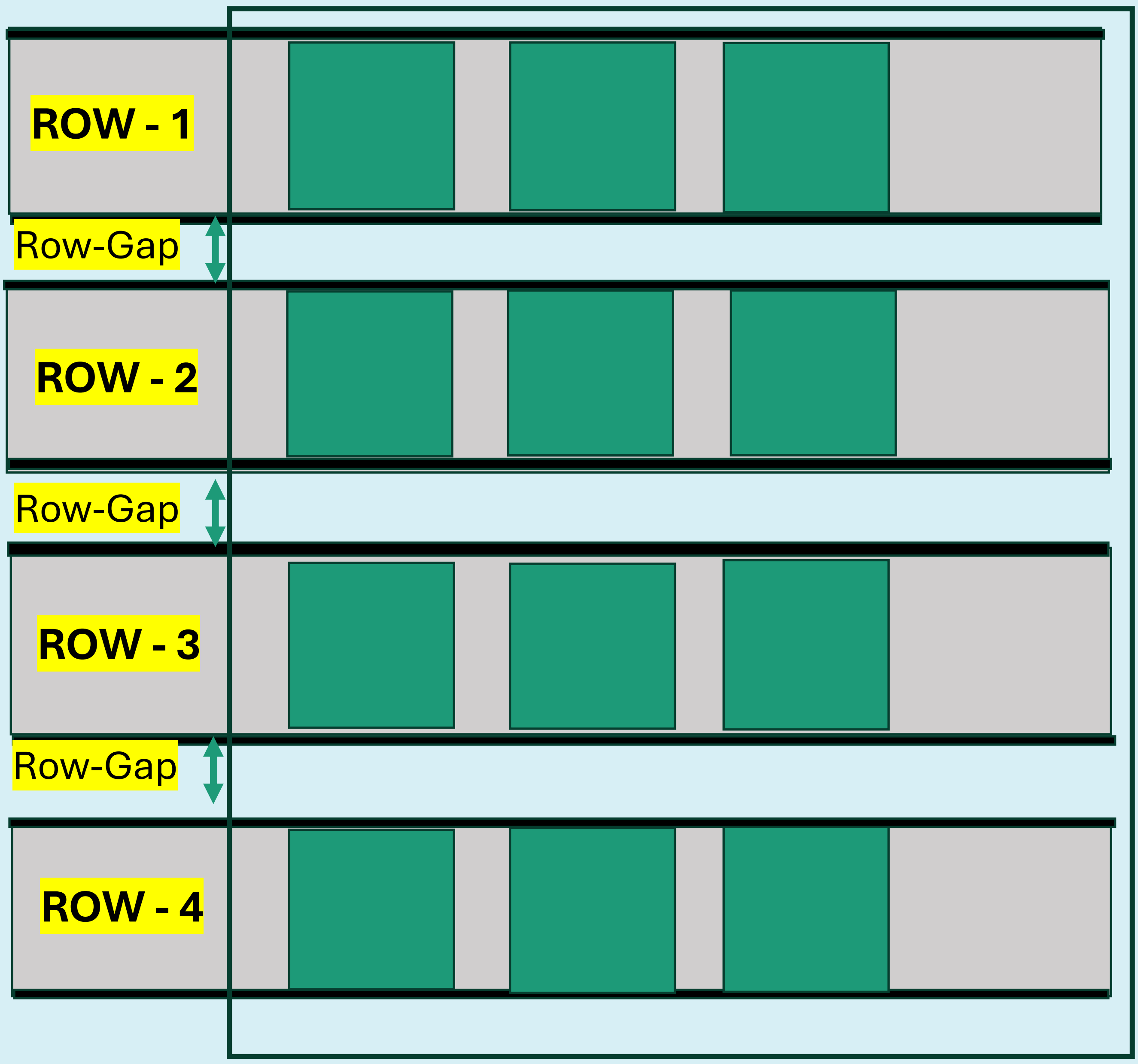
Css Grid Components

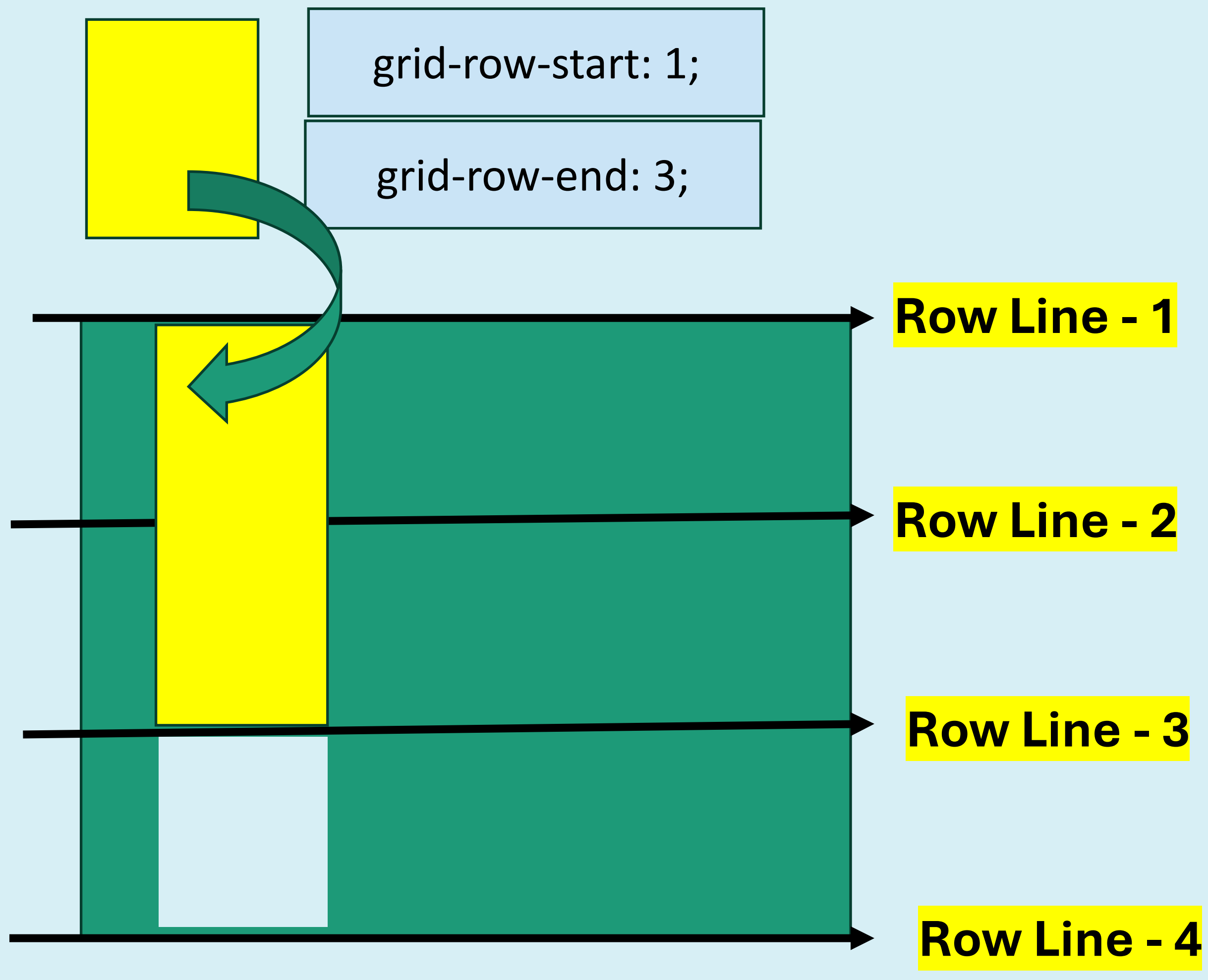


Grid vs Flex

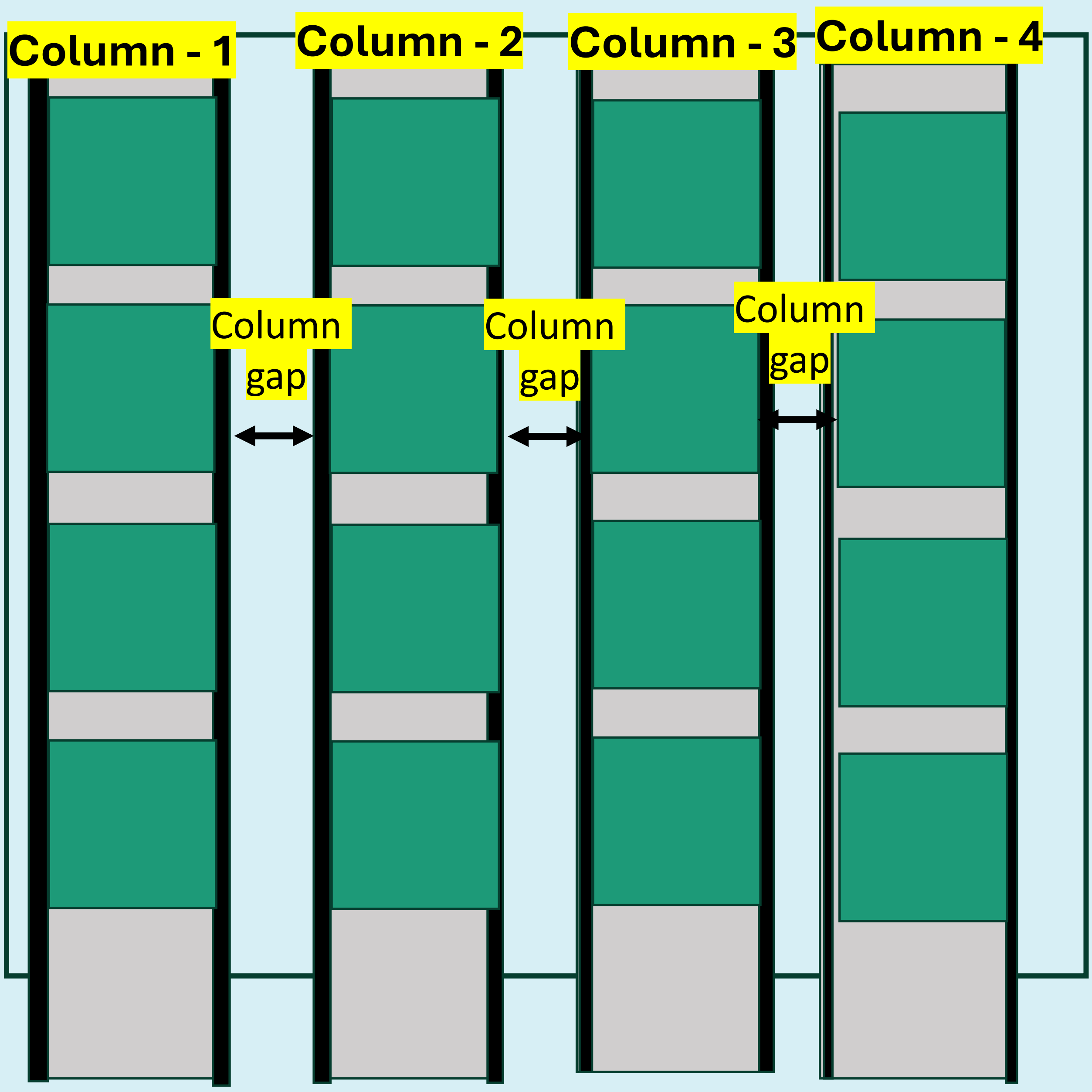
Css Grid is used for two dimensional layout, with rows AND columns
Css Flex is used for one dimensional layout, with rows OR columns

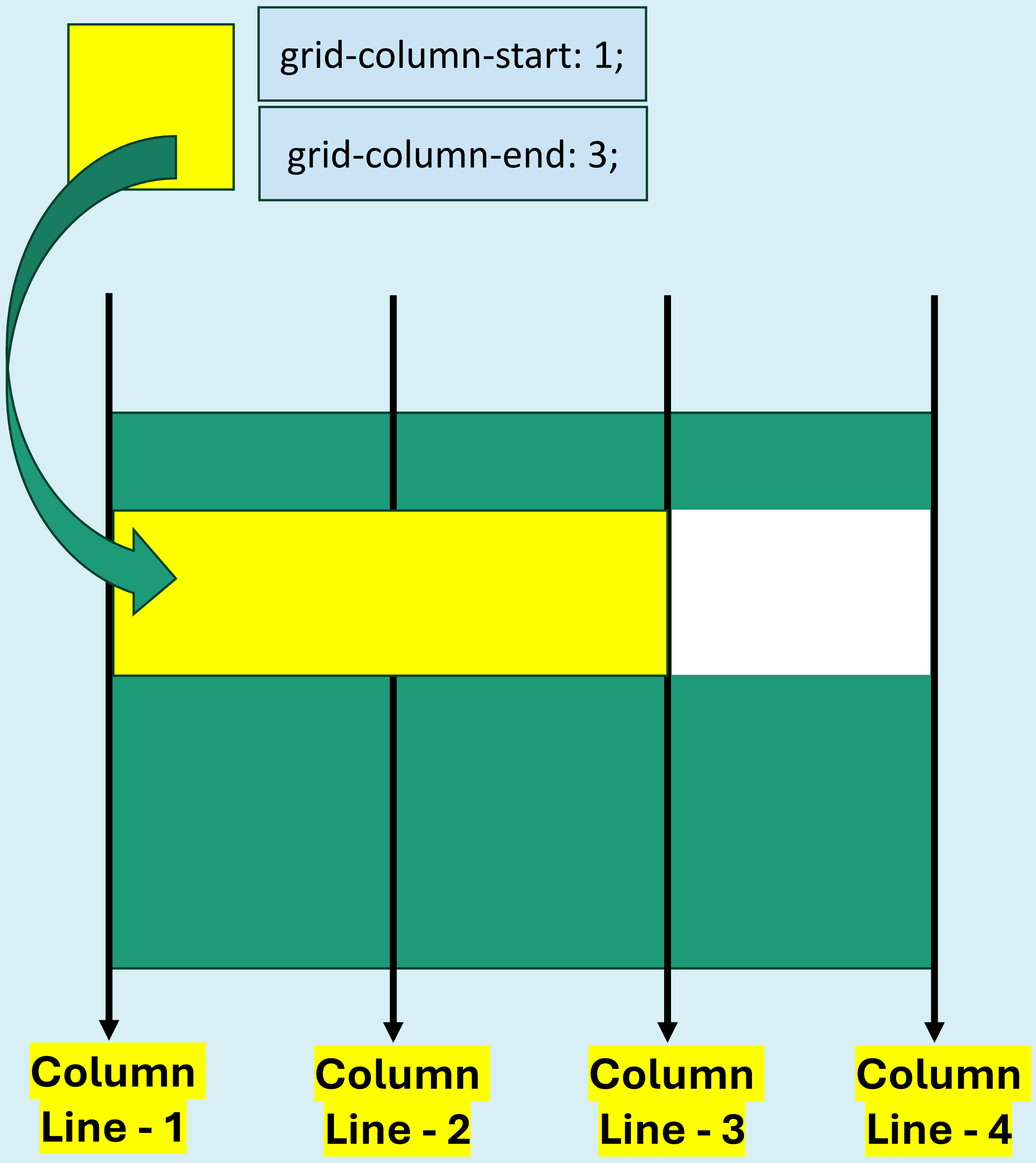
Row, Row-Gap




grid-row-start: 1;
grid-row-end: 3;
Row Line - 1
Row Line - 2
Row Line - 3
Row Line - 4

GRID COLUMN, COLUM GAP

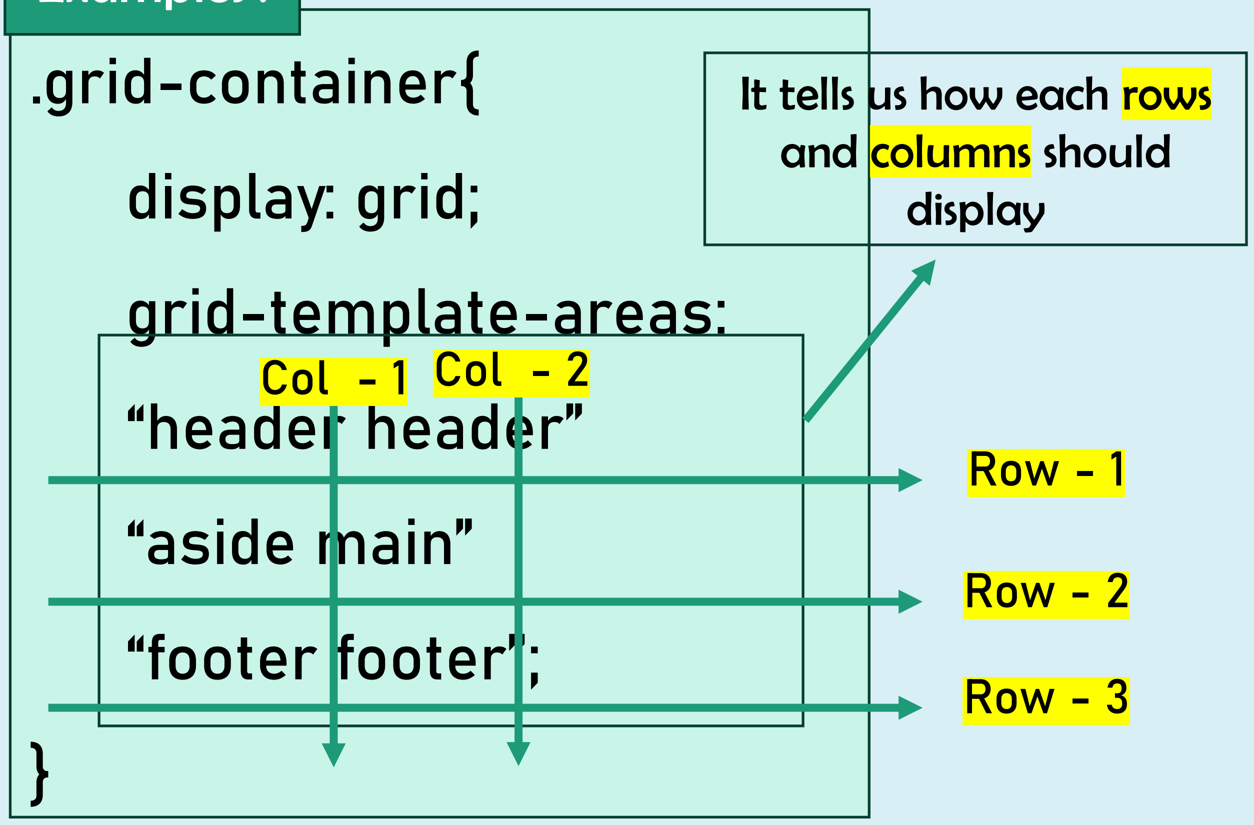



```
grid-column-start: 1;
grid-column-end: 3;
```

Grid Templates Area

The grid-template-areas property specifies areas within the grid layout
You can name grid items by using the grid-area property, and the reference to the name in the grid-template-areas property
Each area is define by apostrophes. Use a period sign to refer to a grid item with no name

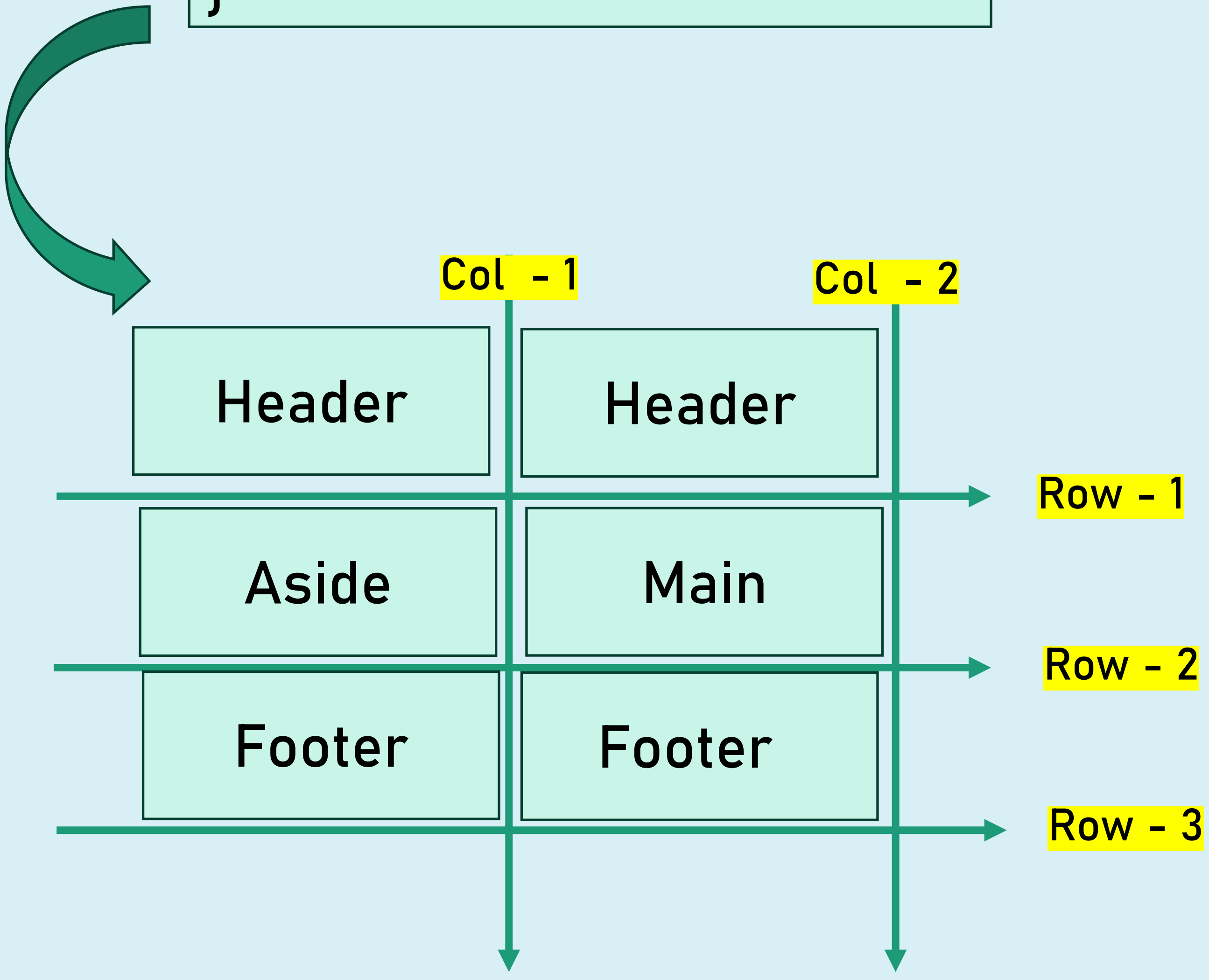
Examples :

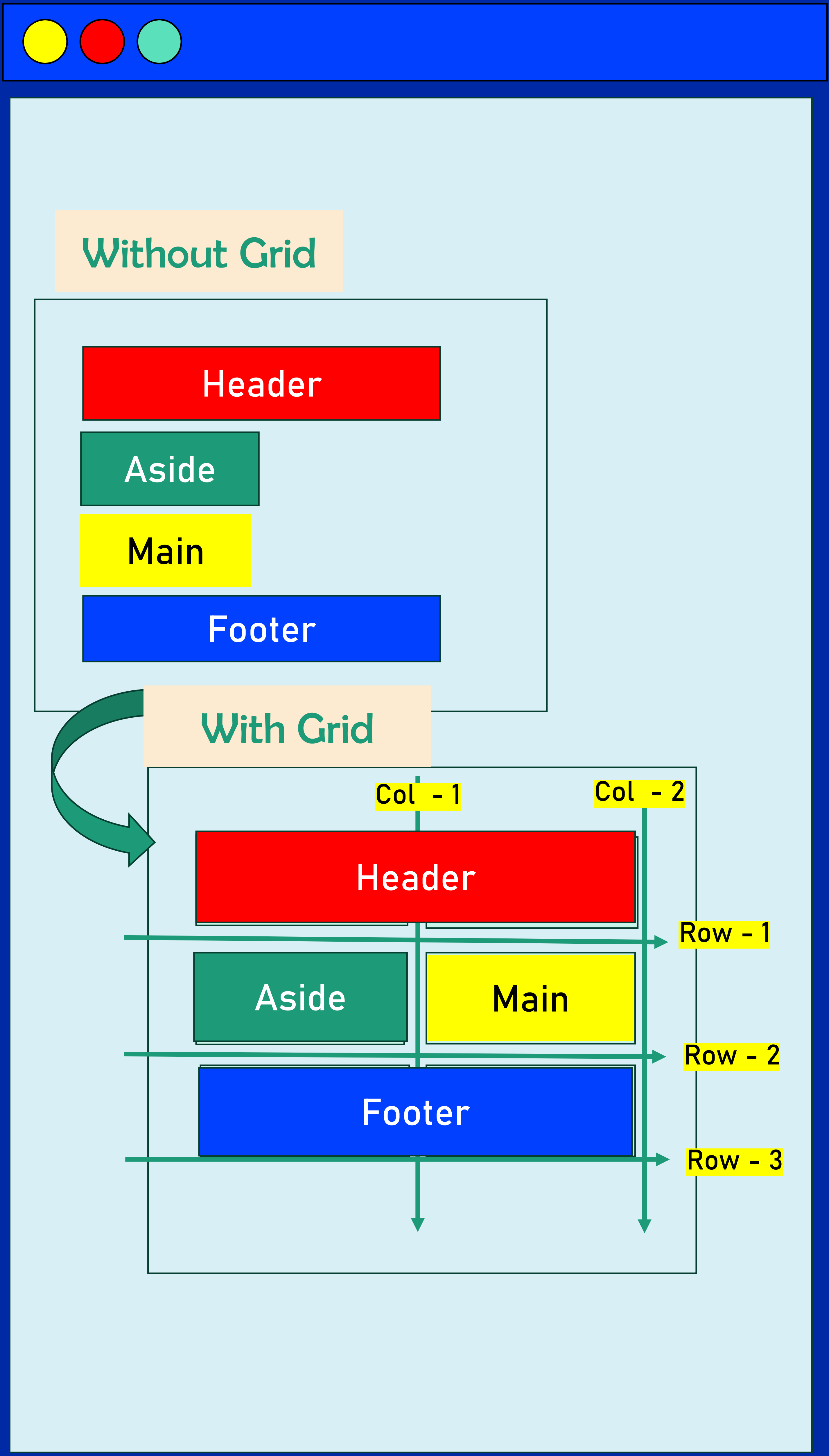


```
.grid-container{
    display: grid;
    grid-template-areas:
    "header header"
    "aside main"
    "footer footer";
}
```

It tells us how each rows and columns should display

```
.grid-container{
      display: grid;
      grid-template-areas:
      “header header”
      “aside main”
      “footer footer”;
}
```




Without Grid
Header
Aside
Main
Footer
With Grid
Col - 1
Col - 2
Header
Row - 1
Aside
Main
Row - 2
Footer
Row - 3

Must set! & make sure container is bigger than items

CSS – Grid Properties

display: grid or inline-grid;
gap: row-gap colum-gap;
grid-area: itemname | grid-row-start / grid-column-start / grid-row-end / grid-column-end;
grid-column: grid-column-start/ grid-column-end;
grid-row: grid-row-start / grid-row-end;
grid-template: grid-template-rows / grid-template-columns;
grid-template-areas: itemnames;

Grid Properties – 2

Vertical

align-content: center|stretch(default)|start|end|space-around|space-between|space-evenly;

align-items: center|start|end|stretch|baseline|normal(default);

align-self: center|start|end|stretch|baseline|normal(default);

Horizontal

justify-content: center|stretch(default)|start|end|space-around|space-between|space-evenly;

justify-items: center|start|end|stretch|baseline|normal(default);

justify-self: center|start|end|stretch|baseline|normal(default);

Both Vertical & Horizontal

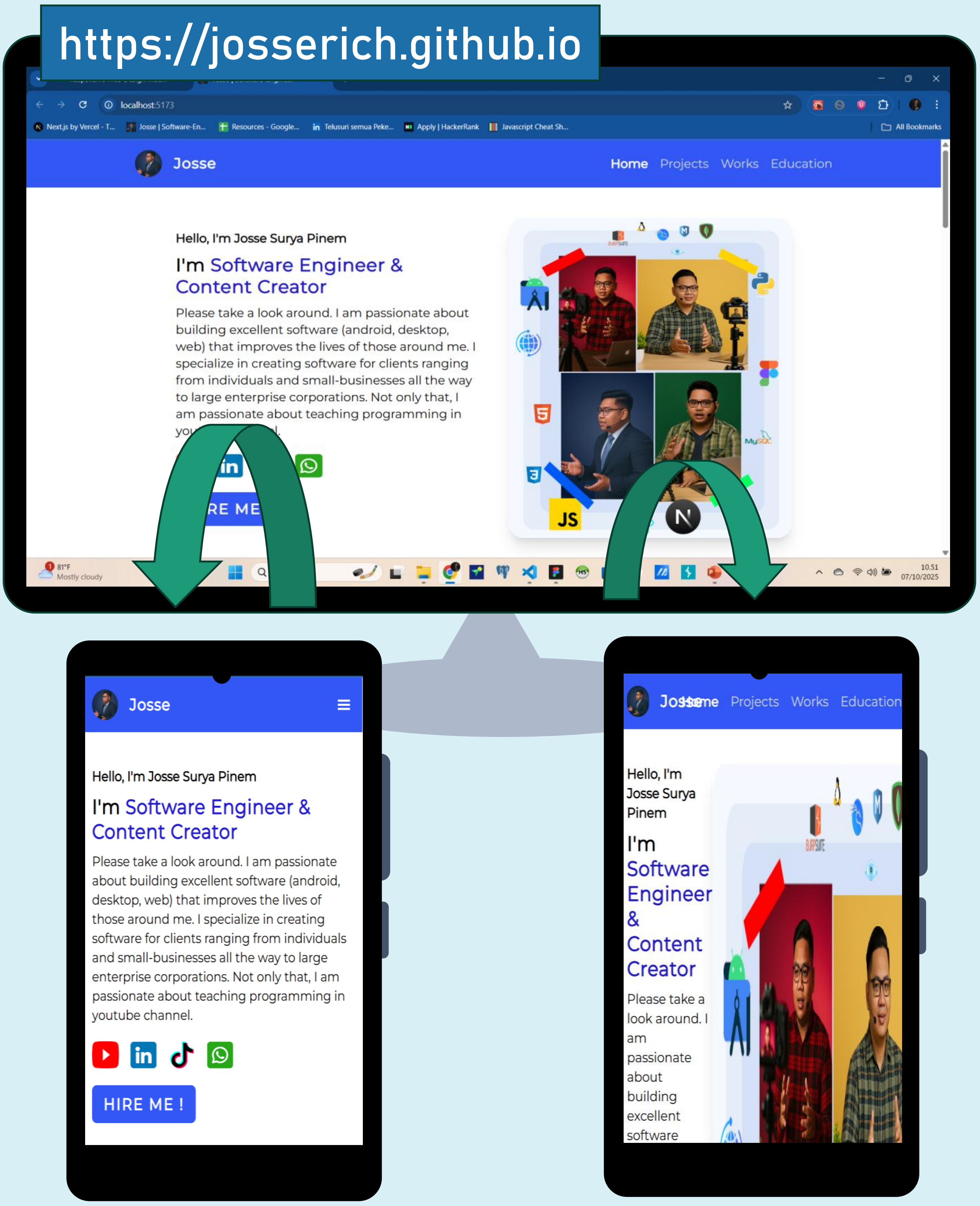
place-content: algin-content justify-content

place-self: algin-self justify-self

Implement this similar with flex

EXCERSICE GRID

https://josserich.github.io



Hello, I'm Josse Surya Pinem

I'm Software Engineer & Content Creator

Please take a look around. I am passionate about building excellent software (android, desktop, web) that improves the lives of those around me. I specialize in creating software for clients ranging from individuals and small-businesses all the way to large enterprise corporations. Not only that, I am passionate about teaching programming in youtube channel.

Responsive

Not Responsive

CSS – Responsive Web Design

Responsive Web Design makes **your web page look good on all devices**
Web pages can be viewed using many different devices: desktops, tablets, phone
Web pages should not leave out information to fit smaller devices, but rather **adapt its content to fit any device**

@mediaquery

```
/* Extra small devices (phones, 600px and down) */
@media only screen and (max-width: 600px) {...}

/* Small devices (portrait tablets and large phones, 600px and up) */
@media only screen and (min-width: 600px) {...}

/* Medium devices (landscape tablets, 768px and up) */
@media only screen and (min-width: 768px) {...}

/* Large devices (laptops/desktops, 992px and up) */
@media only screen and (min-width: 992px) {...}

/* Extra large devices (large laptops and desktops, 1200px and up) */
@media only screen and (min-width: 1200px) {...}
```

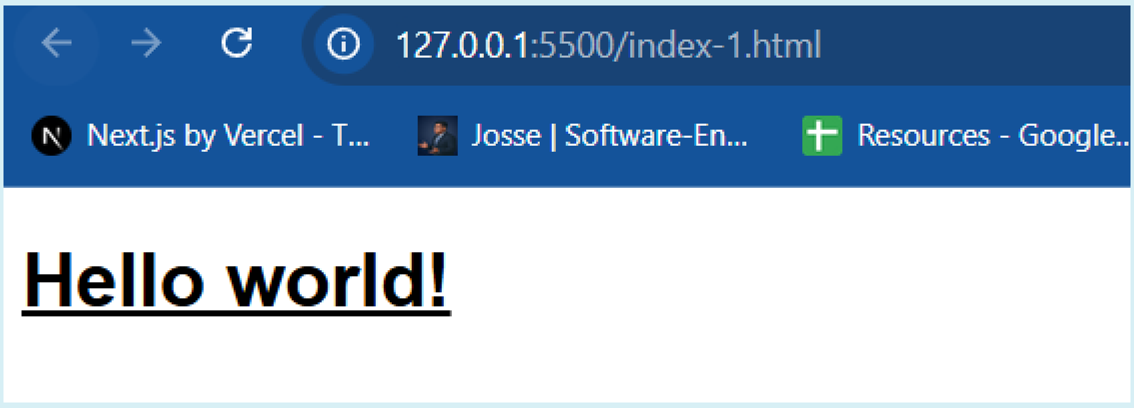
CSS Framework – Tailwind

A utility-first CSS framework packed with classes like *flex*, *pt-4*, *text-center and rotate-90* that can be composed to build any design, directly in your markup.
CSS Utility = a small class that has one specific function.

Rapidly build modern websites without ever leaving your HTML.

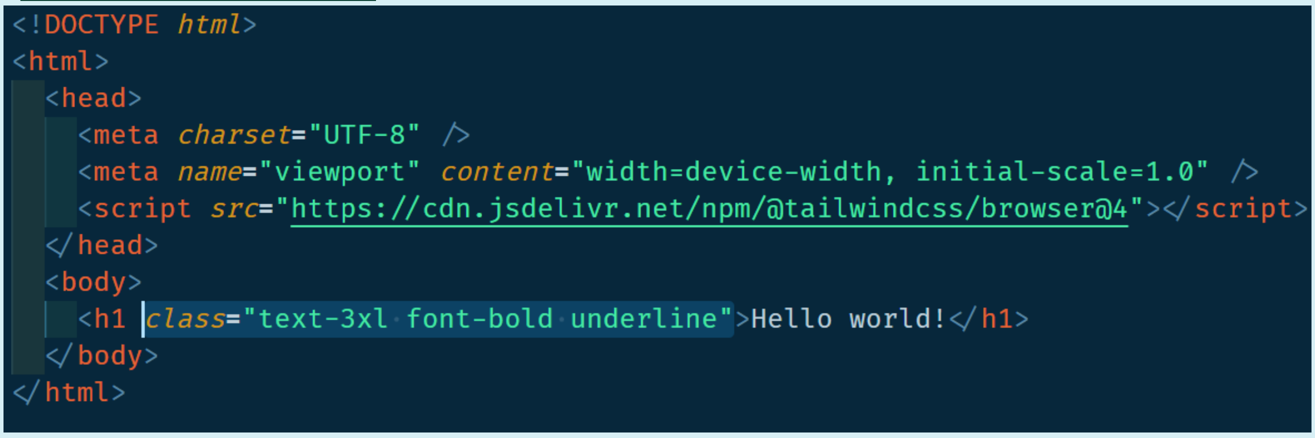
Adam Wathan, 1 November 2017

Tailwind	CSS
text-3xl	font-size: 30px;
font-bold	font-style: bold;
underline	font-decoration: underline;



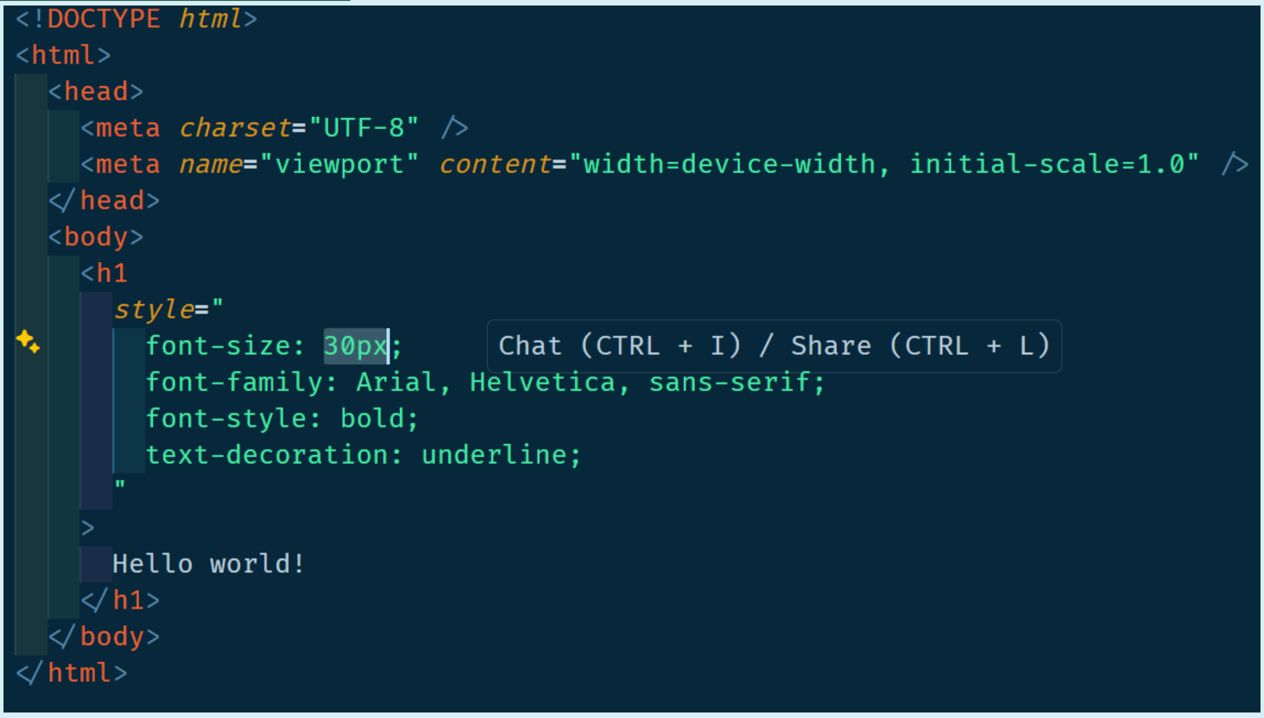
Hello world!

With Tailwind



```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
    <script src="https://cdn.jsdelivr.net/npm/@tailwindcss/browser@4"></script>
  </head>
  <body>
    <h1 class="text-3xl font-bold underline">Hello world!</h1>
  </body>
</html>
```

Without Tailwind



```html
<!DOCTYPE html>
<html>
  <head>
    <meta charset="UTF-8" />
    <meta name="viewport" content="width=device-width, initial-scale=1.0" />
  </head>
  <body>
    <h1
      style="
        font-size: 30px;        Chat (CTRL + I) / Share (CTRL + L)
        font-family: Arial, Helvetica, sans-serif;
        font-style: bold;
        text-decoration: underline;
      "
    >
      Hello world!
    </h1>
  </body>
</html>
```

Sources

https://www.w3schools.com/css
https://tailwind.css
https://www.w3.org/TR/css-cascade-3/
https://www.smashingmagazine.com/2010/04/css-specificity-and-inheritance/
https://specificity.keegan.st/
https://www.flaticon.com/

THANK YOU !

Don't forget to visit

https://josserich.github.io