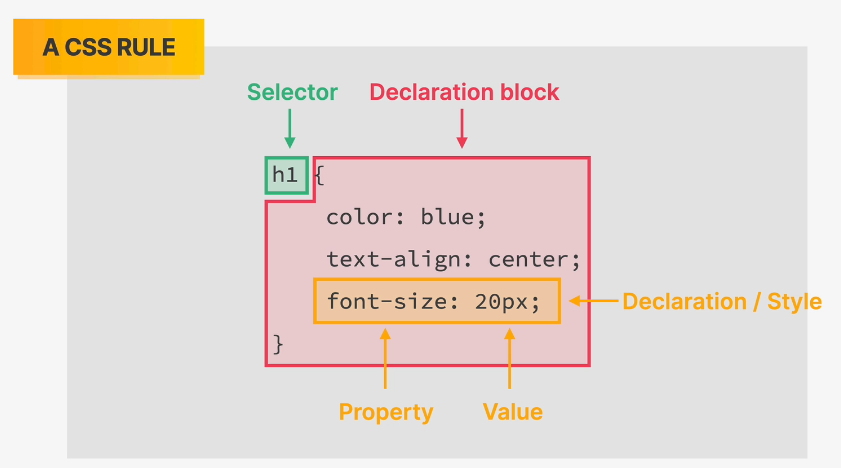
**WHAT IS CSS?**

* Cascading Style Sheets
* CSS describes the visual style and presentation of the content written in HTML
* CSS consists of countless properties that developers use to format the content: properties about font, text, spacing, layout, etc.

**How we Select and Style Elements?**



**INLINE STYLE CSS**

 <h1 style="color: blue">The Code Magazine</h1>

**INTERNAL STYLE CSS**

<head>

    <meta charset="UTF-8" />

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

    <title>Document</title>

    <style>

      h1 {

        color: blue;

      }

    </style>

  </head>

**EXTERNAL STYLE CSS**

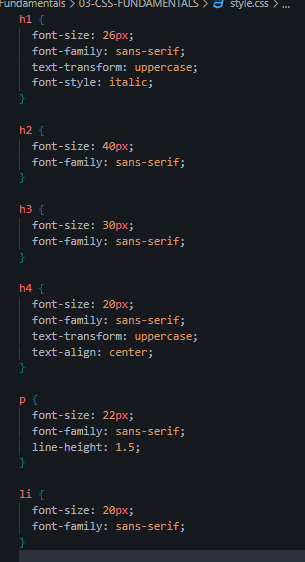
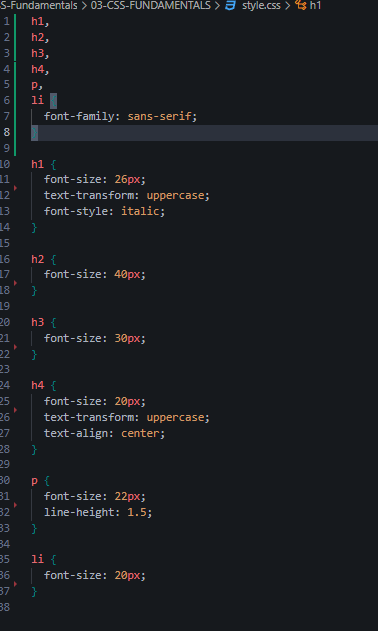
Creating an external CSS file and calling the stylesheet into the html:  
 <link rel="stylesheet" href="./style.css" />

**STYLING TEXT**

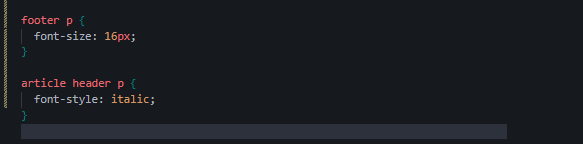
1. Font-size
2. Font-family
3. Text-transform: capitalization, uppercase, lowercase etc.
4. Font-style: bold, italic etc.
5. Line-height: space between two lines
6. Text-align: alignment of text.

**COMBINING SELECTORS**

Instead of this: Making this:

**DESCENDANT SELECTOR**

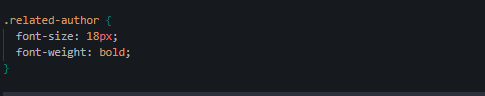


First one means the p inside of footer

Second one means the p inside the header and the header inside article.

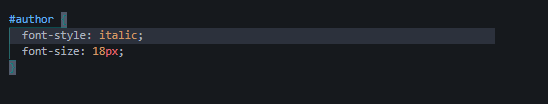
This helps us differentiate between the tags.

**CLASS SELECTORS**



We can set the class attribute in the html tag and then call it with a dot in the css file.

**ID SELECTORS**



We can set the id attribute in the html tag and then call it with a hashtag in the css file.

*Difference between class and id is we cannot repeat id names but class is allowed  
In real-world scenarios we use classes not id because we are preparing for the future in case of new changes.*

**STYLING LISTS**

1. List-style: none means no bullet points can change what the bullet points look like.

**THE RGB MODEL**

* Every color can be represented by a combination of red, green and blue.
* Each of the 3 base colors can take a value between 0 and 355 which leads to 16.8 million different colors.

**DEFINING COLORS IN CSS**

1. **RGB/RGBA NOTATION**

* regular rgb model
* rgb(0,255,255)
* rgb with transparency (“alpha”)
* rgba(0,255,255,0.3)

1. **HEXADECIMAL NOTATION**

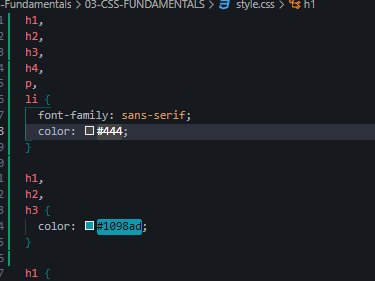
* Instead of using a scale from 0 to 255 we go from 0 to ff (255 in hexadecimal numbers}
* #00ffff

*In practice we mostly use hexadecimal colors and rgba when we need transparency.*

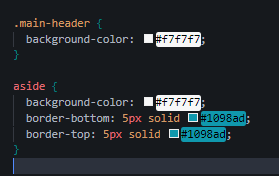
**SHADES OF GREY**

* When colors in all 3 channels are the same, we get a grey color
* There are 256 pure grays to choose from.
* 

**WORKING WITH COLORS**

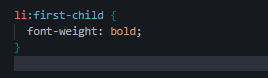


Setting the color of the text or anything.   
color for h1, h2, h3 remains the blue one because it gets executed after the list above.

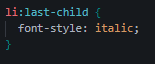


Background color sets the color of the background of the selected tag  
border: width typeOfBorder{solid, dashed etc) color;

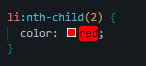
**PSEUDO-CLASSES**



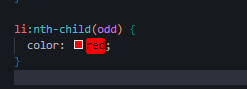
Selects all of the li tags and then the first child of all of those li elements.’



Selects all of the li tags and then the last child of all of those li elements.’



Selects all of the li tags and then the nth (index number starts from 0) child of all of those li elements.’

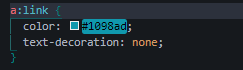


All of the odd numbered childs.

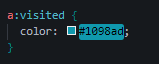


It will not select the first p of the article tag because p isn’t the first child of the article tag it is the header.

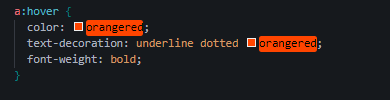
**STYLING HYPERLINKS**

****

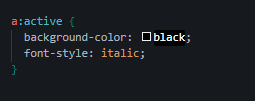
Will only target the anchor tags which have an href attribute.  
text-decpoooration:none removes the hyperlink underline



Visited means the links that have been clicked on before.



When you hover over the hyperlink you will see these stylings.

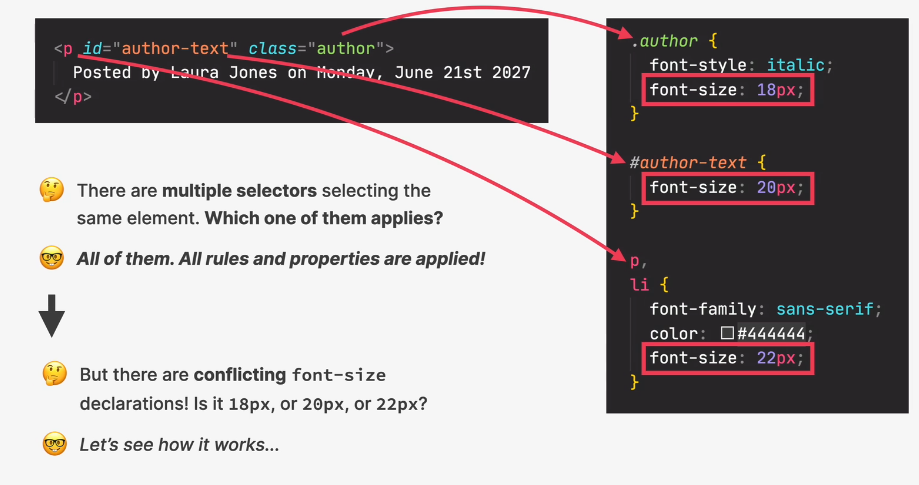


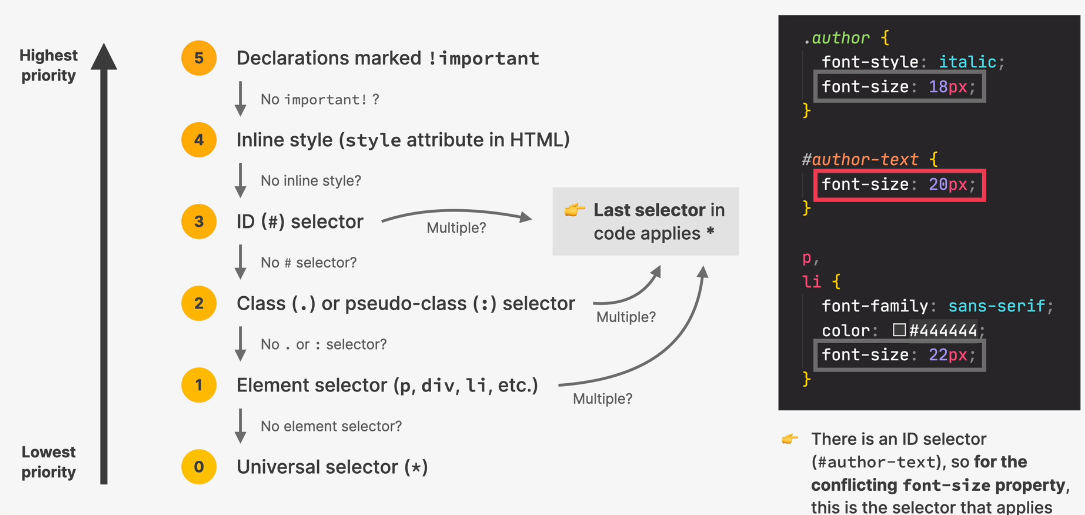
These stylings can be seen when the hypperlink is selected and moved here and there.

*The above order is to be kept in mind when creating the stylings for hyperlinks   
link-> visited -> hover -> active*

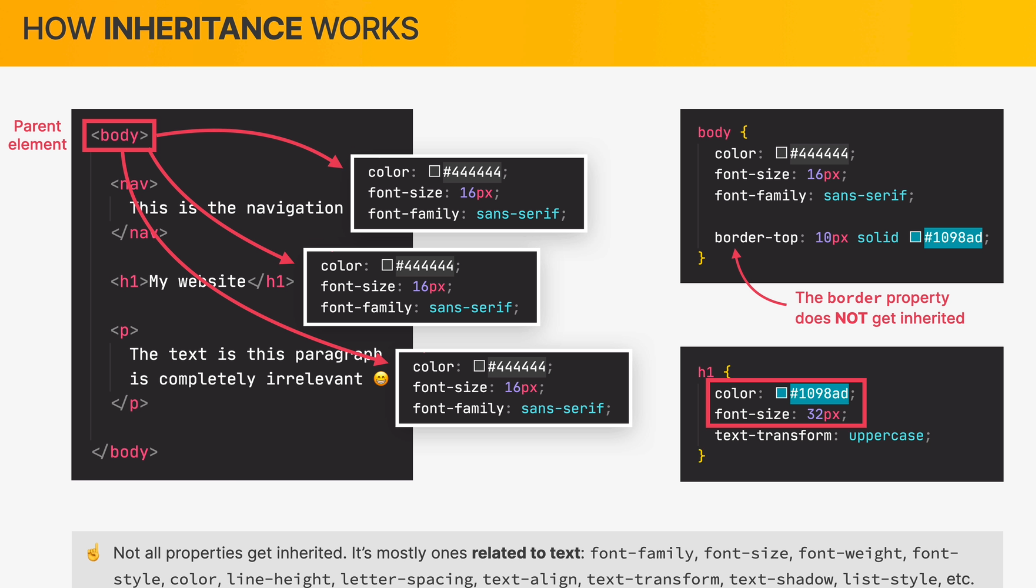
**CSS THEORY #1: CONFLICTS BETWEEN SELECTORS**

Conflicting selectors and declarations

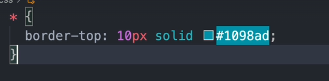




**CSS THEORY #2: INHERITANCE AND THE UNIVERSAL SELECTOR**

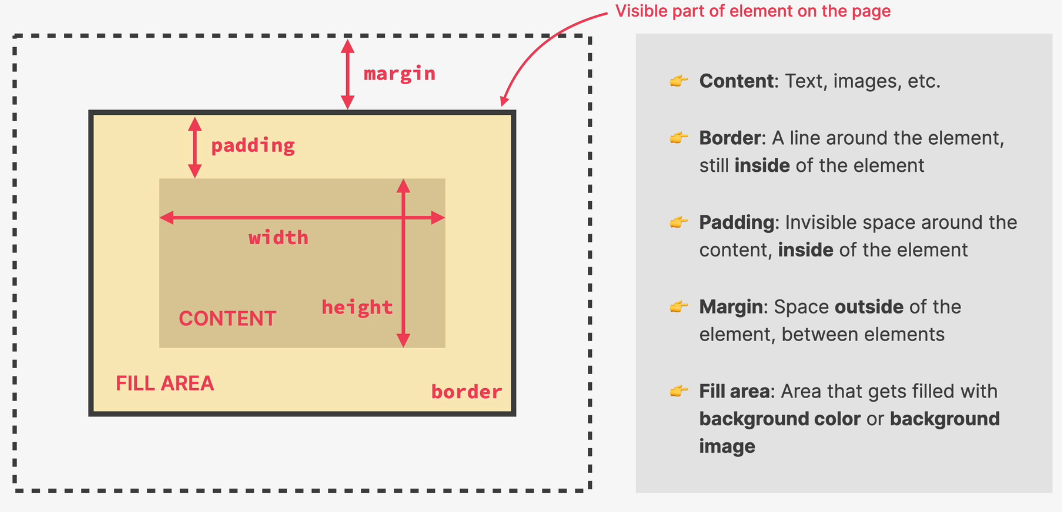


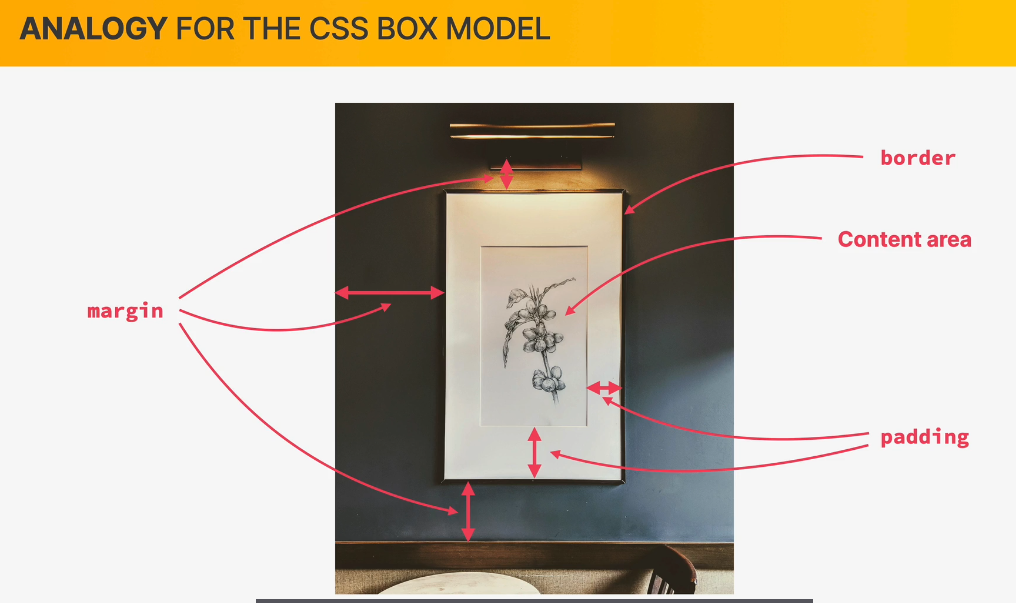
UNIVERSAL SELECTOR (\*)

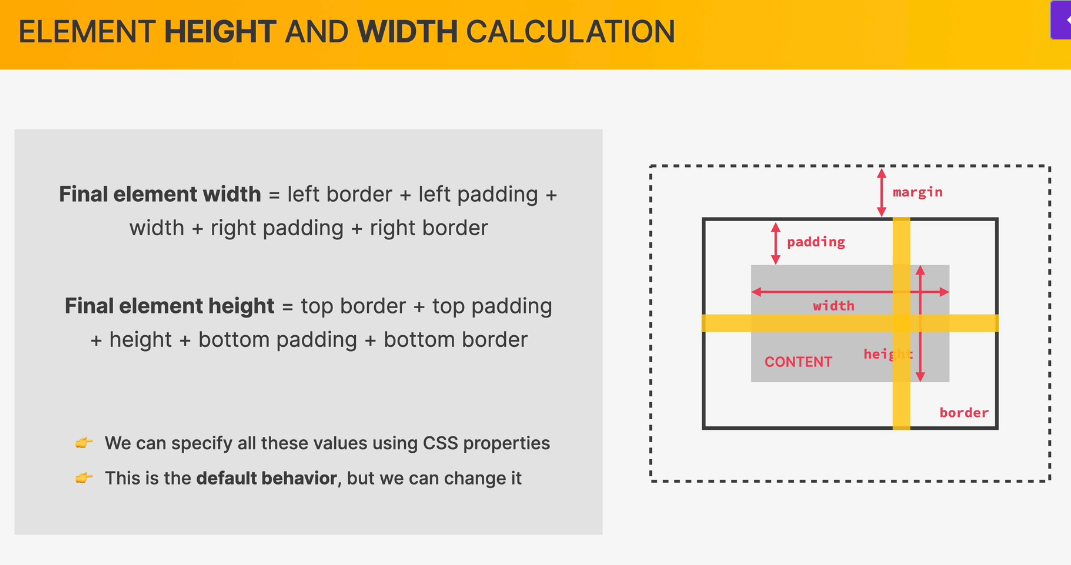


Something you want that should apply to all of the attributes.

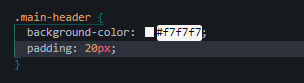
**CSS THEORY #3: THE CSS BOX MODEL**

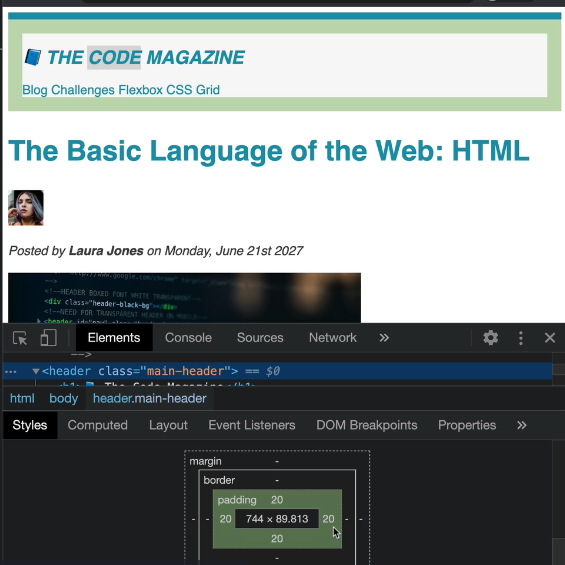


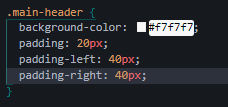




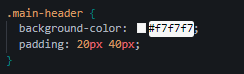
**USING MARGINS AND PADDINGS**





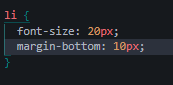
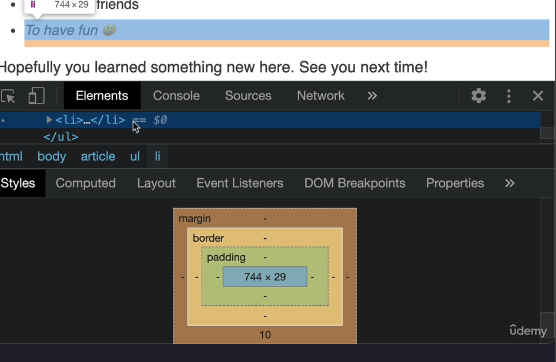




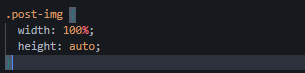




MARGINS

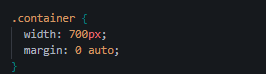
**

**ADDING DIMENSIONS**



Width: 100% means it occupies all of the width of the parent container this helps create responsive websites which respond to screen width

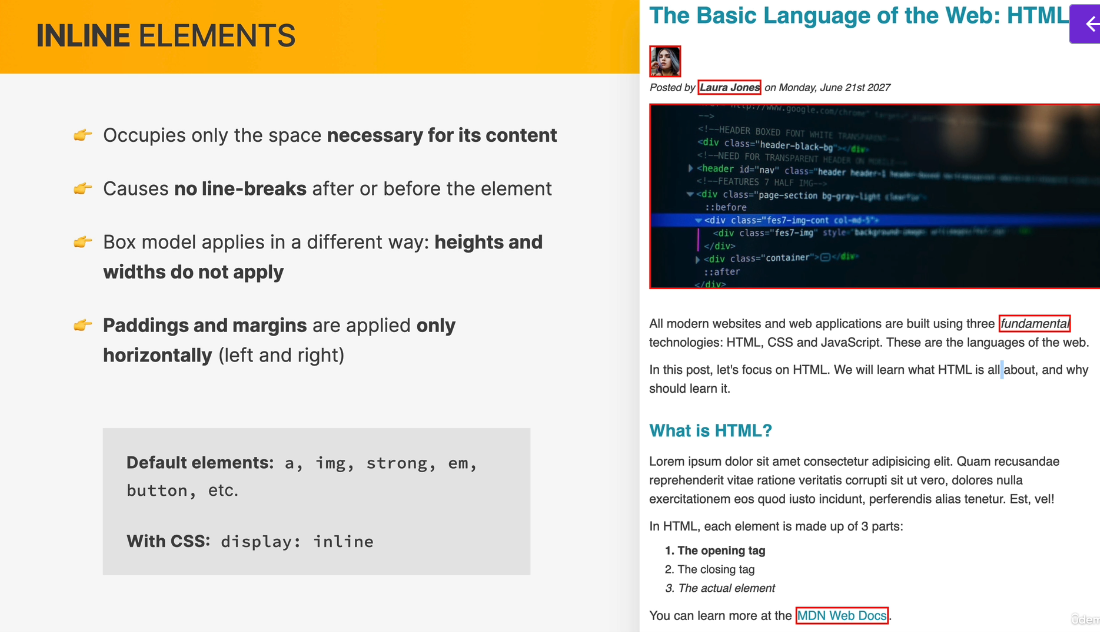
**CENTERING OUR PAGE**

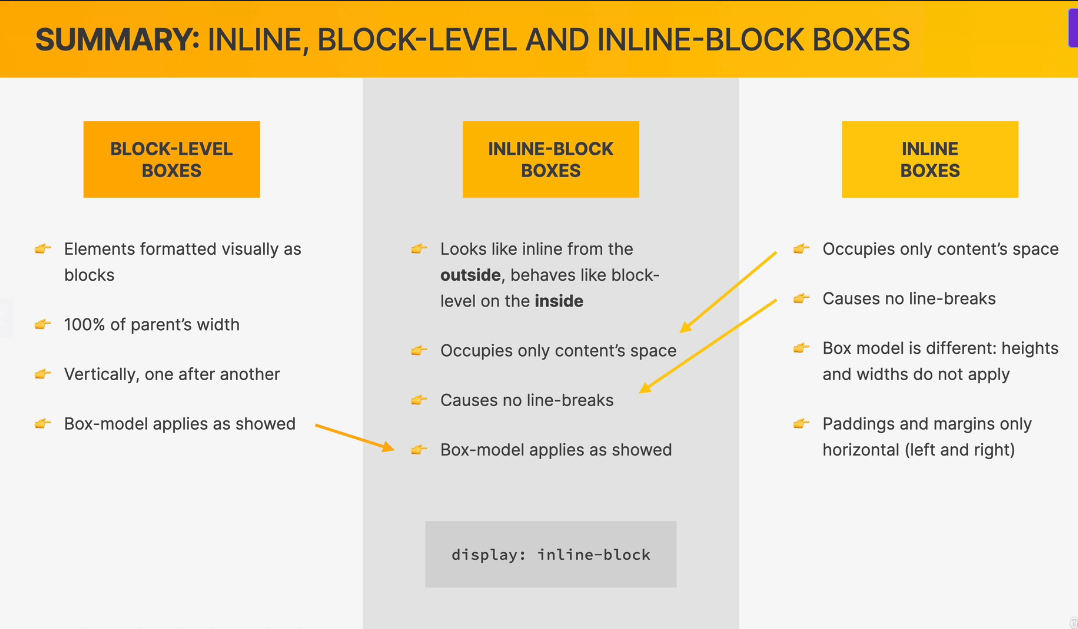


This helps us center our page

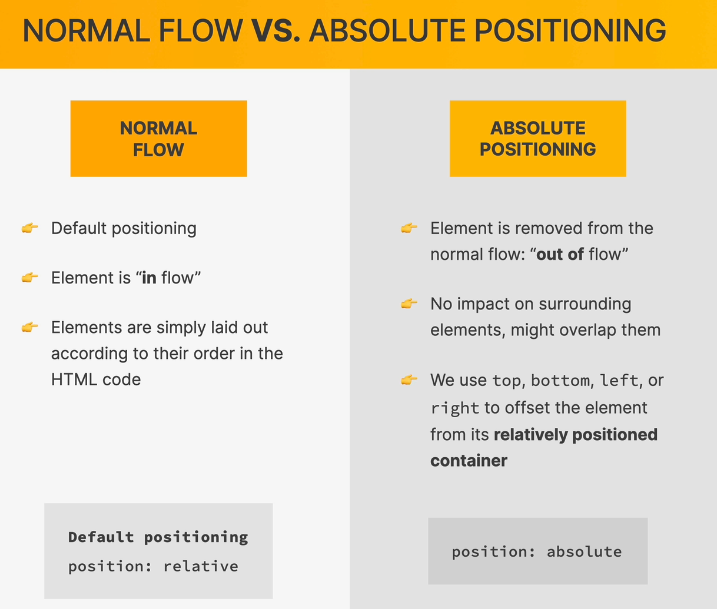
**CSS THEORY #4: TYPES OF BOXES**

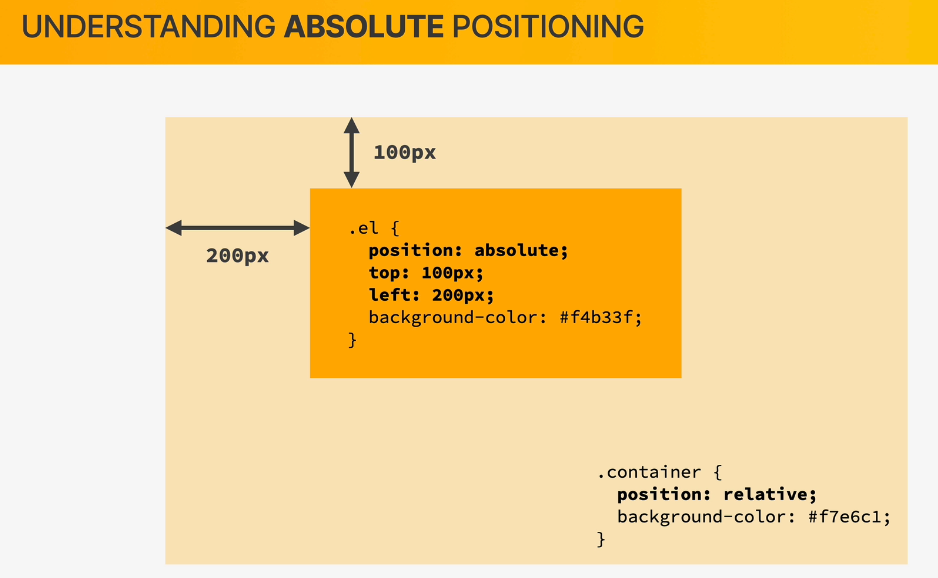




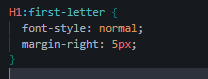


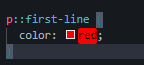
**CSS THEORY #5: ABSOLUTE POSITIONING**

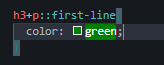




**PSEUDO-ELEMENTS**





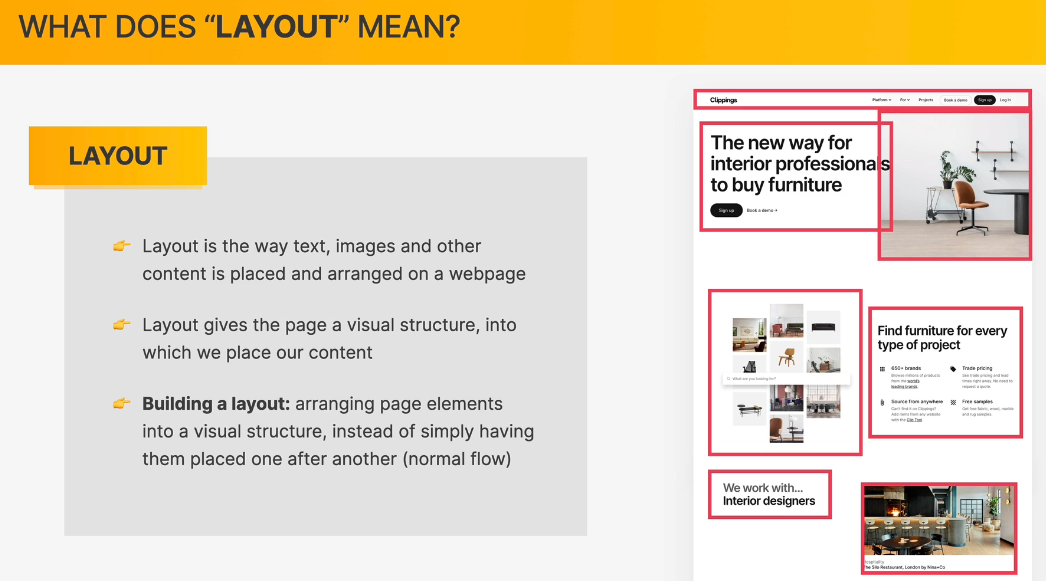
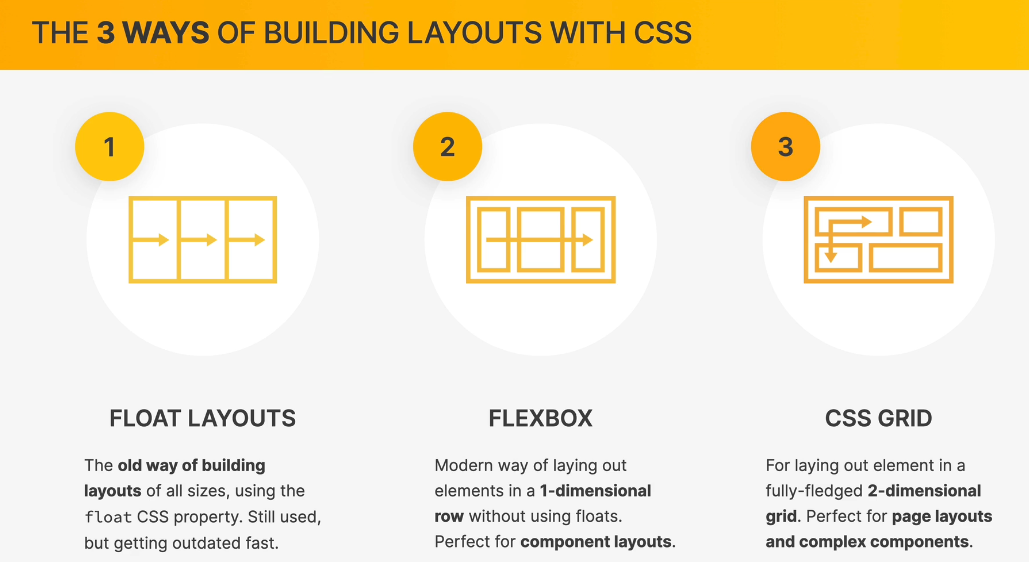
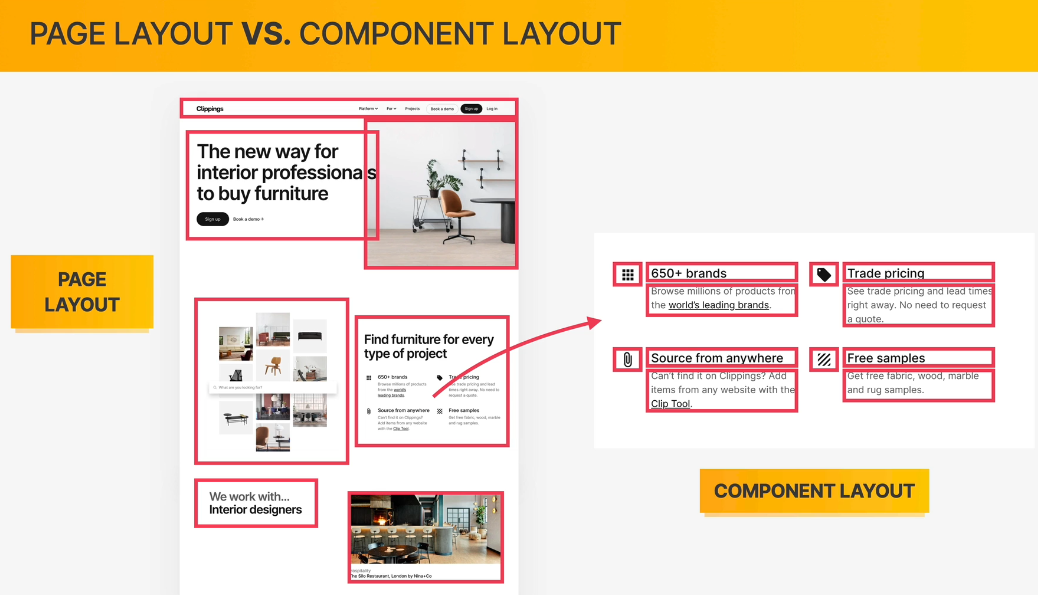


+ indicates the adjacent element to h3 the one that is just next to h3.

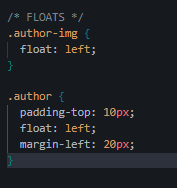
 

The after created a new element which is for cosmetic purposes and we don’t really want to create an element for it so we used after pseudo element to create one in css after the h2.

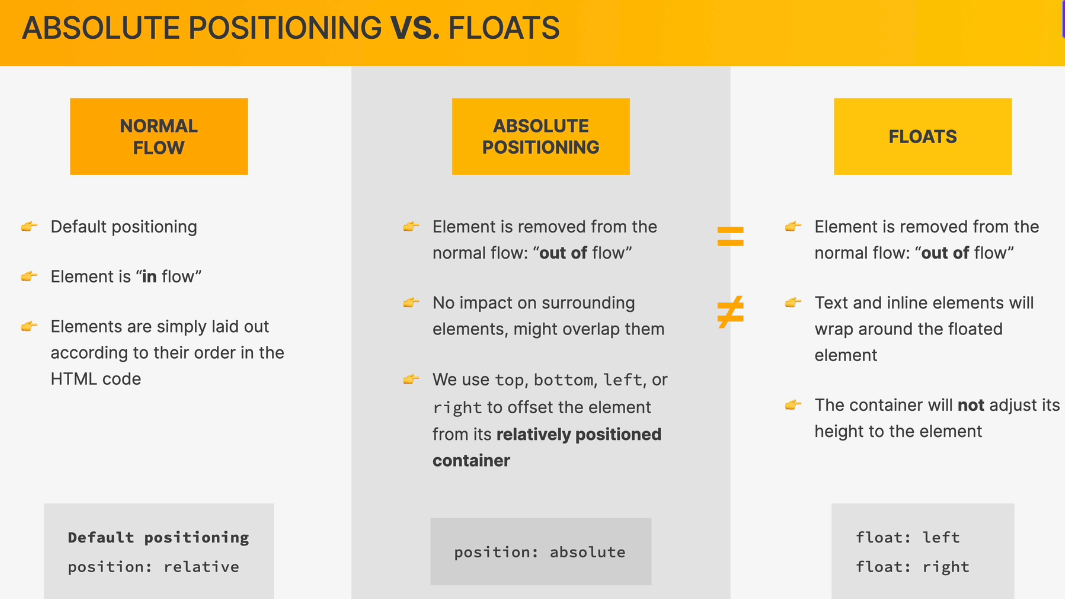
**3 WAYS OF BUILDING LAYOUTS**

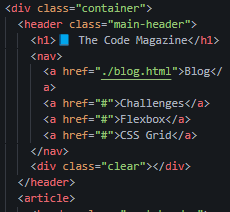
**USING FLOATS**







**CLEARING FLOATS**

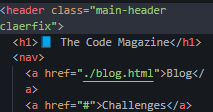


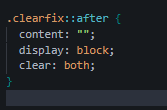
Create an empty div for clearing the float



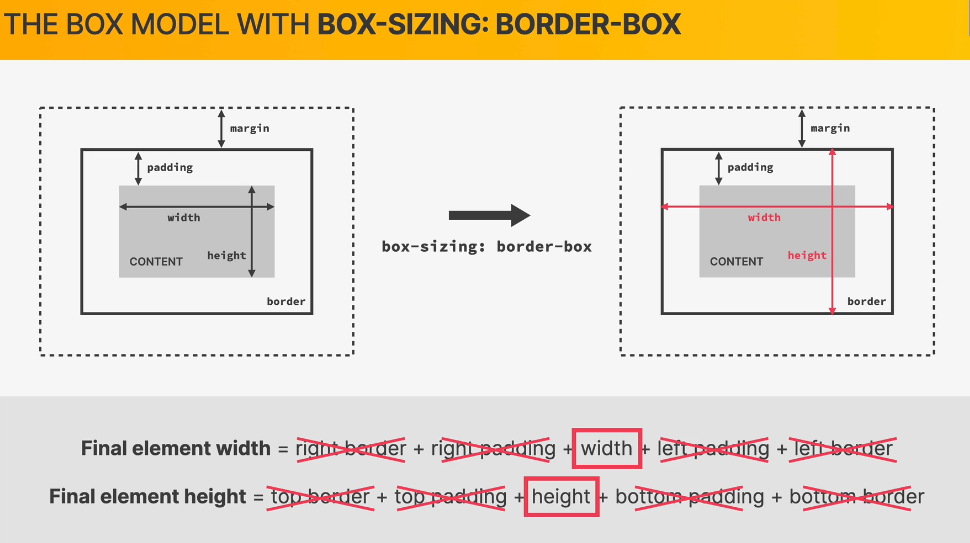
But this is just bad practice.

So instead of using this we use the clearfix hack.





**Box-sizing: border-box**

****

**INTRODUCTION TO FLEXBOX**

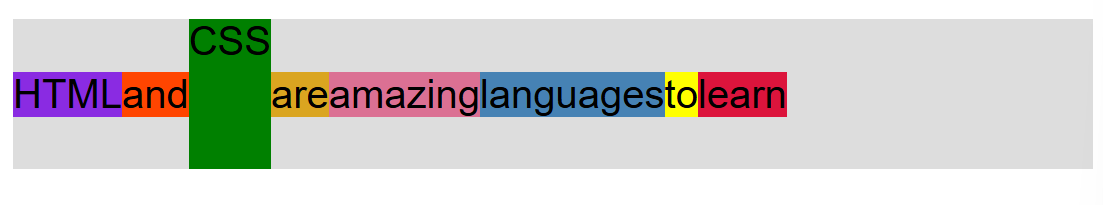
Originally:



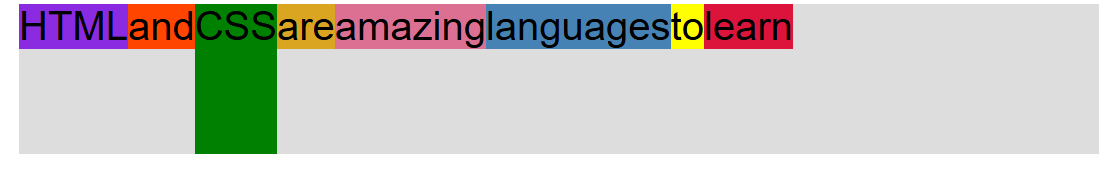
After applying display: flex to the container element.



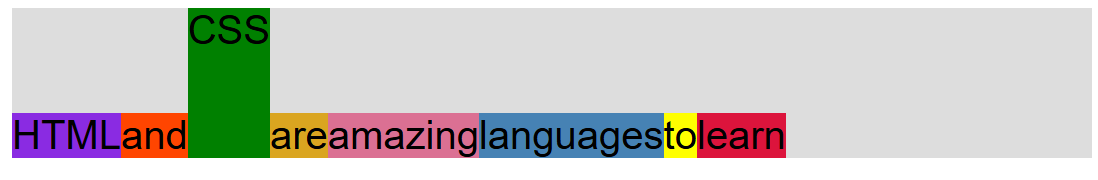
All of them are the size of green element because its height has been set to 150 px

After applying align-items: center  


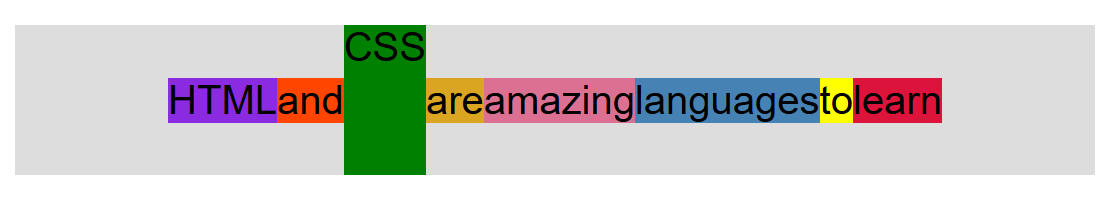
After using align-items:flex-start



After using align-items:flex-end



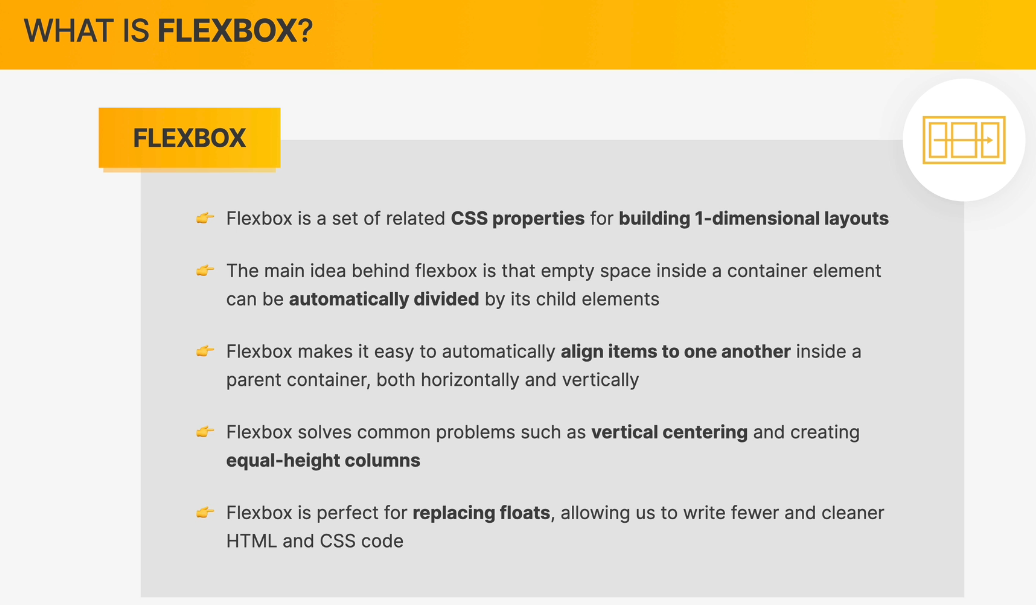
After using justify-content: center {the elements are in the center of the container

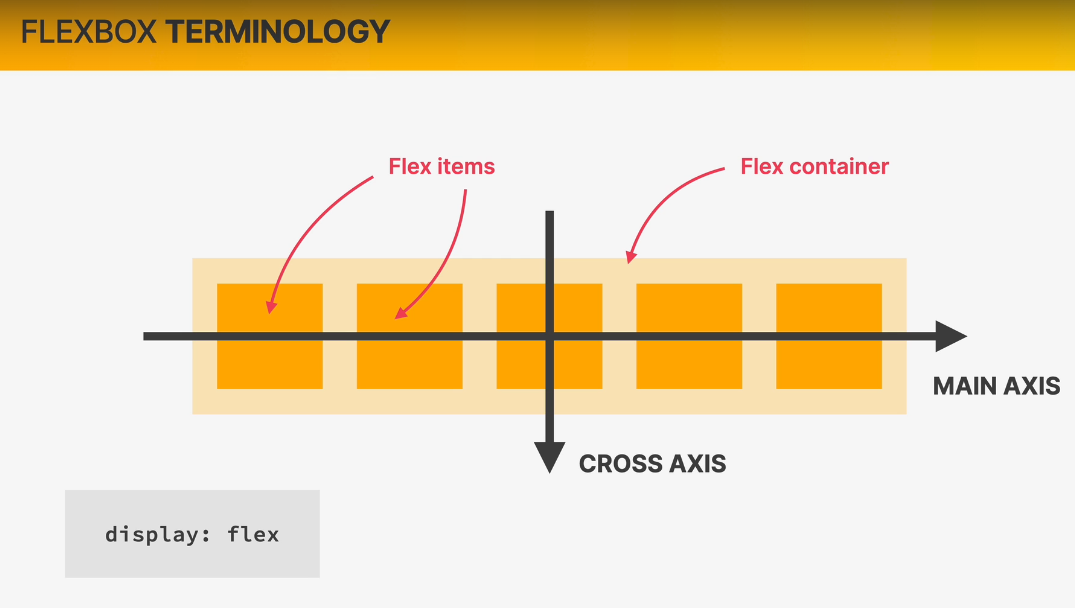


After using justify-content: space-between



**FLEXBOX OVERVIEW**

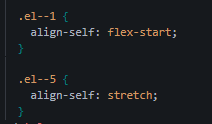
****

****

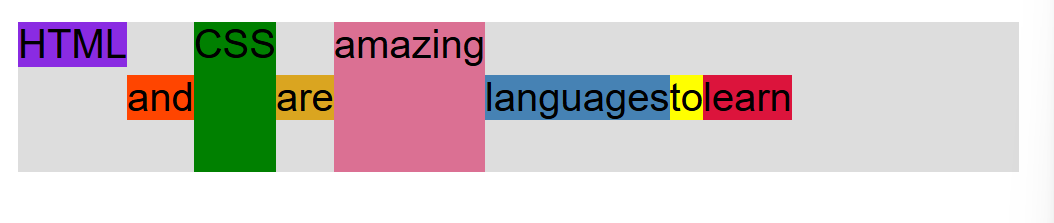
**We need to set the display:flex for the container to start with flexbox**

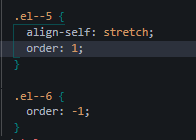


**SPACING AND ALIGNING FLEX ITEMS**



If we want one of the items to behave differently we use align-self



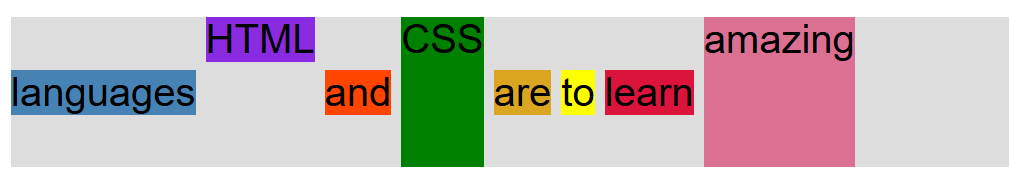


Initially the order of all of the elements is 0.

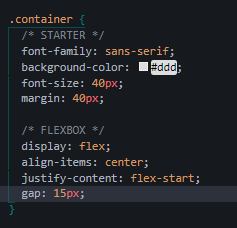


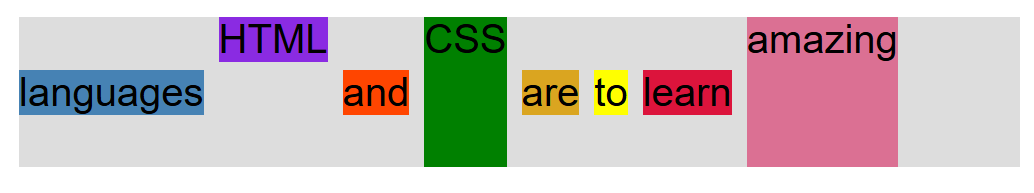
Adding spaces





Instead of doing the above we can use gap property of the container





**THE FLEX PROPERTY**

Flex-basis: is used to set the width of the elements. We don’t really use width we use this.

