


Box model of css

Content, padding, border, margin

(in that order) 

This is what the box model looks like.

- Each layer represents a different part of the model

Each layer can be stretched and sized either symmetrically or asymmetrically

PADDING

- Padding represents the space between the content and the border

Padding: 100px 40px 5px 0;

i.e padding:(top, right,bottom, left)

Note: they are separated using space

BORDER

- The border is the divider between the padding and margin
- It can be styled using a css property called **border**
- (border: 10px double red) or (border: 10px dotted red) or(border: 10px dashed red) or(border: 10px solid.red)
-

MARGIN

The margin is the space between the border and all other content.

Margin: 80px 40px;(this styles both the top/bottom and the right/left respectively)

Ps:using absolute value is better than other types pf values because they are more versatile

FLEXBOX

- Flexbox stands for “flexible box”
- It is a display type that comes with a range of properties allowing you to arrange items easily
- It is an alternative to using displays, floats and other layout properties

FLEXBOX COMPONENTS

- A flexbox element is split into two main parts: the container, and the items.
- The container is the parent element in which the display type is active. This is usually in the form of a div
- Flex items are child elements of the container, and make up the contents of the box

Ps: *Setting the display property to flex will load the flex box properties by default*

FLEX DIRECTION AND WRAP

FLEX DIRECTION

- Here we can specify the direction we want our text to go
- We can reverse the direction by using a special suffix (**reverse**)
- The property that we can use is flex direction(the default flex-direction is row)

FLEX WRAP

TEXT ALIGNMENT

- Justify-Content is for horizontal text alignment
- The most common types are flex-start, flex-center and flex-end(same goes for align-items)

there is also space around and space between

SPACE BETWEEN

This creates equal space between the flex items and the items are also set to the same size

SPACE AROUND

This creates equal space around all the flex items

BASELINE

This aligns all the text in the flex box to the same baseline irrespective of their different heights

If you want all flex items to be at the center, you just need to set the values for both the “justify-content” and the “align-content” as “center”

i.e

```
justify-content="center";
```

```
align-content="center"
```

FLEX ITEM ORDER

Style="order:(any number from 1 to the number of items specified)"

SHRINK, GROW AND BASIS

BASIS

Flex-basis specifies the minimum width of the flex item (it doesn't have to be of a particular width or value)

```
<div class="container-item" style="order:1;
      flex-grow;;
      flex-shrink;;
      flex-basis;;">D</div>
```

GROW

Flex-grow means all the items have the same space amount

If you set all the items to say 1, then set another one to 2, the one set to 2 will take up 2 times the extra space taken by the other flex items

The flex-grow items still take up the same proportions, even when the page size increases

Note: the default value for **flex-grow** is **0**

SHRINK

It is similar to flex grow except it defines the rate of shrinkage of each container item relative to the other items.

Also it has a default value of **0**

set it to 0 if you don't want to shrink the item.

Setting it to 3, while the other flex items are 1 means that the particular item will shrink 3 times as much as the other flex items

GRID VS FLEXBOX

SIMILARITIES

- Similar to flexbox, Grid is a display type that can be used to activate certain layout features on a container element.
- They are alternatives to other layout features available in css.

DIFFERENCES

- However, these two display types work in very different ways.
- The Grid system is more manual, providing you with more tools to layout your container in a specific way.
- Grid is also very focused on providing tools for both dimensions [width+height]. Whereas Flexbox is focused on width alone.

SETTING UP THE GRID

```
<div class="grid-container">
  <div class="grid-
item" style=" background: rgb(255, 150, 150)">1</div>
  <div class="grid-
item" style=" background: rgb(170, 255, 140)">2</div>
  <div class="grid-
item" style=" background: rgb(222, 132, 255)">3</div>
  <div class="grid-
item" style=" background: rgb(255, 207, 119)">4</div>
  <div class="grid-
item" style=" background: rgb(148, 210, 255)">5</div>
</div>
```

HTML CODE

```
.grid-container{
  background-color: rgb(250, 250, 250);
  margin: 10px;
  height: 300px;
  display: grid;
}

.grid-item{
  color: black;
  font-size: 25px;
  text-align: center;
```

```
width: 70px;  
line-height: 40px;  
margin: 10px;  
}
```

CORRESPONDING CSS CODE

TEMPLATE TABLE AND ROWS

Their properties are denoted by **GRID-TEMPLATE-COLUMNS** and **GRID-TEMPLATE ROWS**

Setting them to auto is the best as it divides the space evenly

JUSTIFY AND ALIGN GRID

The justify property justifies content horizontally

Unlike in flexbox, both the justify-content and the align-content properties don't have prefix for their values. They look as shown below:

```
justify-content: center;
```

note: flexbox uses **space around** and **space between**, while grid uses **space around** and **space evenly**. This is for **justify content**

align content is for vertical alignment