CSS

Satvik Gupta

January 22, 2023

Basic CSS Rule

```
selector {
  property_a: value_a;
  property_b: value_b;
}
```

Selectors can be of many types, the main ones are Element, ID, and Class.

Element:

Selects a particular type of HTML Element.

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

ID:

Selects HTML Elements by their ID. Each HTML element can have an ID, but no ID should be used for more than one HTML Element. IDs are unique.

We add # in front of the selector in CSS to specify that we are using an ID selector

```
#my-element-id {
  color: red;
}
```

Class:

Selects HTML element by their class. Each HTML element can have a class. Classes need not be unique, multiple elements can belong to the same class.

We add `.` in front of the selector in CSS to specify that we are using a class selector.

```
.text-box {
  color: green;
}
```

Specificity

If a particular element has two competing properties that can be applied to it, *generally* the more specific one will be applied.

For eg - a color mentioned in ID will override a color mentioned in class.

Font

Font Family

```
font-family: Helvetica, Times, sans-serif;
```

The browser will check in order of specificiation.

Font Size

em sizes are relative to direct parent. *rem* sizes are relative to the root (HTML) parent, i.e, the default.

For e.g, let the HTML code be this.

```
<p>
  This is normal text. This text has this word--
  <span class="sizer">MONOLITH</span>
  --in a span.
</p>
```

The following is the CSS:

```
p {
  font-size: 12px;
}

.sizer {
  font-size: 2em;
}
```

This will make the word **MONOLITH**, which has the class *sizer*, 2 times the size of *p*, i.e. 24 px, because we have made it *2em*. If we change the size of *p* to 16px, **MONOLITH** will have size 32px.

If we use REM, however, like in the following CSS:

```
p {
  font-size: 12px;
}

.sizer {
  font-size: 2rem;
}
```

The **MONOLITH** font size will be twice of the default font size. Changing the font size of *p* won't change the font size of **MONOLITH**

Other Attributes

line-height – number

text-align – right, left, justify, etc.

text-decoration – line-through,underline, overline,etc.

Box Rule for HTML

Flex Box

Flex-boxes let us align,size and position items in rows and columns easily.

HTML:

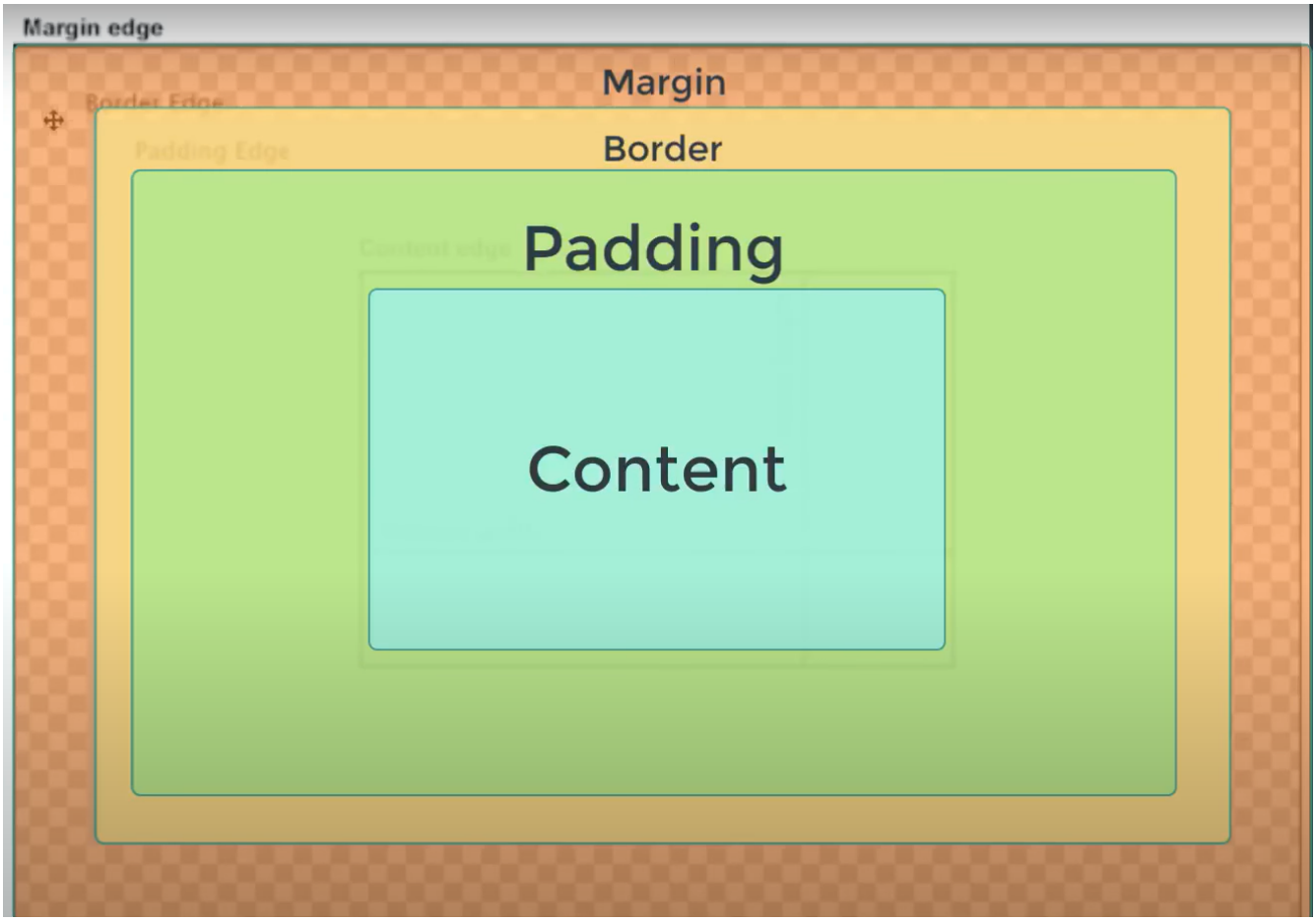

Figure 1: Box Rule for HTML, Margin->Border->Padding->Content

```
<div class="container">
  <div class="flex-item" id="flex-item-1">1</div>
  <div class="flex-item" id="flex-item-2">2</div>
  <div class="flex-item" id="flex-item-3">3</div>
</div>
```

CSS:

```
.container {
  display: flex;
}

.flex-item {
  min-width: 100px;
  min-height: 100px;
  color: white;
  font-size: 20px;
  text-align: center;
}

#flex-item-1 {
  background-color: red;
}

#flex-item-2 {
  background-color: green;
}

#flex-item-3 {
  background-color: blue;
}
```

Output:

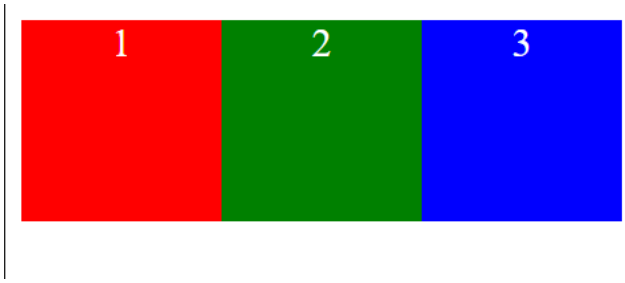

Figure 2: flex-box-basic-output

Options

flex-direction : *row,column,row-reverse,column-reverse*

flex-wrap: *nowrap,wrap,wrap-reverse*
By default flex-box will resize flex items to fit into the container width. If we wrap, they will be shifted to the next row/column if needed.

justify-content: *flex-start,flex-end,center,space-between,space-around,space-evenly.*

This aligns the content along the main axis.

align-items: *flex-start,flex-end,center,stretch,baseline.*

This aligns the content along the cross axis.

Options for the items in the flex box.

flex-grow: *integer*. Allows the flex item to grow, if space is available. Items grow according to the ratio of their *flex-grow* property.

For example, if there are 3 items in a flex-box, with a flex-grow of 1,2,1 respectively, and there is 100px of extra space available – the first item will get 25px extra space, the second one will get 50px, and the last one will get 25px.