

# CSS 3 CHEAT SHEET


## Reset

```
*{
  margin:0;
  padding:0;
}
```


## Fonts

Use `font-family` to change the styling of the font. Default is Times New Roman.

# Web Safe Fonts



Serif



Sans Serif

Examples	Font Families
font-family	Arial, Helvetica, sans-serif
<b>font-family</b>	<b>"Arial Black", Gadget, sans-serif</b>
font-family	"Bookman Old Style", serif
font-family	"Comic Sans MS", cursive, sans-serif
font-family	Courier, monospace
font-family	"Courier New", Courier, monospace
font-family	Garamond, serif
font-family	Georgia, serif
<b>font-family</b>	<b>Impact, Charcoal, sans-serif</b>
font-family	"Lucida Console", Monaco, monospace
font-family	"Lucida Sans Unicode", "Lucida Grande", sans-serif
font-family	"MS Sans Serif", Geneva, sans-serif
font-family	"MS Serif", "New York", sans-serif
font-family	"Palatino Linotype", "Book Antiqua", Palatino, serif
φοντ-φασμλψ	Συμβολ, σανς-σεριφ
font-family	Tahoma, Geneva, sans-serif
font-family	"Times New Roman", Times, serif
font-family	"Trebuchet MS", Helvetica, sans-serif
font-family	Verdana, Geneva, sans-serif

Use `font-weight:bold` to make it BOLD.

Use `font-size` to change the size.

Instead of doing

```
body{
  font-weight: normal;
  font-size: 16px;
  font-family: Arial, Helvetica, sans-serif;
}
```

we could actually use a shorthand notation. Use:

```
body{
  font: normal, 16px, Arial, helvetica, sans-serif;
}
```

Use **line-height** to increase/decrease the line height (space b/w two lines in the text). Usually, use em as a measure instead of px, although px is doable too. em is just for more responsive sites.

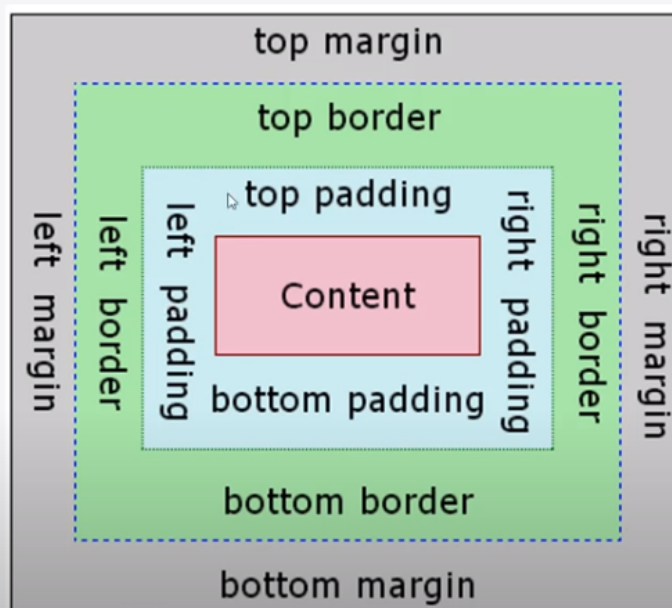
```
body{
  line-height: 1.6em;
}
```

Text Transform, Text Decoration, Spacing and Font Style:

```
.box-1 h1{
  font-style: italic;
  text-decoration: underline;
  text-transform: uppercase;
  letter-spacing: 0.2 em;
  word-spacing: 1em;
  line-spacing: 1.16em;
}
```

## Box Model

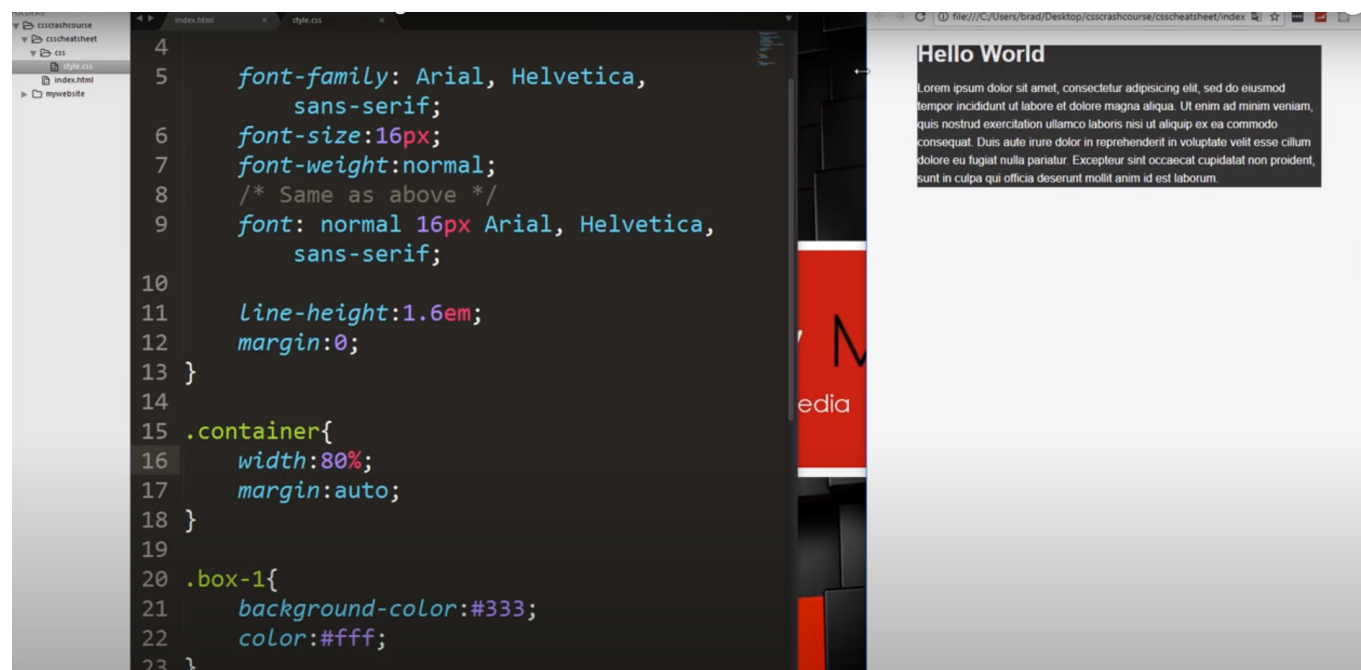
# Box Model



## Centering DIVs / elements:

wrap the elements in a new `div` tag and give it a class name. Let's call this class as `container` for instance. Now create a new style with the class name and enforce the rule of margin.

```
.container{
  width : 80%;
  /* to not to make it go all the way across the page */
  margin: auto;
}
```



## Margins

Shorthand notations goes like: margin-top margin-right margin-bottom margin-left;

```
a{  
    margin: 5px 10px 5px 10px;  
}
```

Since the top & bottom and left & right are same, this could be written as:

```
a{  
    margin: 5px, 10px;  
}
```

This is same as:

```
a{  
margin-top : 5px;  
margin-right : 10px;  
margin-bottom : 5px;  
margin-left : 10 px;  
}
```

## Border

Example:

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

## Links & States

```
.categories a{  
    text-decoration: none;  
    color: #000;  
}  
  
.categories a:hover{  
    color:#333;  
}
```

```
.categories a:active{
  color: #444;
}

.categories a:visited{
  color:#898989;
}
```

## List

Not much to learn from here. although check out [list-style](#).

```
li{
  list-style-image: url('../checkmark'); /*gonna use the real size*/
}
```

## Forms

Forms are basically very ugly by nature. Let's make them look a little bit better.

HTML:

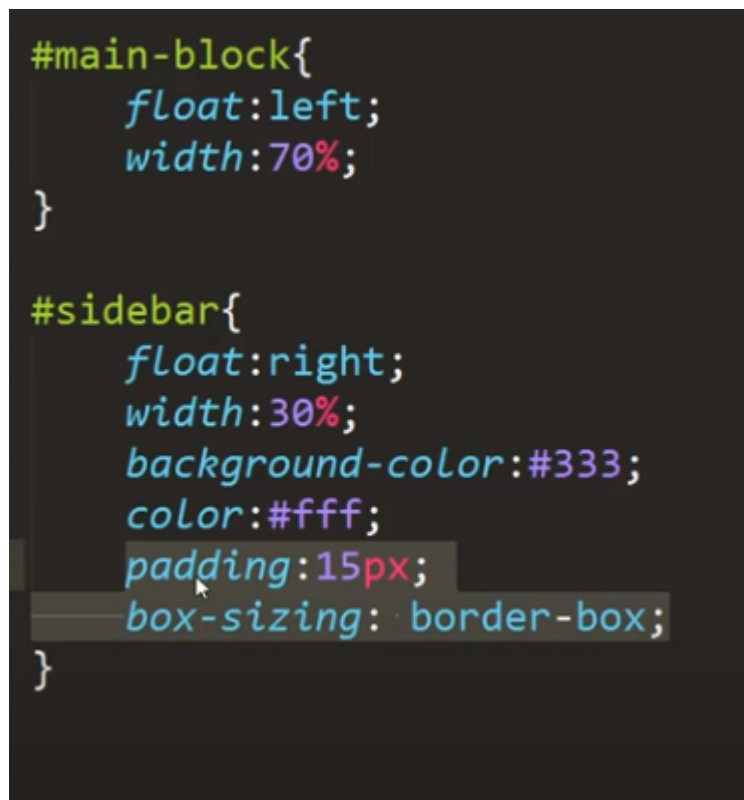
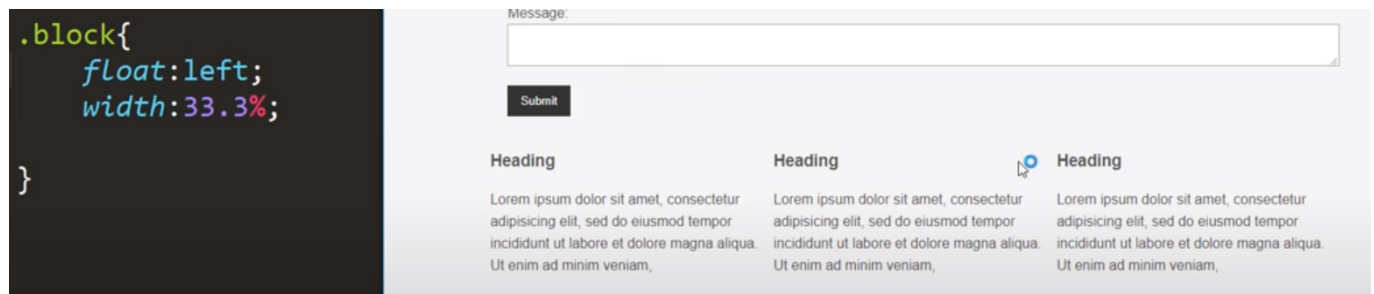
```
<form class = "form">
  <div class = "form-group">
    <label>Name</label>
    <input type = "text" name="name">
  </div>
  <div class = "form-group">
    <label>email</label>
    <input type = "text" name="email">
  </div>
  <div class = "form-group">
    <label>Message</label>
    <textarea name="message"></textarea>
  </div class = "form-group">
  <input class="button" type="submit" value="submit">
</form>
```

CSS:

```
.form{
  margin:20px;
  padding:20px;
}
```

```
.form .form-group{  
  padding-bottom:20px;  
}  
  
.form .form-group label{  
  display:block;  
}  
  
.form .form-group input[type="text"], .form textarea{  
  width:100%;  
}
```

## Floating and Border Box



## Heading

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

Lorem ipsum dolor sit amet, consectetur adipisicing elit, sed do eiusmod tempor incididunt ut labore et dolore magna aliqua. Ut enim ad minim veniam,

## Position

Types of positionings are:

Static

Relative

Absolute

Fixed

Initial

Inherit

## Pseudo Classes

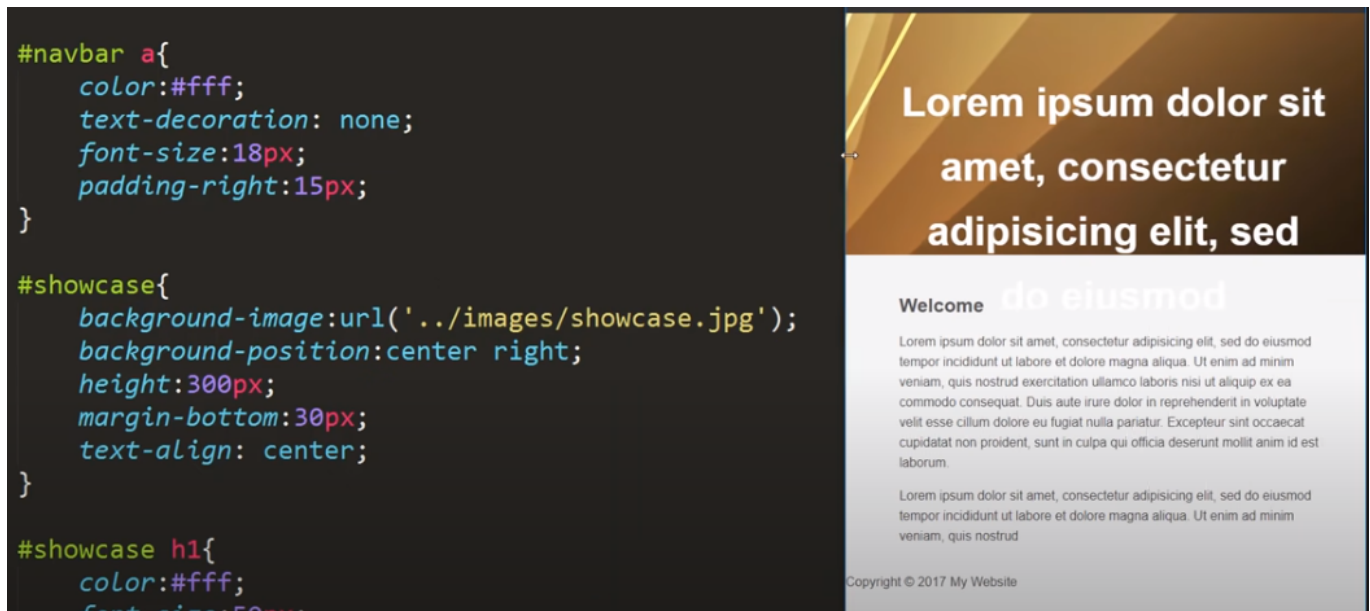
```
.my-list li:first-child{  
    background: red;  
}  
  
.my-list li:last-child{  
    background: blue;  
}  
  
.my-list li:nth-child(5){  
    background: yellow;  
}  
  
.my-list li:nth-child(even){  
    background: grey;  
}
```



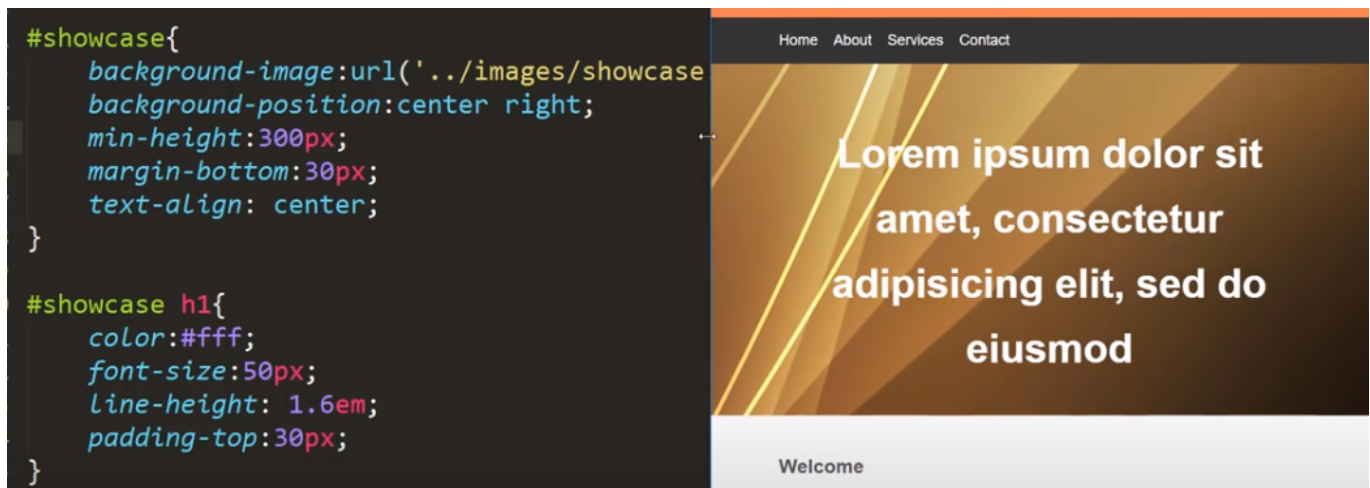
## Min-Height

use min-height instead of height if you want your images to scale along with the text and the window size when moved around.

Before min-height:



After min-height:



When you resize your window really thin, you might not wanna have a sidebar/aside floating to the right. So, you could bring it to the bottom/top.

Follow the following steps:

```

@media(max-width:600px){
  #sidebar{
    width: 100%;
    float:none;
  }
  #content{
    width: 100%;
    float:none;
  }
}

```



**IMPORTANT****FlexBox**

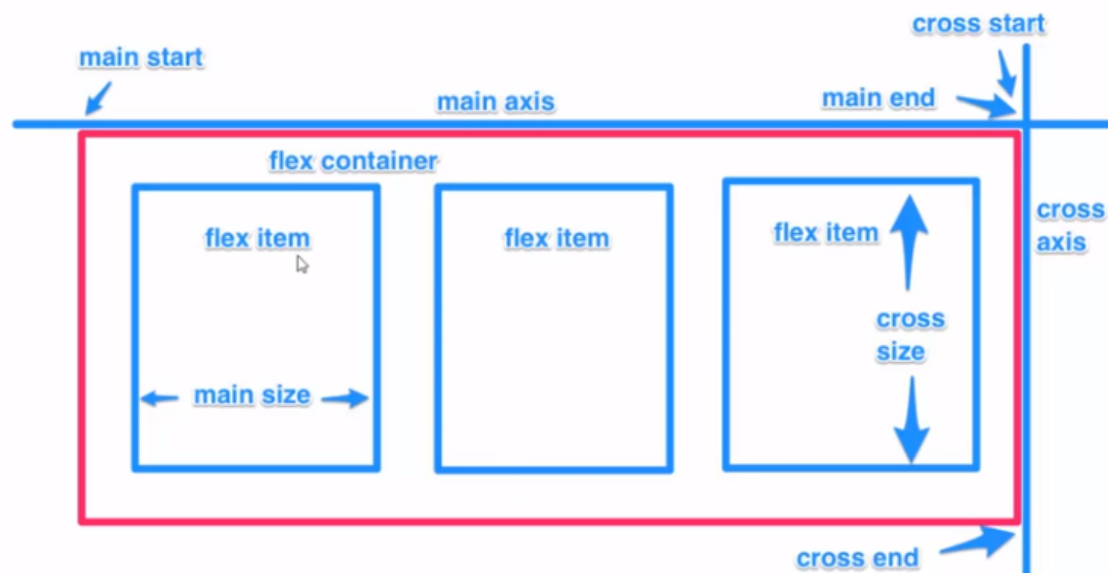
URL to revise quickly:

<https://www.youtube.com/watch?v=JJSoEo8JSnc>

## WHAT IS FLEXBOX?

A CSS3 layout mode that provides an easy and clean way to arrange items within a container

- ✓ NO FLOATS!
- ✓ Responsive and mobile friendly
- ✓ Positioning child elements is MUCH easier
- ✓ Flex container's margins do not collapse with the margins of its contents.
- ✓ Order of elements can easily be changed without editing the source HTML



# FLEX PROPERTIES

`display: flex | inline-flex;`

`flex-direction: row | column`

`flex-wrap: wrap | nowrap | wrapreverse`

`flex-basis: <length>`

`justify-content: flex-start | flex-end | center`

`align-self: flex-start | flex-end | center`

`align-items: flex-start | flex-end | center`

`align-content: flex-start | flex-end | center`

`flex-grow: <number>;`

`flex-shrink: <number>;`

`flex: <integer>;`

`order:<integer>;`

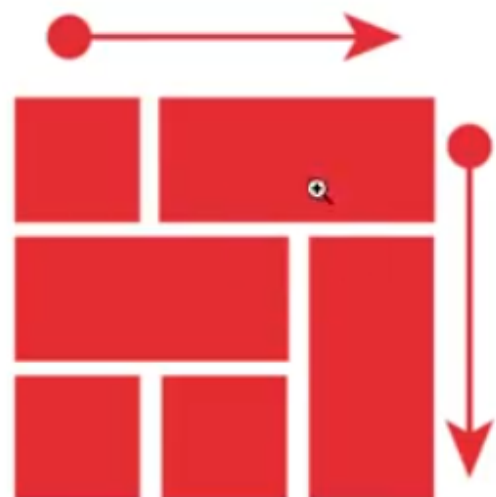
## Grid

URL to revise quickly: <https://www.youtube.com/watch?v=jV8B24rSN5o&t=2s>



Flexbox

ONE DIMENSION



CSS Grids

TWO DIMENSIONS

Basics:

```
.wrapper{  
  display:grid;  
  grid-template-columns:1fr 2fr 1fr;  
  grid-auto-rows:minmax(100px, auto);  
  grid-gap:1em;  
}
```