skillcrush

CASCADING STYLE SHEETS (CSS)

cheatsheet

INTRO

This is a complete cheat sheet to demystify CSS and how certain attributes work, to get you feeling comfortable diving right into adding your own styles to your site!

ANATOMY OF A CSS BLOCK

HERE'S A CSS BLOCK:

```
HTML tag {
   property1: value;
   property2: value;
   property3: value;
   property4: value;
}
```

WHICH MIGHT LOOK LIKE THIS:

```
p {
   color: purple;
   width: 200px;
   font-size: 24px;
   font-family: comic sans;
}
```

TEXT STYLES

color

Sets the font color. Accepts either the color name or the hex value of the color.

```
color: purple;
color:#800080;
```

font-family

Sets the font for the text. Will accept any web safe font (which is about 20, depending on the browser) - we will cover all the web safe fonts in the next step. Can be a prioritized list of fonts; if the first font listed is not available in the browser, it will default to the second, and so on. You can also write `sans-serif` or `serif` and the font will default to the browser's sans serif or serif font.

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

text-align

Sets the alignment of the text – `left` (default), `right`, `center`, or `justify` – the same options you have in Word.

```
text-align:center;
```

font-style

For if you want to italicize your text CSS instead of HTML tags. `Font-style` takes the values `normal` (default), `italic`, and `oblique`.

```
font-style:italic;
```

font-weight

Style your text **bold** without the HTML tags; takes the values `normal`, `bold`, `bolder`, `lighter`, or a number from `100` to `900`.

```
font-weight:bold;
```

text-transform

Allows you to make all your text uppercase or lowercase; takes the values `capitalize`, `uppercase`, and `lowercase`.

```
text-transform:uppercase;
```

font-variant

This is a fun one - you can actually make your text small caps by giving this attribute the descriptor: `small-caps`.

```
font-variant: small-caps;
```

text-decoration

`Underline`, `overline`, `line-through`, `none`. This can be used to remove the default underline on a elements.

```
text-decoration:none;
```

line-height

The default is for the line-height to be equivalent to the font-size. If you want to make it smaller or bigger, you set it with the `line-height` property. Accepts pixels, em, numbers, and percentages.

```
line-height:150%;
```

text-indent

Sets the width of the tab indent - takes pixels, ems, or percentages

```
Text-indent: 3em;
```

word-spacing & letter-spacing

Sets the space between words or letters. Accepts pixels or ems, also accepts negative values.

```
letter-spacing:-2px;
word-spacing: 3px;
```

CLASSES & IDS

Classes

A class is a group of elements that are the same or similar. You can have as many elements as you want in a class. And each element can be the member of multiple classes. Every class has CSS attributes that are specific to that class.

How to add them to your HTML:

```
<div class="bio">
```

CSS Example:

```
.bio {
   width: 100px;
   height: 300px;
   background: purple;
}
```

IDs

An ID is a singular identifier of ONE html tag. You can only have one HTML tag per ID per page, and each HTML tag can only have one ID. Each ID has a specific set of CSS attributes that ONLY apply to that one element.

How to add IDs to your HTML:

```
<div id="main">
```

CSS Example:

```
#main {
   width: 200px;
   height: 200px;
   background: blue;
}
```

Class & ID naming tips

- You can't start a name with a number. That one is non-negotiable, CSS just won't let you get away with it.
- Use a name that relates to what the element is, not how it looks. In the example below, what if you wanted to change the color of the alert from red to yellow?

GOOD

```
<span class="alert">Uh oh, looks like rain tomorrow.</span>
```

BAD

```
<span class="red">Uh oh, looks like rain tomorrow.</span>
```

CSS TRICK

The tag lets you add style portions of inline text. So if you want only a few words to have a certain kind of styling, the span tag is what you need.

For example:

```
<!--Your HTML looks something like this-->
<p>In this paragraph, I want <span class="emphasis-1">these words to be
italicized and green</span>. Nothing more, nothing less. Well <span
class="emphasis-1">maybe this part too</span>.</p>
```

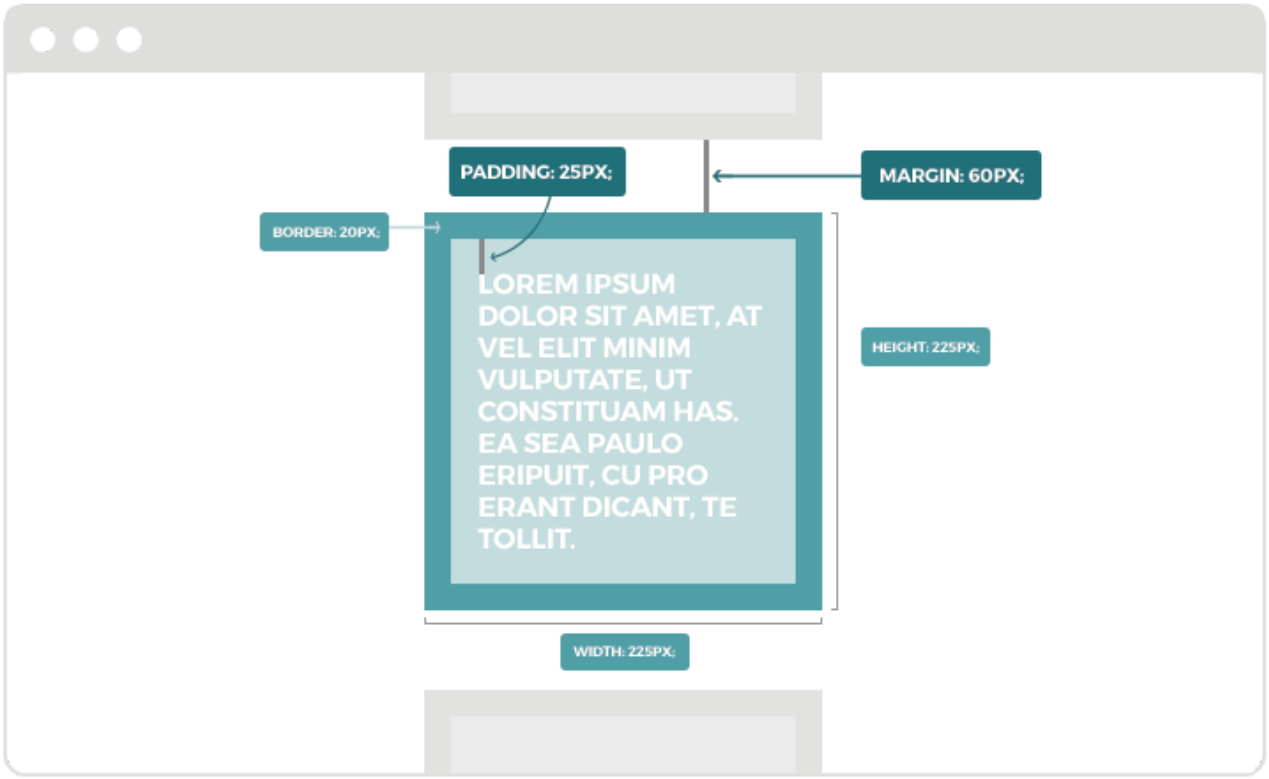
```
/*Your CSS looks something like this*/
.emphasis-1 {
    font-style: italic;
    color: green;
}
```

And the result will look like this:

In this paragraph, I want *these words to be italicized and green*. Nothing more, nothing less. Well *maybe this part too*.

BOX MODEL

Every element in an HTML document can be considered a box. Each box has characteristics like width, height, border, margin and padding. The values of these characteristics can all be altered with CSS.



Height & Width

These set the height and width of an element, and they accept either pixel dimensions or percentages. Percentages set the height and width relative to its parent element.

```
#my-box {
   width:200px;
   height:200px;
}
#my-box-inside-my-box {
   width:100%;
   height:20%;
}
```

Margin & Padding

`margin` sets the distance between one element and the adjacent element, as if it's pushing elements away from one another. `margin` accepts pixels or percentages.

`padding` makes the element itself bigger. It also accepts pixels or percentages.

```
#my-box {
   width:200px;
   height:200px;
   padding:20px;
   margin:10px;
}
```

If you want to set different margins or padding on each side of your element, you can do it like this:

```
margin: margin-top margin-right margin-bottom margin-left;
```

For example:

```
#my-box {
   margin:10px 20px 15px 15px;
}
```

The same can be done with padding.

CSS TRICK

You can use `margin: auto;` to center your entire website in the browser window. To do this you need to give the container div a set width & then set the margin to auto. Like so:

```
#container {
   width: 960px;
   margin: auto;
}
```

Backgrounds, Borders & Colors

Backgrounds are transparent by default. They can be set to a color (name or hex value), or an image.

```
1  background: image-url repeat x-position y-position background-color;
```

All of the following are correct examples of how to use the background CSS property:

```
1  background: #00FF00;
```

```
1  background: blue;
```

```
1  background: url('img/background.jpg') repeat #00FF00;
```

```
1  background: url('http://skillcrush.com/image/background.jpg')
   no-repeat 30px 40px;
```

Borders

You can set the width of your border, the type of border and the border color. Borders can have any pixel dimension, can be any hex or web safe color, and can be `solid`, `dashed`, `dotted`, `ridge`, `groove`, `inset`, `outset`, or `double`.

```
1  border: pixel-width border-type border-color;
```

Examples:

```
1  border: 3px dotted pink;
```

```
1  border: 1px solid #000000;
```

LAYOUT

Floating

Allows you to float elements alongside one another, which is key to creating multi-column web layouts. You can float elements using the values `left`, `right`, or `none`.

```
#main-content {
   width: 600px;
   float:left;
   margin-right: 30px;
}
```

Warning: floating an element remove it from the document flow. This can cause all sorts of weird behavior on your website. Use the clear or overflow properties to fix these issues.

Clear

Tells your element to stay clear of floated elements. Takes the values `none`, `both`, `left`, or `right`.

```
.clear-fix {
   clear:both;
   width:100%;
}
```

Overflow

Useful for telling an element what to do with its child elements that may overflow their container. `overflow: auto;` tells a parent element to automatically increase to the size big enough to encapsulate all of the floated elements within it.

Overflow takes the values `auto`, `scroll`, or `hidden`.

```
.parent-div {
   overflow: auto;
}
```

Position

The position property sets the position of the element relative to its parent, the website, or the browser window. The default position for all elements is static. Relative sets the position relative to its parent element, absolute sets the position relative to the website, and fixed sets it relative to the browser window.

`position` takes the values `static`, `relative`, `absolute`, or `fixed`.

In order to then set the element's position, you need to use the top, left, bottom & right properties. They each accept pixels or percentages.

- `top: 10px;`
- `left: 40%;`
- `bottom: 0px;`
- `right: 10%;`

Example:

```
#my-modal {
   position: fixed;
   top: 10px;
```

```
4     left: 500px;
5  }
```

CSS TRICK

In order to make sure a modal always sits in the middle of the page, the best thing to do is give it a position of fixed, with `left: 50%;` and then set a negative margin equivalent to half of its width.

For example:

```
1  #my-modal {
2     position: fixed;
3     width: 300px;
4     top: 10px;
5     left: 50%
6     margin-left: -150px;
7  }
```

z-index

The CSS property `z-index` sets an element's position on the z-axis, meaning you can set elements to sit above or below other elements on the website. Z-index accepts numerical values from negative infinity to infinity.

```
1  #my-modal {
2     position: fixed;
3     width: 300px;
4     top: 100px;
```

```
5     left: 50%
6     margin-left: -150px;
7     z-index: 99;
8  }
```

Every CSS property has a value of inherit, which we did not list in the above examples. If a property is set to inherit, it will inherit the value of its parent, when perhaps it otherwise would not.

Example:

```
1  <body>
2  <p> Hi, my name is Grace Hopper. I was a US Navy
   officer and was one of the first programmers to work on the Mark I
   computer at Harvard. I am credited with coining the term “debugging”
   for fixing software problems. </p>
3  </body>
```

```
1  body {
2     margin: 10px;
3  }
4  p {
5     margin: inherit;
```

```
6   }
```

A COUPLE OF MORE NOTES

This cheatsheet is complete but not comprehensive! If you want to learn more about CSS, mosey on over to the W3Schools website to read all about ‘em.

And, remember, you are exactly the type of person who should be coding! You are smart, creative, and love to solve problems.

We can’t wait to see what you make!