

BUT FIRST

Do you remember HTML?

TRY OUT ASSIGNMENT 01

SEMANTICS

```
<main>
<section>
 <div>
  Content A
 </div>
 <div>
  Content B
 </div>
</section>
</main>
```

Think about how elements nest, why would developers do this?

TRY OUT ASSIGNMENT 02

HTML TABLES

I made one for you on Codepen: http://codepen.io/staypuftman/pen/NNwzYj

Go check it out!

TABLE THINGS TO KNOW

Tables are like mini-HTML docs:

```
<thead></thead>
```

The head contains the row of identifying columns

The body contains the rows of content

TRY OUT ASSIGNMENT 03

What are the two most important things to remember about HTML?

IT'S NOT CODE! SEMANTICS MATTER.



ANATOMY OF CSS

```
p { color: purple; }
Selector Property Value
```

WHAT STYLES LOOK LIKE

```
p {
  color: white;
  text-decoration: underline;
}
```

Which one is the selector, value, property?

```
p {
  color: white;
  text-decoration: underline;
}
```

Selectors tell the browser where to apply the style.

```
p, div {
  color: white;
  text-decoration: underline;
}
```

You can have two selectors together - mixing HTML elements and custom classes if you like.

```
p, div, .my-class, #yourID {
  color: white;
  text-decoration: underline;
}
```

You can have an unlimited number of selectors together, just separate with a comma. You can also include classes and IDs.

```
div.my-class {
  color: white;
  text-decoration: underline;
}
```

What is being selected here?

YOUR TURN ASSIGNMENT 04

PROPERTIES

```
p {
  color: white;
  text-decoration: underline;
}
```

Properties are parts of the CSS spec that control style behavior. Most are logically named, but not all.

PROPERTIES are

ALWAYS

followed by a colon and a



PROPERTIES + VALUES

```
p {
   color: white;
}
```

You must have at least one property / value pair per selector (otherwise nothing happens). They MUST end with a semicolon - ALWAYS!

PROPERTIES + VALUES

```
p {
  color: white;
  font-weight: bold;
  padding: 15px;
  text-decoration: underline;
}
```

You can have an unlimited amount of property/value pairs in a style declaration.



IMPORTANT PROPERTIES

```
background - what's behind an elements
border - edge of an elements
color - sets color of text
font - controls font family, size, style, weight
height - how tall an element is
margin - space between two elements
padding - space inside of an element
text-align - which direction text lines up
text-decoration - underlines or crosses out text
text-transform - controls capitalization / uppercase
```

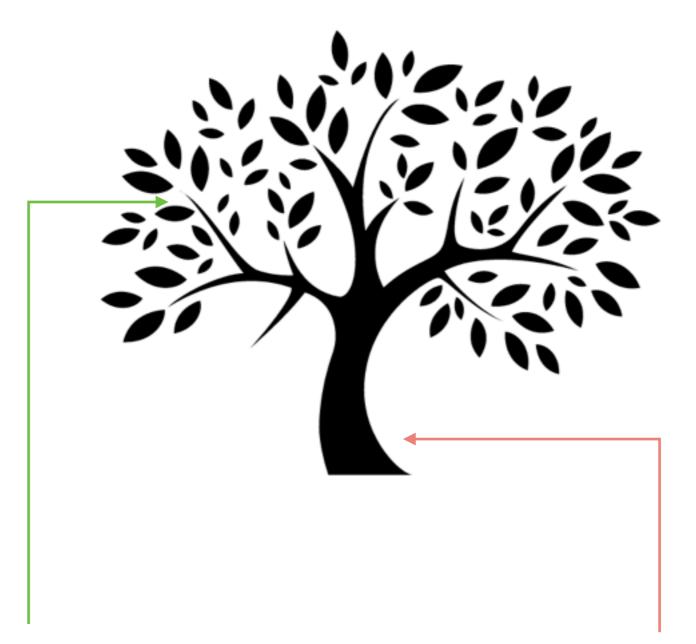
QUICK STYLING TIPS

- Always put a new style on a new line
- Try to alphabetize your styles
- Use external stylesheets
- Use white space so people can read your code - think logically.

YOUR TURN ASSIGNMENT 05



Think back to the DOM Tree



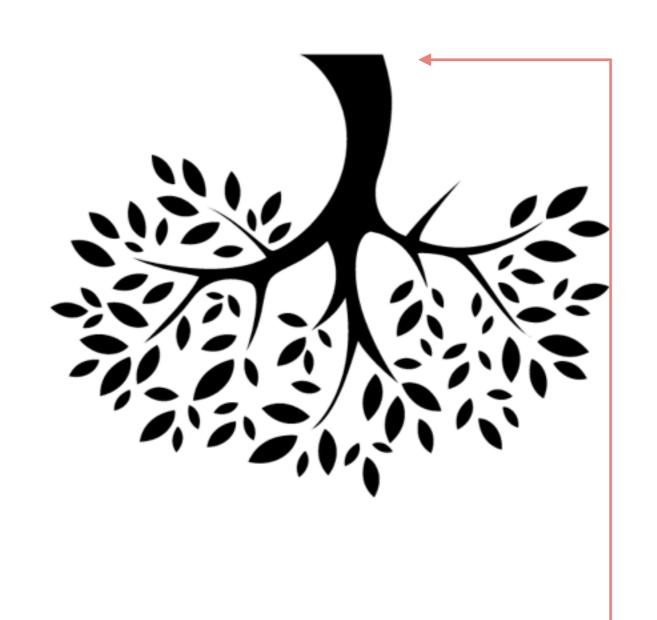
What if I apply my styles at each of these spots? What happens?

The more general your selector is (like body, img, or p) the more your style will

CASCADE

throughout your HTML document(s).

Now flip the tree...



What if I don't apply a style at the branch? What if I only style the trunk?

If you don't styles your branches, they will

INHERIT

parent styles from structural elements like body, img, or any parent element.

TRY IT OUT

Let's pair up and draw a DOM tree



BASIC LINKAGES

For stylesheets (CSS):

<link rel="stylesheet" href="styles/main.css">

ABSOLUTE VS RELATIVE

Relative paths are missing the http:// stuff:

```
<link rel="stylesheet" href="styles/main.css">
```

Absolute paths have the http:// stuff. You'll generally see them with anchor tags, like this:

```
<a href="https://google.com">Google</a>
```

ABSOLUTE VS RELATIVE

Relative paths can use dot notation to reference folders above them

```
<link rel="stylesheet" href="../styles/main.css">
```

TRY IT OUT

Make some relative links and folders in Assignment 05

HOMEWORK TIME

Start on your homework and see where you have questions.

FOR NEXT TIME

HW Assignment Complete
CSS Box Model
CSS Layouts
Come to Office Hours!