HTML & CSS

Note:
The difference between HTML and CSS
For example,
_____
HTML uses p to display a paragraph of text on the web page.
Then CSS uses a type selector to determine the style of the content (color, front size, front weight, etc.).

1. Common HTML Terms:

(1) Elements

- Elements are designators that define objects within a page, including structure and contents.
- Every element takes a specific area of rectangle in the web page. The browser implements everything as a rectangle.
- Types:
  - Block Level Elements: Start a new line, which means fill the whole width of the screen (A line takes up a whole width of the screen). Ex: , , <h1>
  - Inline Level Elements: Start a content without creating a new line. Can be created inside a block element. Ex: , <b>
  - Example:
    - ```
      My mother has <span
      style="color:blue;font-
      weight:bold">blue eyes and my
      father has <span
      style="color:darkolivegreen;font-
      weight:bold">dark green eyes.
      ```
    - It will print as this: My mother has blue eyes and my father has dark green eyes.

(2) Tags
      ```

Elements are often made of multiple sets of tags.
Ex: `<a>__SOME CONTENT__</a>`
Opening Tag: `</a>`
Closing Tag: `<a>`

(3) Attributes
Attributes are properties used to provide additional instruction to given elements.
Example:
`id`, `class`, or `title` to an element, to give media elements a source (`src`), or to provide a hyperlink reference `(href)<a href="http://www.shayhowe.com/">Shay Howe</a>`
It will display as:
Shady Howe
(Click on the words it will redirect to a link specified in the code)

2. HTML Document Structure

```
<!DOCTYPE html>
<html lang="en">
	<head>
		<title>Yoooooo</title>
		<link rel="stylesheet" type="text/css" href="style.css">
	</head>
	<body>
		<h1>Dudes and Dudetees</h1>
		<p>Wheeeeee</p>
	</body>
</html>
```

Another Example:

```
<!DOCFILE html>
<html>
```

```
      <head>
      <title>____</title>
      <link rel = "stylesheet" type = "text/css" href
= "stylecss"
      </head>

      <body>
           <div class = "__________">__________</div>
           <h1 id = "_______">___________<h1>
           <p>____________</p>
           <div class = "____">_____</div>
      </body>

</html>
```

Note:

- Same colors correspondent to a pair of opening/closing tags.
- DOCTYPE: instruct the browser which version of HTML should be used
- <head>: outline meta data and links to any external files (<link>). The content within <head> is not visible in the actually webpage. The visible contents are in <body>.

3. Common CSS Terms

(1) Selector
A selector determines exactly which element, or elements, the corresponding styles will be applied to.
Selector contains the name of the corresponding HTML element/elements, curly braces.
Example:
```
p {......}
```
Correspondence to the HTML element <p>

- Type selectors
  - HTML: `<p>…… </p>`
  - CSS: `p {……}`
- Class selectors
  - Class selectors allow you to apply the same style to an array of elements. It will change all the elements with the same class attribute at one time.
  - HTML: `<div class="awesome">…</div>`
  - CSS: `.awesome {…….}`
- ID selectors
  - ID selectors are similar to class selectors however they are used to target only one unique element at a time.
  - HTML: ….
  - CSS: `#whee {….}`
  - Note: id attribute is very unique, it's only used on one specific element for some special use (like for referring to that element later). So changing the style of element with id attribute only affects this unique element.

(2) Property and Values
The style of a element. Like color, front type, etc.
Values determine the behavior of the property. Like what color it is, what front size it is, etc.
Using the selector to determine the property of element:

```
p {
    color: #ff0;
    front-size: 16px;
    position: xxxxx
}
```

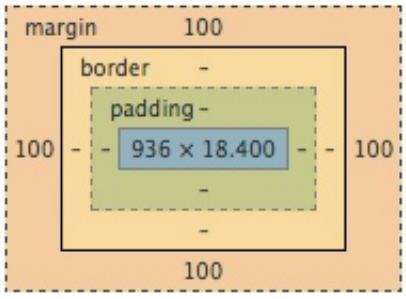
(#ff0 and 16px: values
color and front-size: properties)

Some common types of property:

- Size of the rectangle
  - `element.style {`

```
        padding: __px;
        margin: __px;
        width: __px;
        height: __px;
    }
```

- Padding: Clears an area around the content.
- Margin: The margin (literally) of the element. Changing the size of the margin means changing how big the margin is. Change margin will move the content. The margin does not have background color, it's transparent.
- Width: the width of the rectangle directly containing the element. If we want a single line element to have multiple rows, we can make width smaller so that the words going off the rectangle will go to the next row.
- Height: the height of the rectangle directly containing the element. If the element contains many rows, when we make the height smaller, the content will shrink (not the actual front size of the words within the content. That's another style of the element: front-size)
- Note:
  - Margins will overlap with each other. But the rectangles that directly contain elements will not overlap.



- Background color
  - `background.color: ___`
  - Change the background color of the rectangle that directly contains the element
- Bound Shape
  - `bound.radius: ___px`

    - Make the bound of the rectangle directly containing the element round. But browser still implements as a rectangle even if the shape is not exactly a rectangle.
- Position Property
    - `position: fixed`
        - Element stuck on the screen.
        - top/right/left/bottom: __px ---Distance to the borders of the screen
    - `position: absolute`
        - The element is positioned relative to its first positioned (not static) parent element
        - top/right/left/bottom: __px ----Distance relative to the parent element
        - Note: If the parent element is fixed, the element will stuck on the screen along with the fixed parent element
    - `position: static`
        - Default value. Elements are displayed in order, as they appear in the document(code) flow.
- Display Property
    - `visibility: hidden`
        - does not take out of flow, still affects the flow of page (physically there but it's invisible, like physically blocking other elements from overlapping each other though it's invisible).
    - `display: none`
        - take the thing out of flow. It won't affect the flow of page because it simply disappears (Other elements can just cover above the display-none element because the element has disappeared).
    - `display: block/inline`
        - Turn the element into block or inline

Note:

- Comments in HTML/CSS code

HTML comments wrap the content starting with `<!--` and end with `-->`. CSS comments wrap the content starting with `/*` and end with `*/`.