**Chapter 4:**

**Intro to CSS (cascading style sheets):**

Used to style whereas markup languages are used to program

<body style=”background-color: (content);”>

Style=”(CSS code);” = changes background color of the body portion of your html file using CSS code.

Can use default colors, #(hex values)

You can change styling in the head of your html document to affect everything in the body. If you have a specific change to an element in the body, it will overwrite the internal rule. The most specific changes are kept. Inline CSS – Same line; Internal CSS – in head tag; External CSS – External stylesheet.

Example:

<head> <style>

body { background-color: fuchsia; }

hr { border-style: dotted none none;

width: 5%;

border-width: 5px; }

</style> </head>

You can customize the global css by creating a css file and pasting the css code in it. Then using the link tag, you can link to that style sheet. The CSS file contains everything in the <style> tag.

<link rel="stylesheet" href="(css file locate)">

**Proper CSS Syntax:**

selector { property : value; }

who? { what? : how? }

selector = who or what is being modified (ex: <h1>)

Rules are in the brackets. They will change one property by giving it a new value.

Each rule ends with a semicolon. One rule per line. It’s good practice to organize alphabetically.

**Tags vs Class vs ID:**

You can style HTML elements (tags), classes or ID’s to differentiate between similar elements.

Tags are addressed as the tag name

Classes are addressed as .className

Classes can be used to group elements. Can have multiple classes per element.

Ex: <img class="bacon" src="(link)” alt=”bacon pic”>

.class{ Background-color: red; }

ID are addressed as #IDName

ID’s are unique. Can only have 1 ID per element. ID’s cannot be shared.

Ex: <h1 id="heading">I Love Bacon</h1>

#heading{ color: blue; }

ID > class > tag selectors in terms of precedence as ID is most specific

**Pseudo-class:**

Class assigned to elements with multiple states such as when an element is hovered over by the mouse

:(pseudo-element name)

Ex: .bacon**:hover** { background-color: green; }

**Chapter 5:**

**FavIcons: (favorite icon)**

Originally used in the favorites bar. Now displays on website tab

Create your own favicon at favicon.cc

Link to it with the link element

Ex: <link rel="icon" href="favicon.ico">

Browsers have their own style sheet called ‘user agent stylesheet’ which just give default styling values.

**Div element:**

The <div> element can group different elements together to structure your website more specifically.

<div class="topContainer"> <h1> I'm Mohanned </h1> <p> a cool guy </p> </div>

**The Box Model of website styling**

Different elements of a website are all boxes. Format the boxes to style your website.

**Block display elements:**

Paragraph (<p>), headers (<h1> to <h6>), divisions (<div>), lists and list items (<ol>, <ul>, <li>), forms (<form>)

Extend the entire width of the screen. Height is dependent on the content

You can edit parts of a block element using the <span> element

Ex: <p> a <span class="co">co</span>ol guy </p>

**In-line display elements:**

Take only as much space as needed to contain its elements

<img>, <span> and <a> (anchor)

Unable to adjust width of inline elements like you can for block elements

**In-line block display elements:**

Best of both world. Inline but can be adjusted similar to block elements (ex: width)

**None display elements:**

The element isn’t displayed at all.

This is different from the visibility selector as that makes your element invisible, but it still takes up space

**Rules of rendering elements:**

Content dictates size for inline elements

Order of elements is determined by placement in the code

Children elements are displayed over parent elements

**Positioning:**

Static – HTML element default position.

Relative – Position is changed relative to default position of element

4 coordinate properties (top, bottom, left, right) and value (px or %)

Element is moved away from the border direction stated in the amount stated

Ex: position: relative; left: 30px; moves image 30 pixels away from it’s default left border

This has no effect on other elements. Other elements will behave as if image was never moved

Absolute – Element position is changed relative to border of its parent element

This DOES affect the position of other elements. Absolute elements behave as if they are not there when determining placement of static or relative elements.

On the contrary, absolute elements are affected/pushed by static or relative elements.

Must have a parent which is relative otherwise absolute elements will compare to body.

Fixed – Stays in the same spot relative to website. It does not move when you scroll the page.

Useful for navigation bars or side bars.

**Centering elements:**

text-align: center; is placed in the parent element to center the contained elements.

Only centers elements that do not have specified width.

To center elements with specified widths, you must change margins (ex: margin: auto;)

**Font:**

font-family: (font 1), (backup1), (backup2);

You can embed fonts using google to ensure people can load the fonts you want.

<https://fonts.google.com/>

Add the embedded link in the head of your html file.

**Sizing:**

Font-size: (#)px; = changes size of font

#px is a static size. If user changes size of font, this will not change.

You can use % instead when specifying size to have text be dynamic.

You can also use the unit #em for dynamic scaling as well.

16px = 100% = 1em

You can change the font-size in the parent element and it will have multiplicate effects if you’re also changing font size in the child elements. This is not true for static sizing (#px). Static sizing is only affected by the most specific selector.

This can be worked around by using #rem which changes the root em. This is not affected by any upstream size changes such as in the parent element.

**Text wrapping:**

float: Allows images to float around text to have text wrap around an image.

Can float to left or right.

clear: Tells element to clear anything in direction stated. This forces text not to wrap around an image.

Can clear to left, right or both.

**Button generator website:**

<https://cssbuttoncreator.com/>

make sure to click beautify after implementing the button.