* **What is HTML?**
  + HTML is the structure of the content that goes inside tags.
  + Browsers provide default style, but it’s pretty ugly by itself.
* **Headers**
* <h1> These marks allocate headers </h1>

<h6> you can go all the way up to header 6 </h6>

**Paragraphs**

* <p> is used to allocate paragraphs </p>
* **Inputs**
* for any input you use the tag input, i.e. for our email input we state

<input type ="email">

*BUT WATCH OUT!!* Don’t put spaces next to your equal signs!!!

<input type="email">

* + - The part type is called an **attribute**
    - The part “email” is called the attribute’s **value**
* ??? We use the tag <input type=”submit”> to make a button that says submit which is confusing because that is not a user input, it is a button which I would think would be a different tag.
* **Attributes**
* Think of attributes as options for each tag

e.g. if there were an HTML tag for pizza it may look something like that which is shown below

* <pizza size=”large” crust=”thin” type=”pepperoni”>
* **Placeholder**
* There is an attribute called placeholder that allows us to add default text to our input

e.g. <input type=”email” placeholder=”Your email”>

* **What is CSS?**
  + CSS controls the style of the HTML content
  + It lets you change colors, fonts, layout, and more.
* **<style> tag**
  + The easiest way to add CSS to a small project
  + A style tag must go on the first line before any other code!!!

^ kind of a lie, <head> and <!DOCTYPE html>, go 1st

* + A style tag looks like this:

<style></style>

* + Exmaple:

<style>

h1{

text-align: center;

}

</style>

⇨ h1 is an example of a **selector**, the selector is quite self-explanatory… it simply “selects” the HTML elements we want to add style to. I have selected h1, which if you can remember was:

* + - <h1>Anna Dowlin</h1>

⇨ { } are the **curly brackets** anything inside of these curly brackets will stylize the elements selected by your selector

⇨ text-align: is an example of a **property**. A property controls one aspect of the HTML element’s style, for example we can change the text-align, color, width, background, etc.

⇨ center; is an example of a value. A **value** goes with a property. For example, here the value for text-align: could have been left, right, center or justify.

* **Multiple styles in one <style> tag**
  + You can have multiple styles in the same <style> tag

Example:

<style>

h1{

text-align: center;

}

p{  
text-align: center;

}

</style>

* + If we are centering multiple different items at the same time we can make a parent element that contains all the other elements
* **The <body> tag**
  + HTML gives us a tag, <body> that serves that purpose
    - The <body> tag wraps around all of the contents on the page.

Example:

* + - * <body>
      * <h1>Anna Dowlin</h1>

<p>Hi! I'm Anna, a NYC-based marketer. Say hello!</p>

* + - * <input type="email" placeholder="Your email">
      * <input type="submit">
      * </body>
  + If we then write:

<style>

body{

text-align: center;

}

</style>

* + It will center the contents included in <body>, in this case the entire page.
* **The <head> tag**
  + Before body, we always start with another tag called <head>
  + <head> wraps around the <style> tag and other elements that are not content on the page itself, like the <title> tag.
* **The <title> tag**
  + The <title> tag controls what shows up in the browser tab.

Example:

<title>Anna Dowlin</title>

would show up like this:

Macintosh HD:Users:mollydavey:Desktop:Screen Shot 2015-05-13 at 1.58.28 PM.png

* + The <title> tag goes before the stile tag but after the <head> tag (see above)
* **Specifying the HTML type**
  + To have a valid HTML document, we need to tell the browser we’re using HTML5 (in this case)
  + To do so write:

<!DOCTYPE html>

on the first line before anything else!!!

* **<img> tag**
  + What you use to add an image

Example:

<img src=”/assets/anna.png”>

* + The <img> tag uses the image located at the URL in the src attribute
  + But wait, /assets/anna.png doesn’t look like a URL… where is the domain, etc. like <http://website.com>?
  + Well… there are two types of domains, **absolute** and **relative**
* **Absolute URLs**
  + <http://dash.ga.co/assets/logo.png>
  + Absolute URLs are the types of links you’re used to. The include http:// and the full domain name before the directory (/assets/logo.png)
* **Relative URLs**
  + /assets/logo.png
  + Relative URLs are shortcuts that allow you to skip the domain name. It only works if you’re linking to a file on the same domain as the current page!!
* **Background**
  + You can change the color or make the background an image

Example:

<style>

body{

text-align: center;

background: url(“http://dash.ga.co/assets/anna-bg.png”);

background-size: cover;

background-position: center;

color: white;

font-family: helvetica;

}

</style>

* + Editing the background image:
    - Add “background-size:” **property** (see above)
      * This will prevent the background from appearing tilted
    - Add “background-position:” **property** (see above)
      * This will center the background on the page
* **Editing the submit button and the input bar**
* <style>
* body {
* text-align: center; 🡨 centers all text
* background: url("http://dash.ga.co/assets/anna-bg.png"); 🡨 bckgd to img
* background-size: cover; 🡨 un-tilts the background image
* background-position: center; 🡨 centers the background image
* color: white; 🡨 makes text body white
* font-family: helvetica; 🡨 makes all text Helvetica
* }
* p {
* font-size: 22px; 🡨 changes the paragraph font size to 22 pixels
* }
* input {
* border: 0; 🡨 gets rid of std formatting for input bar
* padding: 10px; 🡨 pads text in input bar by 10 pixels on each side
* font-size: 18px; 🡨 changes font size in input bar to 18 pixels big
* }
* input[type="submit"] {
* background: red; 🡨 makes submit button red
* color: white; 🡨 makes font on the submit button white
* }

</style>

* **External stylesheet**

How to add CSS to bigger projects

**Basic Structure of Every Webpage**

<!DOCTYPE html>

<head>

<!-- Meta-info goes here -->

</head>

<body>

<!-- Page content goes here -->

</body>

* **The <header> tag**
  + The <header> tag holds the website’s header
* **The <ul> tag**
  + <ul> is one of the most commonly used tags in HTML
  + ul stands for unordered list

Example: we used the <ul> tag to make our blog’s navigation bar

**The <li> tag**

* + - The <li> tag stands for list item
    - Each list item is delineated using <li></li>
  + Together the <ui> and <li> tags create bullet points
* **The <a> tag**
  + The <a> tag transforms text into links
  + <a> stands for *anchor* tag
  + The <a> tag has an attribute called href (*hypertext reference*) which controls the URL your browser opens when you click the link on the page

<a href=”<http://www.google.com>”>Google</a>

<ul>

<li><a href="#">About Me</a></li>

<li><a href="#">Best Poems</a></li>

<li><a href="#">Worst Poems</a></li>

</ul>

* **The <link> tag**
  + The <link> tag is another way of adding CSS style to an HTML file
  + The <link> tag lets you include CSS from an external file so that you don’t have to write all of that code in your HTML file, (crucial for larger projects)
  + The <link> tag always needs two attributes
    1. href – a URL where the CSS file lives
    2. rel – which should always be set to “stylesheet”

Example:

<head>

<link href=”/normaliza.css” rel=”stylesheet”>

</head>

* + normalize.css
    - A css file that “makes browsers ‘render all elements more consitstently an in line with modern standards’. That way your page will look the same, no matter what browser you use.”
* **The display: property**
  + Every HTML element has a default value for display. Usually it is “block” or “inline”
  + Block:
    - Elements like headings, paragraphs, and list items display “block” by default, which means they stretch the *whole width of the page* and have *line breaks* before and after.
  + Inline:
    - Links (the <a> tag) and a few other elements we haven’t learned about yet display “inline” by default, which means they exist within the normal flow of the text they’re contained within – no *line breaks,* no taking up the *whole width of the page*.
* **The padding: property**
  + When you pass padding just one value, (e.g. padding: 10px;) then you are telling the browser to add 10px of space on *all sides* of that element.
  + Sometimes you want to control each side’s padding individually. To do that. Simply give padding 4 values, separated by spaces:

Padding: top right bottom left;

* + Margin creates space on the outside of an element, padding creates space on the inside.
  + IMPORTANT TRICK TO REMEMBER:

Margin: 0 auto;

* + - The above property with the auto value will set a “0” margin on the top and bottom and the “auto” margin on the left and right
    - Since both left and right’s margin stretch all the way to the edge of the page (that’s what “auto” means) the effect is the text is centered on the page
* **Responsive Changes**
  + Change width: 500px; to max-width: 500px; and it will re-size itself when you re-size the window.
  + We can set media queries which are basically just like if statements that only occur if the browser is a certain size

Example:

@media (max-width: 500px) {

h1 {

font-size: 36px;

}

li {

display: block;

padding: 5px;

}

}

* **Hex Color Code**
  + #f00
  + always start with #
  + first digit: redness (0=0% f=100%)
  + second digit: greenness (0=0% f=100%)
  + third digit: blueness (0=0% f=100%)
  + #000 = black (because no color) #fff = white
* **RGBA Colors**
  + Add transparency
  + rgba(0, 0, 0, 0)
  + use a 0-255 scale and the last digit is transparency which uses a 0-1 scale (you can use decimals for the in-between transparencies)

Example: rgba(255, 255, 255, 1) makes white

* **What is Javascript?**
  + It allows webpages to be interactive by attaching a *back-end server* (queue Ruby on Rails)
* **Buttons in HTML**

Example: <button>Like</button>

* **The <script> tag**
  + This tag adds Javascript to our code similar to how <style> adds CSS to our code.

Example: <script>alert("Javascript works!")</script>

The above example creates a popup alert that Javascript was functional.

* **Interactive Websites with Javascript**

Example:

<script>

$("button").on("click", function() {

alert("clicked!")

});

</script>

* + First we select the element(s) whose event we want to listen to, the element goes inside quotes and parentheses with a $ to the left of it
  + Next we call the on(); function which sets up an event listener for the element we have called, inside it’s () are two options – sometimes called the *parameters* or *arguments*
  + .on(event-type, thing-to-be-done);
  + First parameter is the type to listen to (click, hover, scroll, etc.)
  + The type MUST BE IN QUOTES
  + The second parameter is the function which contains the thing to be done on the aforementioned event
* **The <div> tag**
  + Allows you to style a group of elements with the same CSS
  + One of the most multi-purpose tags we have
  + The <div> tag is non-semantic meaning you can use it as a container to group *any* type of content so it can be styled by a single bit of CSS
* **Class**
  + Let you *name* and *group* HTML tags

Example of naming a class:

<tag class=”name”>

tag’s attribute tag’s value

* + Class names start with periods

Example:

.first{

background: url([www.googleimage.com](http://www.googleimage.com));

}

* + When <div> and classes disagree the class overpowers the div commands
* *Cruft:* excess code that has no meaningful function
* Line-height:

Lets us change the vertical space between lines

* <br />

*self-closing tag* hence the slash inside, it opens and closes itself

* /\*

stop reading this CSS

* \*/
  + resume reading this CSS