**CSS**

**Review and Issues:**

Save all assets to the folder, upload the folder to dropbox

Close all tags

Saving and linking pages

**THE STORY SO FAR…**

* As we have learned, HTML is used to describe the content of web pages: headline, stories, links, images.
* In the past HTML was also used to determine how the content was presented: Font, font size, alignment, etc.
* We did this to gain a basic understanding of how HTML works, but this is no longer the best way to format web pages.
* Now we will begin to work with Cascading Style Sheets (CSS), the language for formatting web pages.

**CSS OVERVIEW (**css1.html)

* CSS allows us to create “styles” that can be applied to any HTML tag.
* Today we’ll be discussing three ways these styles can be applied: Inline styles, embedded, and external style sheets.
* In your Class 4 folder, open the file called css1.html
* Make sure you have split screen, and you will see that we have structure – H1, H2 and P tags, but no formatting.
* On the design side, we see only the default browser styles. A plain page.
* But using CSS styles (instead of HTML markup) we can design this page.

**INLINE STYLES**

* CSS styles can be applied to ANY HTML element.
* The easiest way is by using inline styles.
* We will use the style attribute, and again, all HTML tags support it.
* We want to make our News Briefs headline stand out by being Blue, and we want it in a sans-serif font.
* In the H1 tag, we’ll treat this like any other HTML property.
* Style, the = sign and the argument in quotes.
* In the argument, we can specify one or more CSS properties that will affect how the content is displayed. In this case we want to change the color and font.
* In the h1 tag, add style=”color:
* Note that we get a color picker. Let’s use that to choose a nice blue.
* Dreamweaver fills out the rest.
* Add a semicolon, which ends the first property, and type font:
* Dreamweaver or Komodo will give suggestions. Choose font-family and hit return
* Dreamweaver now shows lots of suggestions for font families. Choose arial black. <h1 style=”color:xxx; font-family: “arial…”
* Add a CLOSING QUOTE and Click on the Design Mode to see.
* Mark up the Edwards heading to be gray, and arial
* <h2 style="color:#666; font-family:Arial, Helvetica, sans-serif">
* Copy code and attach to the second H2 (Don’t do the third)
* Big part of this is learning all the properties. For now, just know PROPERTY, COLON, VALUE and SEMICOLON
* This is easy to do, but there are problems? What is the big problem?
* Yes, we had to put the same code on both subheads, and each paragraph would need its own code. This doesn’t make sense.

**EMBEDDED STYLES**

* A better solution is embedded styles, where we put all of the style information in one place. Think of it as an instruction sheet for browsers.
* Inside the style sheet, we can create as many rules as we need to properly display the content.
* The browser reads all the rules, and then uses them together to decide how to display the HTML content.
* Let’s move our inline styles into a style sheet

**MAKING A STYLE SHEET**

First thing we need is a style tag in our document’s heading. Right after the Title.

* <style type="text/css">
* Then hit two returns and close it. </style>
* This gives us a container to hold our styles

Let’s move the H1 heading. To do so, we simply type the tag selector H1, and then the style…

* h1 {
* color: #036;
* font-family:"Arial Black", Gadget, sans-serif;
* }

Now delete the code from the H1 on the page and see if it still works. It does.

* Let’s do the same with the H2 tag
* h2 {
* color:#666;
* font-family:Arial, Helvetica, sans-serif;
* }
* Remove the code from BOTH H2 tags.
* They BOTH still work!
* And better yet, the third one works too and we never touched it.
* So this is much better than embedded styles.
* But what about a web site with 5, 10 or 100 pages?
* We would have to have this code at the top of every page. Is that good?

**EXTERNAL STYLE SHEETS**

* Finally we are to the best way to use CSS – External style sheets
* In this case we will make a separate file with all of our CSS rules, and every page in the site can link to that one page – just like we link to an image. A single change to our style sheet affects all pages that are referring to it.
* Let’s move our embedded styles into an external style sheet.

**CREATING AN EXTERNAL STYLE SHEET**

* Because we already have embedded styles, we can just select and copy the existing rules. DO NOT copy the style tag.
* In Dreamweaver, Go to FILE, NEW, CSS and paste after the existing code.
* Save the document as styles.css in your Class 4 folder.
* This is now a standalone style sheet that we can refer to from multiple pages.

**LINKING TO AN EXTERNAL STYLE SHEET**

* Back in our document, delete the style tag and everything inside it.
* Click on the design view and note styles are gone.
* We need a LINK tag instead.
* <link type="text/css" rel="stylesheet" – This tells the browser it is loading a style sheet…
* and then the link… href="styles.css" />
* Click on the Design view and the style HOPEFULLY come back.
* Code: <link type="text/css" rel="stylesheet" href="styles.css" />

**TRY IT ON ANOTHER PAGE**

* Copy the style sheet code and close the page
* Open the page css2.html and paste the text
* The page is now styled! Isn’t that nice?

**HOW LONG DID THAT TAKE? \_\_10:15\_\_\_\_**

**GIVE THEM A BREAK – 10 MINUTES**

**CSS SELECTORS** (go back to css1.html)

* Now that we know how style sheets work, let’s look closer at CSS selectors. Remember selectors are what we use to attach properties to our content.
* Specifically selectors specify the part of our page we want to apply properties to. Like a stencil, as the movie mentioned.
* Let’s keep working with our current document. It is linked to a style sheet.

**TYPE SELECTORS**

* Type selectors are the simplest selector. They refer directly to an HTML tag
* Take a look at the CSS file we created, and we are already using type selectors. We have set styles for our h1 and h2 tags.
* *Note: CSS file is accessible as a tab at any time.*
* These rules will apply to all h1 and h2 elements in the document. If there aren’t any, the rule won’t do anything at all.
* Let’s set up one more. Let’s set our paragraphs (p tag) to verdana.
* In the CSS file, add this a new rule:
* p {
* font-family:Verdana, Geneva, sans-serif;
* }
* REMEMBER to SAVE the CSS file, and go back to the html file to see what happened.
* Note that ALL paragraphs are now Verdana. This would apply to every page linked to this style sheet.
* But what about our list? It is not in a p tag. But we can target it with the same rule format: p, li {
* Check it now and it is also verdana.

**ID SELECTORS**

* If we have a single element we want to style, we can use ID selectors.
* Take a look at our page in Code View and you’ll notice that the Copyright text is in a DIV tag. That’s for divider and it is a special tag that allows us to separate content into “boxes” so we can work with it.
* Note that on the Design view we see a dotted box to indicate that the copyright info is in a DIV.
* Click LIVE VIEW and you’ll see that the box doesn’t really show.
* We’ll talk more about DIV tags later, but for now, let’s take advantage of this to specifically target this text.
* First we need to give this tag an ID. This allows us to target it later.
* <div id="copyright">
* Where did copyright come from? I made it up. I can make my own names.

**STYLING AN ID TAG**

* Now that we have an ID, we can style it using CSS
* Go back to the CSS file and let’s style this ID.
* The format for an ID is # + ID with no space and then the rule
* Let’s do this…
* #copyright {
* font-size:9px;
* text-align:center;
* }
* SAVE and check the file.
* Only the copyright text changed. This is powerful because we can target a specific ID.
* HOWEVER, only one ID per page.
* So if we wanted to make a special rule that would be used in multiple places – say we want the intro paragraph of each brief to be bold – we could not use IDs for that. We are talking not about a specific paragraph, but about a type of paragraph. And there are likely to be many on a single page.

**THE CLASS ATTRIBUTE**

* The CLASS attribute can be added to multiple items, so it is our tool here.
* Go to the html page and let’s add this to both lead paragraphs…
* <p class="intro">
* We now have a way to refer directly to these to paragraphs, and any others that we want to attach the class attribute to.
* Next we need a rule. Go back to the CSS file and let’s add a rule for this…
* The format is almost the same as an ID, except that we use a . instead of a #
* .intro {
* font-weight:bold;
* }
* SAVE the CSS and go back to the html page to view.
* Any text with the class intro will now be bold.
* We can also apply to different types of elements. Add the intro class to our list. Style: <ol class="intro">
* Now it is bold too.

**SUMMARY**

These are the three basic selectors, element, ID and class.

(Show each in the file as we mention them again.)

**COMPOUND SELECTORS**

The three basic selectors are a start, but we can go even further. We can combine the basic selectors to make COMPOUND selectors.

* For instance, let’s say we want our intro paragraphs to be centered.
* Let’s add that in the CSS: text-align:center;
* Save and look. Good? Not on list.
* What we need is a rule that says to center intro ONLY if it is in a paragraph.
* We can do that. Make a new selector…
* p.intro and cut and move the text alignment into it.
* .intro {
* font-weight:bold;
* }
* p.intro {
* text-align:center;
* }
* SAVE and view. Only paragraphs are centered.
* The first rule says “make anything with the intro class bold.”
* The second rule says “if it is also in a paragrah, center it.”
* This allows us to be very specific based on the content we are styling.

**PSEUDO CLASS SELECTORS**

* By default hyperlinks are blue, and when you visit them they turn purple.
* Using pseudo class selectors, we can control this in our style sheet.
* a:link {
* color:#900;
* }
* The pseudo class is the :link, which tells which use of the a tag we want.
* Se we are saying “match all elements of type a when they are a link”
* This only applies to unvisited links. Open page in browser, click link and click back. Link is default of purple. We want to change this.
* So now we use the pseudo class a:visited
* Try and change our visited link to another color.
* a:visited {
* color:#F00;
* }
* Other pseudo elements are hover (mouse over), active (when clicked) and focus (when the element has keyboard focus.) Play with these and do cool stuff.

**EXERCISE:** Open css1.html, create a new stylesheet, make the headline blue, add an image