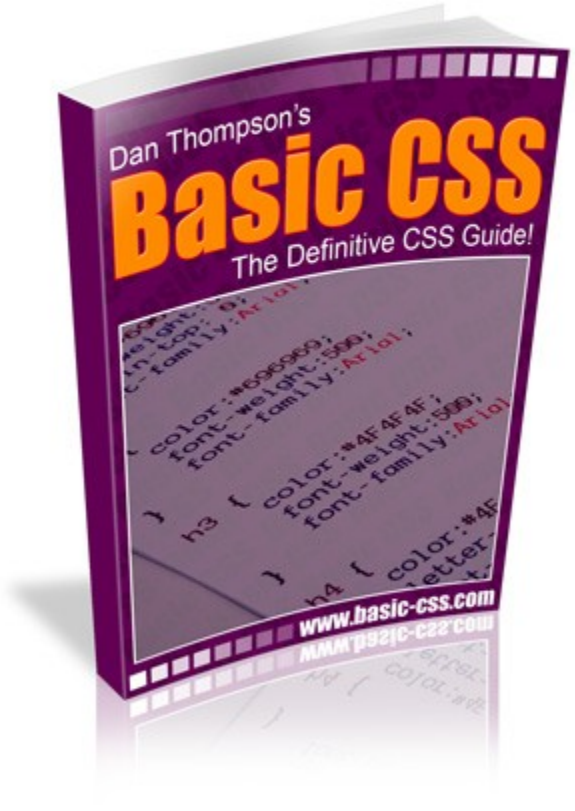
“Basic CSS”

- 1st Edition -



By Dan Thompson
www.basic-css.com

Copyright 2007

Resale Rights

This publication comes with full resale rights, meaning you can distribute the product as you wish, you can sell it, give it away or simply keep if for your own collection. The mini-site that you can use to resell the eBook is contained in the “Basic CSS” folder

Introduction

Hello and many thanks for downloading this eBook. Basic CSS is a follow up to my popular “Basic HTML” eBook, in this book I take a look at exactly how you can use CSS to spice up your website!

What Is CSS?

I'm glad you asked, there's no point me telling you all about CSS unless you know how it works, and how it can benefit your website!

CSS (Cascading Style Sheets), were brought about by the World Wide Web Consortium (W3C), and not by evil programmers with too much time on their hands as originally believed, to develop a standard which could be interpreted by different browsers. This became a necessity as Microsoft and Netscape battled for world domination and kept adding html attributes which were only understood by their browser. Great for them, bad for us.

Now, through the implementation of CSS, we can design ONE website for all browsers instead of multiple websites, one for each browser.

What started out as a way of standardizing the way code was displayed through basic things, like font and color tags, has quickly grown into something much more powerful, which makes sense because all of the not so evil programmers could now spend more time working on the one standard which could be understood by all browsers.

CSS can be utilized in your web pages a few different ways, lets have a look at them in the next section.

Using CSS On Your Website?

Inline CSS

The first way to use CSS on your website is known as **Inline CSS** and is used inside the html tag directly and only affects what is inside that element. This is a useful method when you are doing something like adding a Drop Cap or something which will only be used once or twice however this isn't ideal if you are working on a large site.

Here's an example of an inline CSS tag:

```
<p style="background: blue; color: white;">
Here's our nice white text on a blue background.
</p>
```

Don't worry if this isn't making sense, I will be going through everything in more detail later on.

Internal Style Sheet

The second way to utilize CSS on your site is the **Internal Style Sheet** which is placed in head of your page between the <head>and</head> tags. You might want to use this when you only have one page on your website or you need a page to look differently than the other pages which use your main CSS External Style sheet You could then use your Inline Style Sheets to overwrite anything on a line by line basis should you need to. Remember from above, the Inline CSS takes priority over the others.

Here is an example of an internal style sheet:

```
<style type="text/css">
.style1 {
        font-family: Tahoma, Verdana, Arial;
        font-size: 12px;
        color: #FFFFFF;
        }
</style>
```

External Style Sheet (My Favorite!)

The THIRD method is the **External Style Sheet**. This is a document that lives outside of your web page. By placing some code in the head of your web page between the <head>and</head> tags, you tell your web page to look for and use the instructions from that file to interpret how to display your page.

This method is useful when you have two or more pages and will allow you to control the look across all of your pages maintaining a consistent look throughout.

What a huge time saver!

Just imagine what will happen when the guru's and "powers that be", decide the **BIG, RED, BOLD HEADLINES** are no longer cool! With External Style Sheets, you can just edit your .css file (external style sheet), and get ready for a night out, although you may find yourself lonely because all of your web master friends will be working feverishly through the weekend making this change on each individual page!

To make this a little clearer, what would happen if Amazon wanted to change the color of the headings on each page of their website? If they used HTML formatting, Inline CSS or Internal CSS they would have to change each page individually, as you can imagine changing millions of pages by hand would be a soul destroying task. Using an external style sheet all it takes is one alteration to one file and the entire site design can be changed!

Here is an example of an external style sheet on a page:

```
<link  rel="Style sheet" type="text/css" href="css/main.css" />
```

The above statement simply tells the browser to pull the style sheet in from an external location.

Can I Mix The Three Together?

Good question! I'm glad you asked!

This is where the word “cascading” comes onto the scene. Let's say that you have an External Style Sheet which sets the look for your entire site.

On one page in particular you add some new elements and some inline css which differs from what you've outlined in your external style sheet.

Everything will “cascade” or fall into place. It will first use any inline code, then it will look for any internal code in your pages head and then it will use any external style sheet. If you've used inline code to change the color and font of one particular paragraph, the rest of the paragraphs and all other elements will still be defined by the external style sheet. They play together nicely, so to speak.

I will be going into a lot more technical detail later on in this book but I believe that the best way to learn something is by going out and getting your hands dirty! So I am going to throw you in at the deep end and lead you through the next section of the eBook – the hands on guide!

Hands on Tutorial

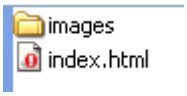
In this guide we will create our very own 5 page website using CSS, I have started by providing you with a basic, unformatted template which you can download by clicking here.

The template is very simple, the template has a header, content and footer section. We could use a CSS layout for the site but as I don't want to over complicate things I am going to use a simple HTML table layout with 1 column and 3 rows. Lets begin:

Step 1

Download the template by clicking here. Unzip the package and you will see the folder called "CSS Website"

You will then see the following file and folder:



If you double click on "index.html" to open it up, you will see that we have got the basic layout already setup, you will be able to see a table with 3 rows, the top and bottom row are currently blank and the middle row has some filler text in there, as well as some links at the bottom of the page.

The first thing I am going to do is insert the background graphic and set the body text for the document!

This isn't nearly as difficult as it sounds, I am going to use an external style sheet to insert the image and the formatting. Follow the steps below to create the style sheet:

1. Open up notepad or a similar text editing program (Start > Programs > Accessories > Notepad)

2. Enter the following text into the blank space, this will be a simple CSS statement that defines the the body background image and the body text style.

```
body {
        font-size: 14px;
        font-family: Verdana;
        background: url('images/bg.gif');

    }
```

Just a quick note, I will be giving all of the code a shaded out background as you can see above to make it easy for you locate!

Now that you have written out the above text (make sure you don't miss out any of the quotes or the code won't work!) let's have a look at what each line does.

The first thing you can see is:

```
body {
```

This is called the selector and tells the browser we are defining styles for the body of the document. Next:

```
font-size: 14px;
```

This is dead simple, we are telling the browser that by default all body text will be 14px in size.

```
font-family: Verdana;
```

Another simple one, this tells the browser that the default font for our document will be Verdana.

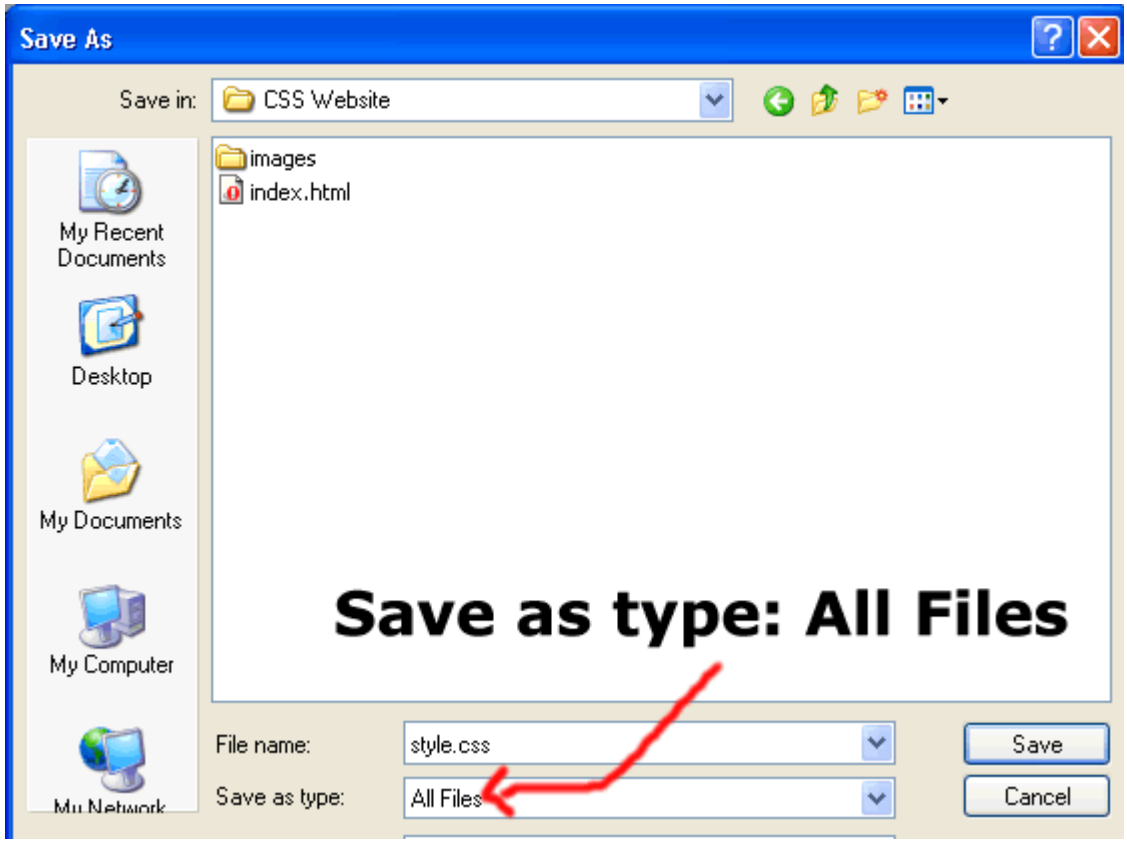
```
background: url('images/bg.gif');
```

Finally we are setting the background image for the page. The statement then closes with a }

So there we have it, your very first style sheet! The next step is to save the style sheet and insert it in our page.

Step 2

Now I want you to save the document you just created. In notepad go to File > Save As and save the document as “style.css” in the same folder as the index.html file



Once you have saved the file I want you to then open up the index.html file in notepad. To do this either go into notepad and select File > Open or right click on the file and choose Open With > Notepad

You will then see what looks like a lot of complicated code, don't worry – it isn't!

This is nice friendly HTML code but it is somewhat lost, it is sitting there looking all ugly and lonely, what it needs is a nice style sheet to make it more presentable – don't worry HTML code you SHALL go to the ball!

We are going to insert the style sheet we just created to format the HTML code for us. Before we do that let's just take a look at what the page looks like before we add the style sheet:

Big Catchy Heading Goes Here!

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem nonummy nibh euismod.
Sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat. Ut wisis enim ad minim veniam, quis nostrud exerci tution ullamcorper.
Ut wisis enim ad minim veniam, quis nostrud exerci tution ullamcorper. Sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem nonummy nibh euismod.
Sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat. Ut wisis enim ad minim veniam, quis nostrud exerci tution ullamcorper.
Ut wisis enim ad minim veniam, quis nostrud exerci tution ullamcorper. Sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat.

Lorem ipsum dolor sit amet, consectetuer adipiscing elit, sed diem nonummy nibh euismod.
Sed diem nonummy nibh euismod tincidunt ut lacreet dolore magna aliguam erat volutpat. Ut wisis enim ad minim veniam, quis

As you can see it all looks a bit dull, so lets now add our style sheet to the page and see if it makes any difference. Open up index.html in notepad and look for the following:

```
</head>
```

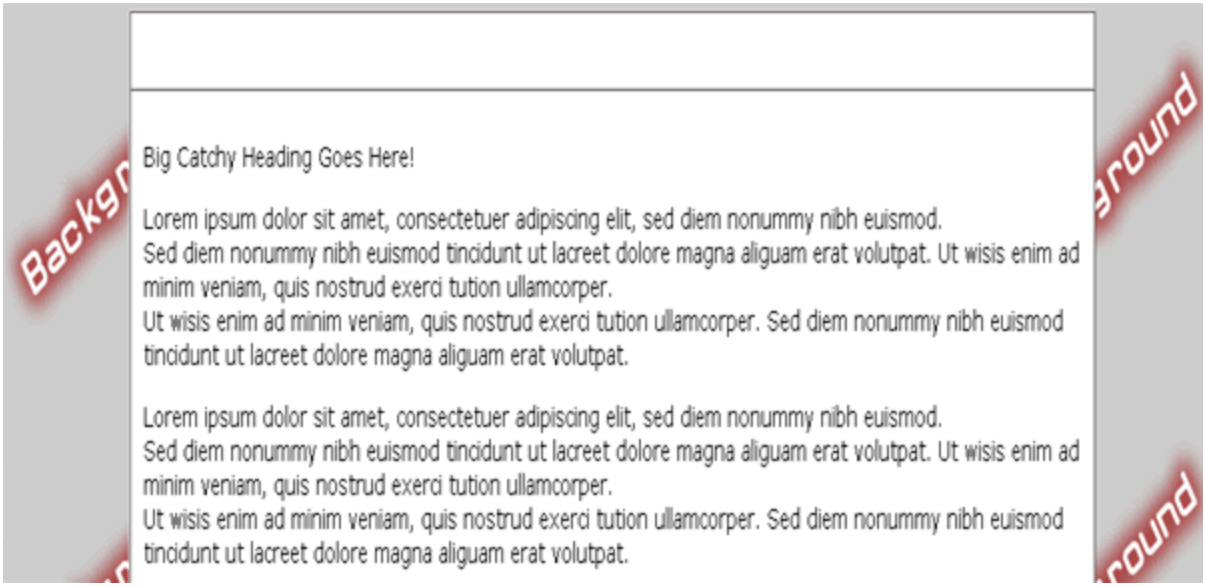
It will be about 5 or 6 lines down the page, then directly **ABOVE** that tag insert the following line:

```
<link href="style.css" rel="style sheet" type="text/css" />
```

A quick look at what we just inserted, you can see that the <link href= is telling the browser where the external style sheet is located, the rel= is telling the browser that the thing we are linking to is a style sheet, and the type= tells the browser that the document is text/css

I now want you to save the file (File > Save) and then double click on index.html to open it up.

You should then see something like this:



As you can see, the text has all been changed to “Verdana” as stated in our style sheet, the text is 14px in size and our background image has now been inserted.

The next step is to insert our header and footer into the document.

Step 3

I have already added default header and footer graphics that you can use, so all we need to do now is insert them into the page. If using HTML you would use a simple <img src= tag, but remember we are creating this site using CSS, not HTML!

So the first thing we need to do is define the header, and the footer in our style sheet So to do this, you guessed it! We need to open up style.css in notepad.

At present you can see we have only defined the body so far. So the next step is to define the header. The header will only be an image and will be displayed in the background of our top row of the table, we can also define the positioning, the border, and the size. As I don't want to over complicate things I am just going to define the image, and the size.

To do this we first of all give our header a name, again to keep it simple I will simply call it “header”:

```
.header {
            background:url('images/header.gif') no-repeat;
            width:780px;
           height:200px;
            }
```

Go ahead and insert that code below your body code, the style sheet should now look like this:

```
body {
          font-size: 14px;
          font-family: Verdana;
          background: url('images/bg.gif');

    }
.header {
            background:url('images/header.gif') no-repeat;
            width:780px;
           height:200px;
       }
```

If we break down the above code for the header you can see the first thing we did was the define the name for the style.

```
.header {
```

We then specified the background, this is the location of our header image, we also told the browser not to repeat the image.

```
background:url('images/header.gif') no-repeat;
```

We then defined the width, and the height of the style. I made this the same width and height of our header imagine so it would fit in the table perfectly.

```
width:780px;
height:200px;
```

Finally we closed the style as normal using:

```
}
```

The next step is to attach that style to our webpage. (make sure you have saved style.css) To do this:

1. Open up index.html in notepad
2. Directly under the <body> tag you will see the table is defined. We want to attach a “class” (or in English the header we just defined in the style sheet) to the first row of the table.

   Again, don't worry it isn't nearly as difficult as it sounds, look for the following text: (It's around line 12)

   <td bgcolor="#FFFFFF"> </td>

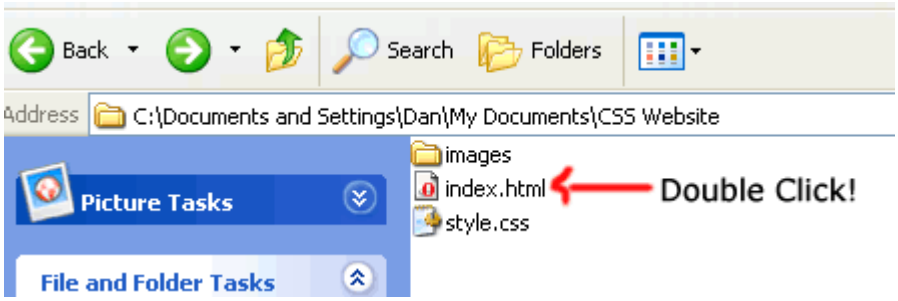
All we want to do is tell the browser the format that table row using the header style we defined in the style sheet To do this we simply add the text:

```
class = “header”
```

We add that text after the “#FFFFFF”, so the full line will look like this:

```
<td bgcolor="#FFFFFF" class="header"> </td>
```

Now I want you to save that index.html file (File > Save). The double click on it to open it in your web browser



You will then see that our header image has appeared in the top row of our table. If nothing shows up double check that your style sheet isn't missing any punctuation, I can't stress how important it is to include all of the punctuation as shown in my examples, if you miss a single ' then your code won't work!

The next step is to insert the footer, this is done in the exact same way as the header. So I am going to let you try and define that in style.css on your own, as a bit of a help the location of the footer image and the sizes are:

images/footer.gif
height: 40px
width: 780px

Now go and open up style.css and try to define a new style called “footer”.

Don't cheat by scrolling down to my example, have a go on your own and then compare it with mine!
.....
.....
.....
.....
.....
.....

Have you tried it? Yes, good you will go far my friend!

Have a look at mine and see if you have got the same:

```
.footer {
          background:url('images/footer.gif') no-repeat;
          width:780px;
        height:40px;
      }
```

You can see that it is set out in the exact same way as our header style. We just gave it a different name (footer) and changed the image location and the size. Your full style sheet should now look like this:

```
body {
        font-size: 14px;
        font-family: Verdana;
        background: url('images/bg.gif');

   }

.header {
          background:url('images/header.gif') no-repeat;
          width:780px;
        height:200px;
     }

.footer {
          background:url('images/footer.gif') no-repeat;
          width:780px;
        height:40px;
```

```
    }
```

We now need to tell the browser that the bottom of the table will be formatted using the footer class that we just defined in our style sheet

Do this in the same way as before, open up index.html in Notepad and scroll right to the bottom of the page and find:

```
<td bgcolor="#FFFFFF"> </td>
```

You can find the code 5 lines from the bottom of the page.

Can you remember what we do next? Thats right, we tell the row to be formatted in the style of our footer we just created in the style sheet Do this by adding the following code:

```
<td bgcolor="#FFFFFF" class =“footer”> </td>
```

When you have added the code, save the file and then double click on index.html to open it. You will then see the header graphic at the top and the footer graphic at the bottom.

Step 4

So now we have got our header, footer and background sorted out I think it is about time to start formatting our text.

The first thing I am going to do is make the “Big Catchy Headline” text stand out a little but more, so I am going to make it big, bold and red!

So to start with we need to define this new style for the headline in our style sheet, as always open up style.css in Notepad and start by declaring the name of the new style, lets call this style “headline”

```
.headline {
```

Next we will specify the properties, we will specify the font style, font weight, font size, font color and the alignment:

```
font-family: Tahoma;
font-size: 20px;
font-weight: bold;
text-align: center;
color: #990000;
}
```

See, that wasn't too bad was it? Simply add that code to the bottom of your style sheet and you have just created the headline style. You full style sheet should now look like this:

```
body {
        font-size: 14px;
        font-family: Verdana;
        background: url('images/bg.gif');

    }

.header {
          background:url('images/header.gif') no-repeat;
          width:780px;
        height:200px;
      }

.footer {
          background:url('images/footer.gif') no-repeat;
          width:780px;
        height:40px;
      }

.headline {
font-family: Tahoma;
font-size: 20px;
font-weight: bold;
text-align: center;
color: #990000;
                }
```

With the headline style now defined we need to open up index.html in notepad and tell the browser which text we want to apply that style too.

So open up index.html in notepad and find the following line: (it's about 16 lines from the top)

```
<p>Big Catchy Heading Goes Here! </p>
```

All we need to do is tell the <p> what class to display, or in English we need to tell the <p> tag to format everything inside of the tag with our headline style. This is dead simple to do, all we need to do is add the following after the first <p:

```
class = “headline”
```

That's it, the headline will now be formatted via our style sheet, just to confirm the full line will look like this:

```
<p class = “headline”>Big Catchy Heading Goes Here! </p>
```

Save the file and then close it, then double click on the index.html file to open it in your browser and view the results. You will see that the headline has been formatted in the style we set out in our style sheet!

Step 5

So far we have changed the heading, the background and the body text. Next I am going to format the rest of the text on the page using various different CSS properties, it won't necessarily create a good looking page but it will be a great way to show you what all of the different CSS properties look like when applied to a page!

The first thing we are going to do is create a new style that we can use for the first paragraph of text, you should be an expert at creating a new style by now, but just to go over it again very quickly:

1. Open up style.css in notepad
2. Create a new style called “para1”

You can try and do that on your own or you could cheat and use this code below:

```
.para1 {
```

After we have given the new style a name, we can then define the style properties. As I said, I am not really looking to create a masterpiece here but I am going to try and use as many different properties as possible, as it's only the first paragraph we wont go to wild with this one! Add the code below to your para1 { statement:

```
background-color:#FFFF00;
color:#000000;
border:dashed;
text-align:center;
                    }
```

This will make the text have a nice bright yellow background, the text will be black, the border will be dashed and the text will be aligned to the center of the page.

Save the style.css file and then open up index.html in notepad. We now need to add the style “para1” to our first paragraph.

With index.html open in notepad look for the following text:

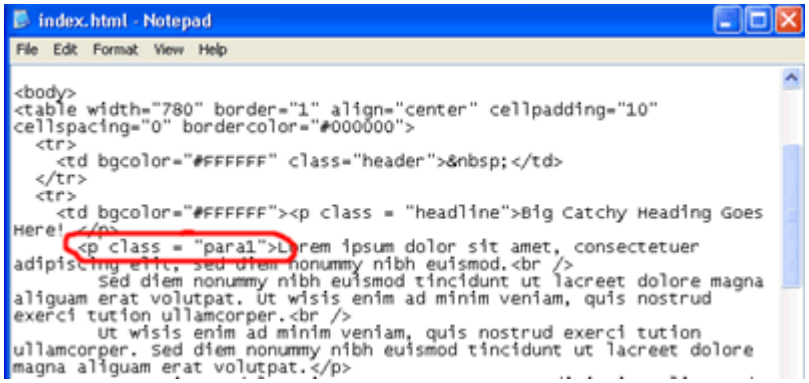
```
<p>Lorem ipsum dolor sit am..etc
```

It is directly underneath the headline text.

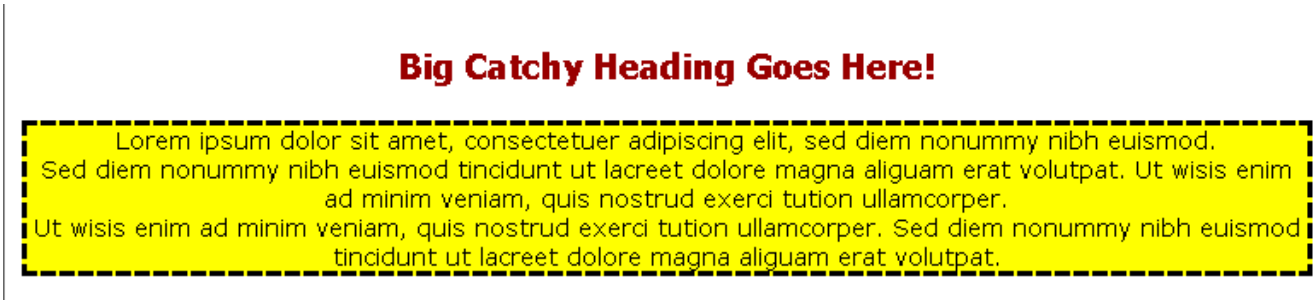
As before we just need to assign the style to the lonely looking <p> tag! This is dead easy, simply add “class = “para1” after the <p

So your index.html should now look like this:

```
<p class = “para1”>Lorem ipsum dolor sit am..etc
```



After you have edited index.html save it and then double click on the index.html file to open it up in your browser. You will notice that the first paragraph of text is now nice and yellow!



Next is paragraph 2, we do this in the exact same way as paragraph 1. So to start off we open up style.css in notepad and create the new style, we then define it's properties:

```
.para2 {
font-family:"Courier New", Courier, mono;
font-weight:bold;
text-indent:inherit;
color:#009900;
text-decoration: underline;
                }
```

We then add that style to the second paragraph. So we open up index.html in notepad, find the lonely <p> that is sitting there on it's own, waiting to be given a style! We then change that <p> to read:

```
<p class = “para2”>
```

And we then save the index.html file. Your page should now look something like this:

I am now going to let you have a go at formatting the remaining four paragraphs on your own. Here is what my style sheet looks like in full so far, I now want you to create styles for the remaining four paragraphs on your own using any properties that you wish, I will add a list of all properties available to you below the style sheet code:

```
body {
        font-size: 14px;
        font-family: Verdana;
        background: url('images/bg.gif');

   }

.header {
          background:url('images/header.gif') no-repeat;
          width:780px;
        height:200px;
      }

.footer {
          background:url('images/footer.gif') no-repeat;
          width:780px;
        height:40px;
      }
.headline {
```

```
font-family: Tahoma;
font-size: 20px;
font-weight: bold;
text-align: center;
color: #990000;
                }
.para1 {
background-color:#FFFF00;
color:#000000;
border:dashed;
text-align:center;
                }
.para2 {
font-family:"Courier New", Courier, mono;
font-weight:bold;
text-indent:inherit;
color:#009900;
text-decoration: underline;
                }
```

I have used a wide number of properties already (text-decoration, font-weight, border, text-align etc.) but there are still loads of properties that are yet to be used!

I am going to link to a full list of all properties available to you, feel free to use any of these properties when formatting your paragraphs! (The properties are listed towards the bottom of the pages)

List of Text Properties | List of Font Properties | List of Border Properties | List Of Background Properties

Once you have created the styles in your style sheet (style.css) you then need to add them to your page. Just like we have done previously you need to find the paragraph that you wish to apply the style to and simply add the name of your style after the <p, so for example:

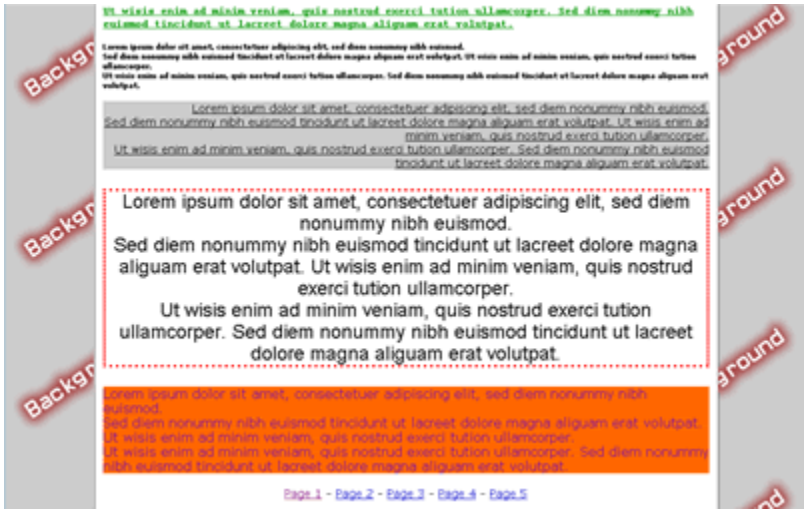
<p class = “para3”>

<p class = “para4”>

And so on...

OK, so have you now created the different styles and added them all to the document? If you have, congratulations! It wasn't so hard was it?

Whilst you have been beavering away creating the new styles, I have been doing the same thing, here's what my page turned out like, does your look any better? :-)



If you had any problems in creating your own styles, you can feel free to use mine below:

```
.para3 {
font-size:7px;
font-weight: bold;
                }
.para4 {
background-color:#CCCCCC;
text-decoration: overline;
text-decoration: underline;
text-align: right;
margin-left: inherit;
                }
.para5 {
text-indent:inherit;
font-family:Geneva, Arial, Helvetica, san-serif;
font-size:24px;
text-align:center;
border:dotted;
border-color: #FF0000
                }
.para6 {
font-stretch:extra-expanded;
font-size:16px;
color: #6600CC;
background-color:#FF6600;
                }
```

Step 6

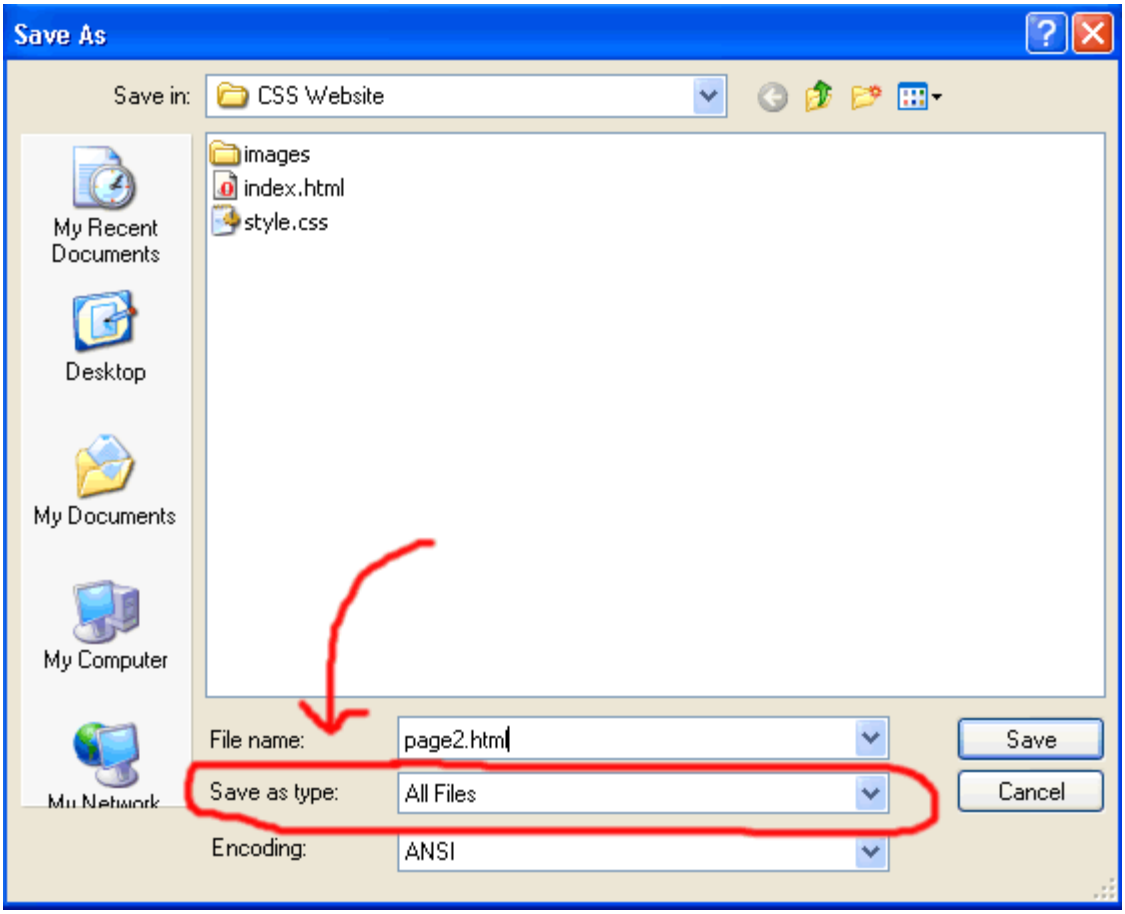
We are now on the home straight. You should now have a page with different paragraphs of text on there, each being “styled” by an external style sheet. Now at the moment the question you are probably asking is “Why can't we just use regular HTML, surely it would be quicker than this?”

Well in a way you are right, but you are about to discover just why CSS is such a powerful way to code websites.

Open up your index.html file in notepad and go to File > Save As

Save the page as “page2.html”

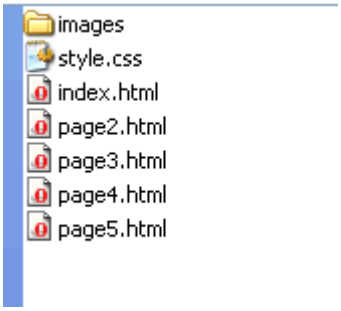
Make sure that you select “All Files” from the “Save as Type” box



I then want you to repeat this step (File > Save As) until you have created the following pages:

page3.html
page4.html
page5.html

You should now have a folder that contains these files:



Now double click on index.html to open it in your browser, if you scroll to the bottom of the page you will see some links that say “Page 1” “Page 2” and so on. Click on these pages and you will see they all look the same.

Now imagine this was a real life website and you decided that you didn't like a particular style of text, lets say for example you wanted to make the “Big Catchy Headline” a bit bigger!

If you used HTML you would have to open up each page individually and code in the new formatting, with CSS we simple do this:

1. Open up style.css in notepad
2. Find the .headline style and change the font-size value from 20 to 30
3. Click Save

Now open up one of your pages in your browser and you will see that your headline has been changed globally throughout your website! You can do this with any of your styles, so for example if you didn't like the formatting of the big yellow box with the black dotted border you

could just edit the “para1” style in your style sheet, click save and the formatting would be changed throughout your website!

That concludes our hands on example, if you would like to download a final version of the example to have a play around with you can do so by clicking here.

For the hands on example we used an external style sheet to format our page, but there are two other ways to use a style sheet These are more suited to smaller websites that don't have multiple pages but they can also be used in conjunction with the external style sheets, so you could try out some of the examples on the following pages on the site you just created.

I am now going to move onto the next section of this report, “Using Inline Style Sheets on your pages”

Using Inline Style Sheets

To use inline styles, you use the, ... (drum roll please).... style tag, otherwise known as the style attribute. Did you guess correctly? You can say yes, No one will ever know but you.

If you've ever taken a peek inside any html or source code, then you may have seen these little critters there because that's where they like to live. For the purpose of our lesson, we've captured one of these elusive style tags and placed it on the page right below here where you can get a good look at it in the light.

```
<p style=
```

There, it doesn't look so scary now does it? Kind of harmless really. The problem is right now, it doesn't know what to do, so it's just sitting there patiently awaiting instructions.

They are very obedient and follow orders extremely well as long as the instructions come in a clear voice which they can understand.

Let's have some fun with it and give it something to do.

```
<p style="color:red; font-family:verdana; margin-left:20px">
This is the text on the page that will be affected
</p>
```

So, what we've done is taken the old (paragraph) tag and added a style that will affect everything up until we close the paragraph with the tag.

In the example above, if you haven't already guessed, we used the style tag to change the color to red, use the verdana font and apply a left margin of 20px.

If we would have used sans serif font instead of verdana, then we would have placed that inside of single quotes like 'sans serif'. *Anytime you use two words **inline**, you must enclose in **single** quotes*.

Taking a closer look at the example above and see what makes it work.

We used the style= followed by a set of instructions. The first part of the instruction is known as the property and the second part is known as the value.
property:value

The first property we wanted to specify was the color and the first value was red. You separate the property and value with the colon => **:**

```
color:red
```

When using more than one property and value, separate those with the semi-colon => **;**

```
color:red; font-family:verdana
```

or

```
color:red; font-family:'sans serif'
```

Remember, two words needs to be enclosed in single quotes when being used inline.

Pretty simple, right? I know you are nodding your head enthusiastically. Tell me more, I can almost hear you saying.

What properties can I define for my fonts?

Excellent question!! I knew you were catching on!

The most common things you may like to define are as follows.

font-family
font-style
font-variant
font-weight
font-size
line-height
word-spacing
letter-spacing
text-decoration
text-transform
text-align
text-indent
vertical-align

That's cool, but... what do they do?

I'm glad you asked. Let's look at each one and see what is possible, see what values are available for each property.

We'll be sticking with the <p> (paragraph) tag to keep this simplified for the time being but you can also use style tags with other font tags and not just the paragraph tag. Like your <H1> (heading) tags for instance. We'll get back to this a little later though. Just wanted to mention it so I wouldn't freak you out later.

font-family

```
<p style="font-family:arial">
```

You can choose your favorite font for this. One thing to keep in mind with the property is that everyone may not have your favorite font installed on their machine. If no other font is defined as a back up, then the browser will use it's default font. You can choose an alternate font by placing more than one in the value section. You do this by separating each value with a comma.

```
<p style="font-family:arial, serif">
```

This tells the browser that if arial is not installed on the surfers machine, to use the generic serif font.

font-style

```
<p style="font-style:italic">
```

Choose from *normal, italic or oblique*, if you can tell the difference between italic and oblique please refer me to your optometrist so I can get a new prescription for my glasses.

font-variant

```
<p style="font-variant:small-caps">
```

Your choices here are “*normal*” and “*small-caps*”, which should have been called “all caps” instead because that's what it does. So, I can sense you're scratching your head right now and wondering, “why would I every need to use the “normal” tag?”. Hmm, this tells me that you're thinking in the right direction!

We'll jump ahead for a moment, not quite time travel, but a definite leap forward in this tutorial. If you were using an Internal or External Style Sheet and had your paragraph text set to small caps, you might want to jump out of small caps for a bit and have a small section of normal text. Ah ha!!

font-weight

```
<p style="font-weight:bold">
```

Choose from *normal, bold, bolder & lighter*. Alternatively, you could choose a number *100, 200, 300, 400, 500, 600, 700, 800 or 900* with 400 being equal to “normal” and “700” being equal to bold.

font-size

```
<p style="font-size:large">
```

xx-small, x-small, small, medium, large, x-large, xx-large
Medium is generally the browsers default size, so you may see at one size and someone else at another.

smaller, larger
These two when used, will adjust themselves according to the parent element. If the section you were working in was defined as small and you used “larger”, then the “larger” value would adjust your new area to medium.

Length
Use this to set an exact size by selecting a pixel size

```
<p style="font-size:18px">
```

%
Use this to set an exact size by selecting a percentage

```
<p style="font-size:75%">
```

line-height

```
<p style="line-height:120%">
```

normal

```
<p style="line-height:normal">
```

This is the default

number

```
<p style="line-height:1.4">
```

Number multiplied by the current font size to set distance between lines

length

```
<p style="line-height:16pt">
```

Use this set set a fixed distance between the lines

%

```
<p style="line-height:140%">
```

Sets the distance between lines based on the percentage of current line size so the 1.4 and 140% are essentially the same. Maybe the programmers do have a mean streak.

word-spacing

```
<p style="word-spacing:20px">
<p style="word-spacing:-0.4px">
```

This sets the distance between words. I can hear you saying, “Nooooo”, haha, Yes, it is! Choose between *normal*, which is default or set a *length*. It is possible to use a negative number here to crunch the words together or overlap.

letter-spacing

```
<p style="word-spacing:10px">
<p style="word-spacing:-0.4px">
```

I know, I know. Obvious huh? As with the word-spacing, you can also use a negative number to crunch the letters together, while leaving the distance between words the same, unless you specified that to be different.

text-decoration

```
<p style="text-decoration:line-through">
```

Guess what this one does. Bingo! It decorates the text. Choose from any of the options below, which I think are pretty self explanatory. If you were working in a section of underlined text, you could use the “none” tag to flip out of it for a bit.

none
Default decoration. Text has no special decorations like below.

underline
Creates a line underneath the text, like hyper linked text.

overline
The opposite of underlined. One of our evil programmers had dyslexia and nobody had the heart to tell him so they just left this one in.

line-through
Creates a line through the middle of the text. This is a mandatory decoration for Internet marketing because product prices, days left and copies available fluctuate on a minute to minute basis. This allows you to quickly cross out the ~~$29.99~~ and list it for $19.99, but only for ~~8 hours~~, ~~4 hours~~, 2 more hours but you better hurry 'cause there's only ~~12~~, ~~8~~, ~~5~~, ~~2~~, 1 copies left!

blink
Doesn't work in IE or Opera, so why even use it??

subliminal
Creates an unconscience reaction to the text on your page, forcing the surfer to follow your instructions without question. Ok, I made that one up, there's no such value! Don't we wish!!

text-transform

```
<p style="text-transform:capitalize">
```

Finally, one that is not so obvious! Here you can set all characters to all caps, 1st letter of each word caps, no caps, etc..

none
Default, will appear as it is typed. If you were working in an all caps section you could flip out of it for the specified text and it would resume when this tag was closed.

capitalize
Use this to capitalize the first letter of each word

uppercase
Set all letters to caps

lowercase
All letters are lowercase, no caps

text-align

```
<p style="text-align:center">
```

Determine how the text is to be aligned. You can select from *left, right, center* and *justify*. Left is the default and will be how it is displayed if this property is not set, depending on the surfers browser or any other specifications set in your internal or external style sheets.

text-indent

```
<p style="text-indent:15px">
<p style="text-indent:-15px">
```

Set either the length or percent. You can use a negative number here. If you have other text, table or pictures to the side, this can place the text on top of it.

vertical-align

```
<p style="vertical-align:super">
```

baseline
Places at the baseline. Common around pictures.

sub
Created subscript, which will place the character slightly lower than the surrounding characters as you would expect to see in H_2O which is a common example of subscript.

super
Creates superscript which is common in pricing, $19.^{99}$, this could be used in conjunction with the decoration:underline to create $19.\underline{^{99}}$

top
The top of this character will align with the top of the tallest element on the line. Generally used with images.

text-top
Aligns with the top of the parent element. Huh? You probably won't see much of a difference between top and text-top. Top, aligns with the tallest element on the line where text-top uses the parent elements font, so if you were working in a size 24px font, you could create some new text with a size 12px font and use text-top and it would magically rise above the base line and the top would be even with the size 24px text.

middle
Aligns in the middle of the element, common for pictures.

bottom
Aligns with the bottom of the lowest element in the line. Common with pictures.

text-bottom
Works like the text-top, only it's the bottom.

length
```
<p style="vertical-align:28px">
```

You can raise or lower by setting specific *px (pixels), pt, (points) inch (uh.. inches), mm (millimeters), cm (centimeters), pc (picas) or em (ems, will cover later)*.

%
```
<p style="vertical-align:120%">
```

Set by calculating a percentage of the line height value. May use negative number for this value.

Now that you have an idea about what can be done with fonts, let's go ahead and play around with it a bit.

Let me introduce to you the background element. Since we are still working on paragraphs and text, this won't be about the page background but rather placing our text on a background.

Using this code...

```
<p style="background: blue; color: white;">Here's our nice white text on a blue background.</p>
```

You could alternatively use an image

```
<p style="background: url(some_cool_image.gif);color: white;">Here's our nice white text on a cool background image</p>
```

...would place our white text on a blue background. If you like, you can take a quick break, stretch your legs and come back ready to get some hands on!

Ok, since you're reading this, I'll assume that you've either had your break and are ready to get to it or you're just so eager to get started that you skipped the break. Either way, congratulations on making it this far. Hopefully, you've come to see that CSS is understandable and that you shouldn't be afraid to dig into it.

When working with your actual pages or external .css files, you should always make a back up copy before making any changes. This way you can feel free to change one thing at a time then see what it changed, then just rinse, wash and repeat until you get it the way you like.

Having your page or .css file backed up is like working with a big ol' etch a sketch, if you mess up, just shake it and start over. I feel a little silly saying this but if you have a laptop and are about to shake it... STOP!!

Ok, why don't you go ahead and open up your favorite text editor and create a new blank page. Write out a paragraph or two, or copy and paste from somewhere else and then look at your html code.

Remember our <p style= critters from above? Of course you do! Go ahead and start adding and changing around some of the elements and values from above just to get a feel for it. Once you're comfortable with that, we'll move on to some other elements.

Hi, I take it that you've mastered the font-size and text-transform elements by now and are ready to tackle something a little heartier.

Good! Let's jump into Internal Style Sheets.

Internal Style Sheets

All the same elements and values apply as they did above, we will just list them differently. Instead of being directly “inline” in the html code, we'll place the css in the head of our document, in-between the <head>and</head> tags.

We start by placing the following opening and closing css tag in the head of our page

```
<style type="text/css">
</style>
```

All CSS code will be placed between the two tags.

In our Internal Style Sheet and our External Style Sheet, we will now use the following syntax.

selector {property: value}

The property and value tags are the same as with our inline, like when we set our property “color” to the value “red”. Our “selector” above was the <p> (paragraph). We actually were defining it when we use <p style=

The selector is basically our html tag we want the css to affect. Like now, I could leave the individual paragraph tags alone in the html and just add the following between my css open and close tags in the head

```
<style type="text/css">
p {color: red; }
</style>
```

Now anywhere I typed, as long as it was a normal paragraph, the text would be red. We can also set multiple values as we did before but since we aren't confined to being “inline” now, we can go ahead and enter after each semi-colon ; to help make things easier to read.

Where it was...

```
font-family:serif; font-size: medium; text-decoration: none;
```

It's now

```
font-family: serif;
font-size: medium;
text-decoration: none;
```

The only other thing we need to do here is tell it which selector to define and show where the property and value begin and end for the paragraph selector.

We do this by first naming our selector, followed the the opening tag, which is **{** we will then place our code and close with **}**

It doesn't matter if the {} are on the same line as above with our {color:red} or as follows.

```
<style type="text/css">

p {
        color: red;
        font-family: serif;
        font-size: medium;
        text-decoration: none;
}

</style>
```

Now, let's expand on that and add a background color. Since our background will affect the body of the page, we will want to use the “body” selector. If we were doing this without css, the body tag is where the background properties would be defined. Like this..

```
<body bgcolor="#000000">
```

Now, we can define in css like this.

```
body {
	background: #000000;
}
```

I've highlighted the selector in red, the property in green, the value in blue and bolded the opening and closing tags.

We can place as many css defines as we want. You've probably seen pages full of these if you've every had a look at a .css file before. You open it up, your jaw hits the desk and you quickly close the page not wanting to disturb the monster.

Next time you open one, you can browse on through it and recognize what the selectors are that are being controlled and what properties and values are being assigned to them.

Let's look at the two defines we've been working on to see how they look together.

```
<style type="text/css">

/* This is a comment  */

p {
	color: red;
	font-family: serif;
	font-size: medium;
	text-decoration: none;
}

body {
	background: #000000; /* This is another comment*/
}


</style>
```

You may have noticed a couple of comments in the code above. Comments can be on their own line or at the end of a line. These are your friends, sometimes your best friends when working with a long style sheet as often they will tell you what part of the web page the section or line of code affects.

Above we set the body to use Black as a background for our page. We can actually define many different things with our “body” selector.

For the following example, I'm actually going to grab the body define from one of my Wordpress themes.

```
body {
	background: url("background.png");
	font-family: 'Lucida Grande', 'Lucida Sans Unicode', Verdana, sans-serif;
	font-size: 14px;
	margin: 0;
	padding: 0;
	}
```

Let's stare this monster down!

We can see by looking, quite simply now that we are setting the body of the page to use a background image. Because no path is set for this image, it must be in the same folder as your web pages. We are telling it to use the Lucida Grande font and enclosing the two words in single quotes, (double quotes will confuse it), we are telling it that if the surfers browsers does not have Lucida Grande font installed to try the ones that following in that order and ending with a generic font.

Additionally, we want the page to be displayed with no margin and no padding and want to use a size 14px font as the page default. We can override this at anytime by creating a new selector which we will do later.

Did you get through that without pulling out any hair? I bet you breezed right through it didn't you?

The above was a custom template, I'm using. Let's take a look at the Wordpress default theme. We'll just look at the body selector define again.

```
body {
	font-size: 62.5%; /* Resets 1em to 10px */
	font-family: 'Lucida Grande', Verdana, Arial, Sans-Serif;
	background: #d5d6d7 url('images/kubrickbgcolor.jpg');
	color: #333;
	text-align: center;
	}
```

Here, we are using the font-size 62.5%, remember from earlier when we covered the font-size property, the percentage is based up line height. So the default font size will be 62.5% of whatever the line size is. So if we define different line heights in various places throughout our page and don't add a new define for the font size in those particular place, it will use the default 62.5%

Ok, I said earlier that we would go over what “em” is and since it's being used here, I guess this is as good a time as any.

When using px to set font sizes, line heights, etc.. you are setting an exact number for that font or line height, etc.. which means that how it is viewed is dependent upon the viewers screen resolution. A 12px font will look differently on a 600x800 screen that it does a 1024x780 screen.

The em setting is basically a scalable size and how it is displayed is calculated based on the font-size property of the parent element.

What the body selector did above is set the default text size to be about 10px. 1 normal em is about 12px. So with this base established, you can use “em” instead of px to allow for scaling. Which means that your page can look the way you designed it on any resolution screen.

Just remember that the “em” calculates from whatever the parent size is. To explain, let's say we've kept our 1 em at 12px because we didn't scale it down by setting the 62.5% like we did in the wordpress css body css above.

So if we were to use 1.2em to define our paragraph text size, this would be multiplied by our default 12px for a slightly larger font, however if we had set our paragraph to 62.55% and then were working within a paragraph and set some text to be 1.2em, it would calculate on our already reduced 62.55% for a figure slightly larger than the 10px but less than 12px.

There really was no simple way to explain that. I hope you didn't get thrown off there, the main thing to remember is that you can always back your style.css file up and make changes until you get the look you want. They key is to not be afraid of it.

The cool thing is that we can also use the em to scale our images, so the whole page is able to expand and contract as needed to fit in the surfers screen, looking exactly as we intended it to look.

Back to the default wordpress body CSS we were looking at.

Notice this time, there are two values for the background property. As you might have already guessed, the first is a color that will display until the image background loads.

background: #d5d6d7 url('images/kubrickbgcolor.jpg');
The color of the font is set to 333 which is almost black. You can use the 3 digit color codes or the full color code or even a color name like “whitesmoke”.

To find out which 3 digit color code is what, which can be difficult when going over your Wordpress Style Sheet, you can use this page which has the codes along with a visual of the color to make it easier on you to make the right changes to your style sheets.

Three Digit Color Codes

You can use this page for color names as well as Hex codes.

Color Names and Hex Codes

Let's take a look at defining our hyperlink colors. First let me start by saying, the selector for that will start with the letter a

If you think about it for a sec, it will make sense. Our hyperlinks start with

<a

Like our paragraph tag started with <p

We used the letter p for the selector to define paragraph css and we'll use the letter a to define anchor (linking) text.

```
a {
	color: #0000FF; /* links will be blue */
}

a img {
	border: #ccc 1px solid; /* linked images will have 1px border */
	padding: 4px; /* linked images will have 4px padding */
}

a:visited {
	color: #0000FF; /* visited links will be blue */
}

a:hover {
	color: #FF0000; /* links hovered over with mouse will be red */
}
```

I've commented the links above though I'm sure you already knew what each one was for.

Since we have several things behaving the same, there is no reason to place each in it's own css open and close tags. We can define more than one selector per line as long as we want the properties and values to be the same.

Let's do this again.

```
a, a:visited  {
	color: #0000FF; /* links and visited links will be blue */
}

a img {
	border: #ccc 1px solid; /* linked images will have 1px border */
	padding: 4px; /* linked images will have 4px padding */
}

a:hover {
	color: #FF0000; /* links hovered over with mouse will be red */
}
```

We can add as many selectors as we like, just separate each with a comma. When looking through a style.css file, often you will see many selectors on one line like this piece I copied from the Wordpress default theme.

```
h3 a, h3 a:hover, h3 a:visited {
		color: #0000FF;
}
```

This is defining that we want all h3 heading tags which are linked to be blue, also that the link should be blue when hovered over or visited.

What if we wanted the link to turn red when hovered over? We can just peel the h3 a:hover out of that line and place it on it's new line or add it into another line that already had the property and value that we wanted applied.

New line with h3 a:hover removed

```
h3 a, h3 a:visited {
		color: #0000FF;
}
```

New line defining h3 a:hover

```
h3 a:hover{
		color: #FF0000;
}
```

Now, any linked heading 3 will turn red when hovered over.

This gives you complete control over your properties and values. Just because a line has several selectors doesn't mean they have to stay there. Strip out the ones you want to have different properties and/or values and define them separately.

So far, we've been talking about basic html tags that are being defined, such as the paragraph, hyperlinks, headings, etc.. But what happens when we want <H1> headings or <p> paragraphs to appear differently in different sections of our page?

That's easy! We use CLASSES!!

That's right, we'll sit all of our little css monsters at little desks and line them up in neat rows and stand in front of the chalkboard. No, no, no. Not that kind of class.

Let's take a peek at our Wordpress default theme style.css file which you received with this ebook. I'm going to grab some css code from there and paste it below.

```
h2 {
	margin: 30px 0 0;
	}
```

```
h2.pagetitle {
	margin-top: 30px;
	text-align: center;
}
```

You see here the h2 selector with the margin set at 30px. Every time there is a <H2> on the page, this will affect it.

BUT... if we want our <H2> to appear differently someplace else, we can tell it that this is different than the normal <H2> tag and to apply different properties and values by defining it with a class! In your web pages html you would add this to your <H2> tag.

```
<H2 class="pagetitle">
```

The same would apply if we want paragraphs to look differently.

Also, remember that the style sheets are cascading, meaning the fall together. So if you were wanting to change the text alignment on your <h3> tag, you might find a line like the following in your style sheet...

```
h3 {
        margin: 30px 0 0;
        }
```

... and add the property and value there, like this...

```
h3 {
        margin: 30px 0 0;
        text-align: center;
        }
```

If that doesn't make the change on your page, then it is likely that the text-align property has already been defined for the h3 tag somewhere else. The quickest way to see is just user your “find” in what ever text editor you are using and search for h3. You should be able to scan through and see what properties and values are defined and where.

Remember, if the selector you want to change is already grouped together on a line with other selectors for the properties you want to alter, just peel it out and create a new entry or add the change to another existing entry.

Sometimes, you may want one of your selectors, like the paragraph to do something different like align right instead of left. This is easy to do by assigning a new paragraph selector like..

```
p.right {
        text-align: right
}
```

It could also look like

```
p.right {text-align: right}
```

Remember, the opening and closing tags can be on the same line, they are just usually broken apart to make reading easier, especially when you have multiple properties and values defined.

In your html, for the area you wanted to align right, you would just tell it to use those properties and values by calling the p.right class.

```
<p class="right">
```

That paragraph will now align to the right instead of whatever your p selector is defined as default.

The p. is placed in front of the right because it is still working off of the paragraph <p> in your html.

The same would apply to any sub settings you wanted to define for any of your other html elements like..

h1.sample
h2.
h3.
p.

Then just call that call in the associated html element by placing the class= tag and the portion that comes after the period. For our h1.sample, we would change our <h1> tag to this...

```
<H1 class="sample">
```

Sometimes you may see selectors like this...

.storycontent

... that aren't a normal html tag. That is perfectly fine. Using the class=, those properties and values are applied the same as the other was by using a DIV tag.

The DIV tag?

Stop! Don't panic, thinking you must have scrolled past the section where we discussed DIV tags because you didn't. There, you can relax and take a deep breath!

The div tag is used to define a section in your document. Keep in mind that a break
 will usually be inserted automatically after the closing div tag.

These will usually be used where no preset html tags exist such as the <p> or <h1> tags and will allow you even greater control of your page.

Going back to my custom Wordpress theme, there are several sections which use the .storycontent selector above. This is the section where the actual blog content is posted for this particular them and there are several properties and values assigned to this. Here is a little copy and paste from that themes style.css

Remember, this is just a part of it that we will stare down.

```
.storycontent,
.storycontent p {
		color: #333;
		font: normal normal 12px 'Trebuchet MS', Verdana, sans-serif;
		line-height: 130%;
		text-align: justify;
		margin-bottom: 10px;
		text-align: justify;
		}

.storycontent a {
		color: #333;
		border-bottom: #333 1px dotted;
		text-decoration: none;
		}

.storycontent a:hover,
.storycontent a:visited:hover {
		 border-bottom: 0px;
}
```

You are probably able to look at the above styles and understand what properties and values have been assigned to these selectors. If so, congratulations! You should pat yourself on the back for being such a good student!

If not, don't be discouraged. We've covered quite a bit and may take reading through a couple of times, actually practicing with this on your own test pages and viewing the style.css file of your blog if you have one before things start to fall into place for you.

In my custom themes index.php page, the following tag can be seen.

```
<div class="storycontent">
```

Everything between the opening and closing will be affected by the selectors starting with .storycontent in the above style.

For example, any hyperlinks falling between the opening and closing div tags above will be affected by this

storycontent a

This section...

```
.storycontent a:hover,
.storycontent a:visited:hover
```

... tells us that all links hovered over and visited withing those div tags will behave the same. Since css is cascading, and no link color has been defined, it will use the color from the...

storycontent a

which is color: #333

If we wanted the hover and visited links to have a different color, we would do what? That's right, just define it by adding that property and value to that section like...

```
.storycontent a:hover,
.storycontent a:visited:hover {
        border-bottom: 0px;
        color: #0000FF;
}
```

We could go a step further and have the hover and visited different by splitting them up into their own sections. We could then go on to make the hover link expand into all caps and increase size by 140% and change font families and other things by setting those properties and values.

In place of div tags, we can also use span tags. This comes in really handy if you just want to alter one character, a word or a line of text quickly without adding a break afterwards like the div tag does.

Here is an example of how you would use it.

```
<p>You are writing something here and it is using the default style set for the paragraph but you want to  <span style="color:#0000FF;">make a few words blue</span> so you overwrite the paragraph style temporarily with the span style</p>
```

We could also have changed the font size or numerous other things.

By now, hopefully some of the fog is lifting and you are able to start seeing your style sheets in a different way. Able to look through and tell what sections are affecting what.

Because it would be impossible to go over every possible combination without filling up phone books full of text, which I think there are some CSS books that do exactly that, I want to help put your mind in a place where you are able to look through a web page, find the selectors, class= and then search through your style sheet to find the section that is affecting it.

Because so many of us operate weblogs and with Wordpress being as popular as it is, I want to spend a minute going over it with you.

When you are editing your Wordpress style.css, the sections are pretty well commented to help you locate the selectors which most directly affect that section of your page.

Here are the section comments from the Default Wordpress Theme.

```
/* Begin Typography & Colors */
/* Begin Structure */
/* Begin Headers */
/* Begin Images */
/* Begin Lists */
/* Begin Form Elements */
/* Begin Comments*/
/* Begin Sidebar */
/* Begin Calendar */
/* Begin Various Tags & Classes */
```

When wanting to make changes to your template, first determine what section you are wanting to change. Is it the header, the way images are displayed, the sidebar, or basic colors? You can then scroll through your style.css file a little quicker when you know what section the style is most likely located in.

You do have to keep in mind how any changes you make may affect the rest of the page. If you have three columns and you increase the size of your right sidebar by 30px, you may want to decrease the size of your left sidebar or the main, content column by 30px to allow your page to fit on the screen the same.

Don't be afraid to experiment! Just save your existing style sheet under a different file name like style-BACKUP.css and if things get out of hand, you can put everything back to it's original condition quickly.

Now, I don't expect everyone that has gotten to this point to rush out and apply for jobs as css designers for the top Fortune 500 and software development companies but I do hope that you have a good concept of how things work and won't be afraid to get in there and get their hands dirty, make a few changes and see how things change what.

External Style Sheets

While what we were working on above referred a lot to the style.css, we need to cover one more thing before we move on to some fun stuff. We have already used external style sheets in our hands on example earlier but this is some theory that you will find useful.

The code for external style sheets and internal style sheets is the same. The only difference being that the internal style sheets are place in the <head> section of your document.

To use external style sheets, we will do this.

Look in the head of your document and cut everything between your css opening and closing tags, then delete the tags.

```
<style type="text/css">
Copy everything in here
</style>
```

Paste what you cut from there into a text document and save as style.css and upload to the directory you web pages will reside. Your style.css must be plain text with no other html tags, just the css code itself.

Then in the head of your document, place the following line of code which tells the browser that the information needed to render this page is located in the file style.css

```
<LINK href="style.css" rel="style sheet" type="text/css">
```

That's it. If you will be using this style sheet on pages within different directory levels, you can specify a complete url instead of just the style.css.

Make sure that the link href code above to your style sheet is in the head of every page you build or it won't know where to find it and won't display correctly.

Also, for testing purposes, well any purposes actually but especially for testing purposes, you can, instead of backing up your style.css file as we mentioned earlier, you can create a new file and name it something, anything like myteststyle.css and then just change in the code above to use that style sheet. If you don't like it, just switch it back.

If you feel so compelled, you can visit W3 Schools for even more information which hopefully after reading through this will make understanding and growing an easier process for you.

Fun Examples!

After all, what's the point of learning any of this if we don't use it. As web designers there are several things you could be doing immediately to make your sales pages and web pages come alive with pop and sizzle and we'll get into some of that now.

Add a shadow to your images.

This is a great technique for making testimonials/quotes stand out. You can use an extra div tag to do this.

```
<div align="center">
<div
style="width:115px;height:115px;filter:shadow(color:gray);"><img
src="http://www.yoursite.com/images/image1.jpg"
width="100" height="100"   style="border:1px solid black;">
</div>
</div>
```

For this effect, you want to use a width and height for your shadow filter which is slightly larger than your image. In the example above, we are using a picture that is 100x100 and the shadow filter is 15px larger.

If you image was 80x80, you could use a shadow that was 90x90 and adjust from there for more or less shadow. You can also adjust the strength and direction of the shadow by defining those withing the color property, like...

(color:gray,strength:18, direction:132)

How about some cool box borders?

Dashed Box

```
<div style="margin:0px auto;BORDER: black 1px dashed;PADDING: 15px;FONT-SIZE:
12px;WIDTH: 200px;BACKGROUND-COLOR: white;TEXT-ALIGN: center">
Place an image, text or both here
</div>
```

You can, of course, adjust the size of the box, text alignment, font size, font family and what ever else you like here.

To see some more really cool things you can do like using drop caps, creating magazine style layouts and plenty of other cool things you can do to spice up your sales pages, keep an eye on your email inbox for some more tips and tricks!

That concludes this tutorial!

I wish you the best and hope you will continue to work on your CSS skills as it is becoming more and more a norm in website creation and can allow you to put yourself out in front of others that haven't taken the time to learn.

Thanks Very Much For Reading,
Regards,

Dan Thompson

P.S. If you require any hosting for your website then I can highly recommend D9 Hosting. D9 Hosting was set up by myself and my business partner (Paula) in early 2007 with the aim to provide web designers and marketers with a reliable and affordable hosting service with top class technical support. We would be happy to have you on board and we offer a free site transfer for any new customers.

Please feel free to get in touch with us via the website if you would like any further information:

D9 Hosting (UK) | D9 Hosting (Rest Of World)