The first workshop dealt primarily with HTML which we used to provide content. This workshops addresses formatting our content using Cascading Style Sheets (CSS). We will use CSS to control the look and layout of text, images, hyperlinks, and the page itself.

Contents

ABOUT CSS ........................................................................................................................................... 3

.Class Styles ....................................................................................................................................... 3

Inline Styles ......................................................................................................................................... 3

#ID Selectors ...................................................................................................................................... 4

HTML Selectors .................................................................................................................................. 4

Descendant Selectors .............................................................................................................................. 5

Definition in an External Style Sheet (.CSS) .......................................................................................... 5

Importing the Workshop Files and Creating a Expressions Site ............................................................ 6

A. Getting the Workshop Files ............................................................................................................. 6

B. Creating a New Site ........................................................................................................................... 6

Step C: Importing the Workshop Files into your Expressions Site ......................................................... 6

Working with HTML Selector Styles ....................................................................................................... 7

Exercise 1: Floating All Images Left with an HTML Selector Style <img> ................................................. 7

Exercise 2: Affect Multiple HTML Selectors with One Style: h1, h2...................................................... 9

Inline Styles: CSS Defined and Applied within an HTML Tag.................................................................. 11

Exercise 3: Inline CSS Style to Float an Image Right ............................................................................ 11

Exercise 4: Inline CSS Style to Override another CSS Style Defined Earlier <h1> ............................ 12

.Class Styles ....................................................................................................................................... 13

Defining .Class Styles ............................................................................................................................ 13

Applying .Class Styles ............................................................................................................................ 13

Exercise 5: Creating a .Class Style (.noticeme) ................................................................................... 14

Exercise 6: .Class Style to Indent a Quote ........................................................................................... 15

#ID Styles ............................................................................................................................................ 17

WRAPPING TEXT AROUND A FLOATING <DIV> CONTAINER ................................................................ 18

Floating Multiple <div> Containers ....................................................................................................... 19

Exercise 6: Floated <div> Container & Descendant Selectors ................................................................. 20

Part A: Creating the Content .................................................................................................................. 21

Part B: Creating and Naming the <div> Container: id="motto" .............................................................. 22

Part C: Creating the #motto Style & Floating the Container .................................................................. 23
Part D: Descendant Selectors - “#motto img” Style to Center the Image ......................................................... 26
Part E: .Class Selector – Style to Bold & Center "Today's Quote" ........................................................................... 27
Part F: Creating & Stylizing the Sidebar Hyperlink: “View More Quotes” .......................................................... 28

Relative and Absolute Positioning .................................................................................................................. 30
Relative Positioning ........................................................................................................................................ 30
    Exercise 7: Relative Positioning <h2> Tags Lower .................................................................................. 31
Absolute Positioning ......................................................................................................................................... 32
    Absolute Positioning: Control Stack Order with Z-Index ............................................................................. 33
Summary: Floating and Positioning .................................................................................................................. 33
Combining Positioning with Floating ............................................................................................................... 34
    Exercise 8: Nesting Absolute Elements in a Relative Floated Container ...................................................... 37

TAKE HOME EXERCISE – USING MULTIPLE RELATIVE CONTAINERS ...................................................... 44
Breakdown of Different Items Used .................................................................................................................. 44
<div> Layout Sizing & Z-Index Scheme .............................................................................................................. 45
Overall Structure: Relative vs. Absolute Positioning ..................................................................................... 46
Contents of the “Header” div ......................................................................................................................... 46
Step A: Creating the Container Structure ...................................................................................................... 47
Step B: Create the #Header Style .................................................................................................................. 48
Step C: Create the #USC_Campus Style ......................................................................................................... 49
Step D: Create the #USC_Logo Style ............................................................................................................. 50
Step E: Create the #USC_Text Style ............................................................................................................... 51
Step F: Create the #USC_Motto Style ............................................................................................................ 53
Step G: Create the #Content Container Style ............................................................................................... 54
Step H: Create the #Page_Container Style .................................................................................................... 55
Style Definition Codes Used In Example ......................................................................................................... 56
Div Structure & Application of Styles Used in Example .................................................................................. 57
ABOUT CSS

The majority of web pages utilize HTML to provide the content and CSS to format it. There are several methods of applying CSS but with the exception of html selectors, all CSS must be applied by inserting its name or definition into an opening html tag. While almost any html tag will do, the most commonly used html tags CSS is coupled with are <p>, <div>, and <span>.

.Class Styles

.Class styles always begin with a period and are typically defined in the <head> section of a page (or on a style sheet) by giving them a name and properties. They can be applied as often as desired by inserting the style’s name into the opening HTML tag in the body section of the page. This allows the user to instantly update all occurrences of the style throughout the page(s) by simply changing the style’s definition up in the <head> section. The example below shows the definition and application of a .Class style.

In the example to the right, a class style we called “.NoticeMe” is created in the up in the head section of the document. As required, all styles created within the head section must be within the opening and closing <style> tags. The name of the style preceded with a period indicates it is a class style.

The .NoticeMe style makes text italic, bold, and cardinal. When you define .Class style in the head section, it must follow this syntax:

```
SelectorName {property1; property2; property3; etc.}
```

To apply a .Class style, you must place the following text in an opening html tag:  class="StyleName"

In this example, the name of the style is “NoticeMe” and we have inserted it within the opening <span> html tag. Note that period is not used in the name of the style when applying it.

Inline Styles

A style is said to be an “inline style” when it is both defined and applied from within an HTML tag. In the example below, the page’s heading “Mission Statement” is between open and closing <p> tags (Before). An inline style is then applied to the opening <p> tag to center the text and increase its size (After). Inline styles are typically used to override another style such as a .Class style.

Before:  

```
<p> Mission Statement </p>
```

After:  

```
<p style="text-align:center; font-size:xx-large"> Mission Statement </p>
```
### ID Selectors
Like *Class styles*, *ID styles* are defined in the `<head>` section and applied in the body section by coupling them with an HTML tag (typically `<div>`). Unlike *Class styles*, *ID styles* always begin with a # sign when defining them and can only be used once per page (although most browsers don’t enforce this). *ID styles* are most often used to control page layout by creating containers to hold and position content as shown in the example below.

```html
<head>
<style>
#QuoteColumn
{
    float:right;
    Width:100px;
    background:blue;
}
</style>
</head>

When people learn how we prepare each Mama’s recipe from scratch, they immediately understand something about why the products taste, well, homemade.

```html```
</div>
```

### HTML Selectors
When you use an HTML tag name as the name of the selector it is known as an “HTML Selector”. When you use this type of selector, you are redefining what the chosen HTML tag does everywhere it is used in the web page it is redefined in. For example, `<h1>` is an HTML tag that makes the text it is applied to large. If you use “h1” as the selector name in a style that makes text cardinal and centered, then any text in the body section using the `<h1>` tag will still be large but it will now also be cardinal and centered.

```html
<head>
<style>
h1
{
    text-align: center;
    color: #990033;
}
</style>
</head>

When people learn how we prepare each Mama’s recipe from scratch, they immediately understand something about why the products taste, well, homemade.

```html```
</div>
```

### Grouping HTML Selectors
You can also change multiple selectors at once by grouping them. For example, if we wish to make any text using the `<h1>` and `<h2>` tags red and italic, we would define our style by placing commas between the selectors involved:

```html
h1, h2 {font-style: italic; color: #990033;}
```
Descendant Selectors

Like HTML selectors, these are used to change the way an HTML tag functions but only when it is found within a specific “container” or embedded within other specified HTML tags. This technique is most often combined with #ID Selectors.

For example, the <h1> tag normally makes text large. We will redefine <h1> to also be italic and red but only when <h1> is found within an HTML tag we named “Content”.

| Definition in Head Section: | #Content h1 {font-style: italic; color: #990033;} |
| Application in the Body Section: | <span id="Content"><h1>Topics</h1></span> |

The word “Topics” will be h1, italic, and red because it is between h1 tags which are between HTML tags named “Content”. (We used the “span” tag but just about any HTML tag would work.) If you were to use <h1> elsewhere on the page, it would behave normally.

Getting More Explicit with Descendant Selectors

If desired, you can even be more specific by embedding more HTML tags in the definition path. Below is a variation on the example above. Here, h1 will make text large, red, and italic but only if it is within the tag named “content” and also within the <strong> tag. Any occurrences of the <h1> tag without this combination of both “content” and “strong” will behave normally.

| Definition in Head Section: | #Content strong h1 {font-style: italic; color: #990033;} |
| Application in the Body Section: | <span id="Content"><strong><h1>Topics</h1></strong></span> |

Definition in an External Style Sheet (.CSS)

CSS can also be defined in a separate file (ending in “.css”) and applied within web pages by specifying the path to where the .CSS file is located. Each web page that you wish to use the styles defined in the CSS file must reference the file in its head section. This is useful when you have a multiple page web that is to have a consistent look throughout. For example, should you wish to change the background color of all your pages, you just change the code in the external .CSS file and the backgrounds of all of the web pages linked to the external file will automatically change.

MyStyles.css

| Strong {font-style: italic; color: #990000;} |
| .CaptionStyle { font-variant: small-caps; color: #800000;} |

Page1.htm

| <head> <link rel="stylesheet" type="text/css" href="MyStyles.css" /> </head> |

Page2.htm

| <head> <link rel="stylesheet" type="text/css" href="MyStyles.css" /> </head> |
Importing the Workshop Files and Creating a Expressions Site

For this workshop we will create a new Expressions Web site and then import the text and images files we will be using.

A. Getting the Workshop Files
The files we need for this workshop are in a Zip file located on the Marshall documents web site.

1. Go to: www.marshall.usc.edu/computing/docs
2. Click “Microsoft Applications” from the left menu.
3. Scroll down and click the “EW2” zip file and save it to your computer.
4. Unzip the file (right click it and select “Extract All”)

B. Creating a New Site
1. From the menu, click “Site” – “New Site”.
2. Click the settings shown in the image to the right.

Choose your location by selecting “Browse” and then type the name of a folder you would like to create.

Step C: Importing the Workshop Files into your Expressions Site
In this step, we will import the files you unzipped in step A into the site you created in step B.

1. On your Folder List, click the EW2 folder.
2. From the menu, click “File” – “Import” – “File”.
3. Click the “Add File” button.
4. Locate the files you extracted.
5. Press Control + A to select them all.
6. Click “Open”.
7. Click “OK” to add them to your site.

They should now be in your folder list.

9. Drag all of the image files into the Images folder you created above. Note that because we made a “Site”, Expressions web will automatically update any links on our pages to the images to their new location.
Working with HTML Selector Styles

If you name your style the same name as an HTML tag, the style will change what the HTML tag normally does. For example, if you make “h1” green using a CSS style, then it will retain its original “h1” size but also be green. If the CSS style is in conflict with the HTML tag’s original properties, the CSS style wins.

This type of CSS is known as an “HTML Selector Style” and will immediately update all instances of the HTML tag on your page as soon as you define the style. HTML CSS styles are typically used to update older web pages that are using HTML formatting tags or to specify default settings.

Exercise 1: Floating All Images Left with an HTML Selector Style <img>

Images appear by using the HTML <img> tag. If we name our style “img”, we can affect all images on our page (or site) at once. In this example, we will float all images to the left making text wrap around the right side of the images.

1. Open "CSS_Overview.htm".
2. Make sure you are in “Split” view.

Note that this file is already using heading 1 and heading 2 and contains images.

3. Click in the Design section of the split window.
4. From the right side of your screen, click “Manage Styles” then “New Style”.

5. Name your style: img (Be sure to remove the period)
6. Click the “Layout” category.
7. Set “Float” to “Left”.

If you want to see the effect at this point, you can click “Apply” to see that your pictures are now on the left side of your page.
8. Click the “Box” category.
9. At “Margin”, uncheck “Same for All”.
10. Set the Top, Right, and Bottom margins to 9 pixels.

This will give us some space between our images and the text wrapping around them.

11. Click “OK”.

All the images should now be on the left and text should be floating around them. Note that some further adjustments might be necessary such as moving the images or resizing them. The CSS code Expressions created for us is up in the <head> section of your page between the <style> tags.

```html
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<style type="text/css">
    img {
        float: left;
        margin-top: 9px;
        margin-right: 9px;
        margin-bottom: 9px;
    }
</style>
</head>
```

Note also that there is no reference to any styles where the img tag is applied. This is because the style has changed how all HTML img tags work on our page.

```html
<img alt="Pasta" src="Pasta.jpg" width="129" height="112" />
```

**Editing a CSS Style**

You can edit your CSS style directly in code view or reopen the “Styles” menu and edit it there. To reopen the “Styles” menu:

a. Click in the Design view of your page. (i.e. below the split.)

b. Click the “Manage Styles” tab.

c. Right click your style and select “Modify Style”.

![Image of Modify Style dialog box](image.png)
Exercise 2: Affect Multiple HTML Selectors with One Style: h1, h2
You can modify multiple HTML tags at the same time by placing commas between them when you name your style. For example, rather than naming a style “h1” to affect only heading 1 tags, you can name it: h1, h2
This will affect both heading 1 and heading 2 HTML tags.

The file we used in Example 1 uses both h1 and h2. (“Mission Statement” is h1 and the rest of the titles are h2.) We will modify their font, size, weight, and color. We will also use relative positioning to move the titles closer to the paragraphs they are above.

1. If closed, open “CSS_Overview.htm”.
2. Click in the Design section of the split window.
3. From the right side of your screen, click “Manage Styles” then “New Style”.
4. Name your style: h1, h2 (Again, be sure to remove the period before the style name.)
5. Click the “Font” Category.
6. Make the settings shown.

If you would like to see the effects of what we have done so far, click the “Apply” button.

7. Click the “Position” category.
8. Set “Position” to “Relative”.
We just specified to make the heading appear 15 pixels lower from the top of where it would have normally appeared. (“Relative” positioning refers to a distance relative from where an object would have appeared if you hadn’t used a style.)
10. Click “OK”.

© Marshall School of Business Wayne Wilmeth EW4_WS2_PageLayout.docx 3/19/13 Page 9 of 57
The code created should look like the image shown to the right and your page, depending upon how you have sized your browser, should look similar to the image shown below.

Note that the coding of our h1 & h2 tags in the body is the same. Again, when you name a CSS style the same name as an HTML tag, all instances of the HTML tag are instantaneously affected throughout your document (or pages if you are using style sheets).

```css
h1, h2 {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 22px;
  font-weight: bold;
  color: #800000;
  position: relative;
  top: 15px;
}
```

<h1>Mission Statement</h1>
<h2>Our Kitchen</h2>

Mission Statement

Our Kitchen

When people learn how we prepare each Mama’s recipe from scrat, understand something about why the products taste, well, homemade. In our kitchen where the staff begins each day with the hand chopping fresh onions and carrots...bull vegetables, and proofing doughs for pizza crust, we know our tofu is made in the traditional Japanese way, may be large (400 gallons!), the techniques for the same as those used in cooking at home.

“Don’t walk behind me, I may not lead. Don’t walk in front of me, I Beside me and be my friend.”
— Albert Camus

Our Philosophy

The sauces perfected, the fillings just right, we begin the fun of the rest, is hand rolled, each pizza crust hand stretched and topped, and bro...pot pies. Hands sprinkle the enchiladas and just the right placed in each can of the N cooking for you in the same way. From choosing the baking pastry to perfection...
Inline Styles: CSS Defined and Applied within an HTML Tag

Most of the time, CSS styles are defined within the <head> section of your document or on a style sheet and then applied from an HTML tag on the page they are used; however, you can define and apply a CSS style from within an HTML tag. More often than not, this is done to override a style defined in the <head> section or on a style sheet. The syntax is:  

<opening html tag style="attribute:property; attribute:property; attribute:property">  

Be sure to:

- Separate multiple styles with semi-colons.
- Place a colon between the attribute and its property.
- Place all attributes and properties of the same style within a single set of quotes. (i.e. style="""")

Exercise 3: Inline CSS Style to Float an Image Right

In a previous exercise, we floated all of our images to the left by naming our CSS style “img”. However, we wish our “Pasta.jpg” picture to be floated right. To override the img style, we must define and apply our float right style directly within the img tag for the Pasta.jpg picture.

1. If closed, open the file “CSS_Overview.html”.
2. Make sure view is set to “Split”.
3. Locate the code that inserts the “Pasta.jpg” image (see below.)

```
<img alt="Pasta" src="Images/Pasta.jpg" width="238" height="213" />
```

4. Edit the code as follows inserting: style="float:right" between any of the spaces.

```
<img alt="Pasta" src="Images/Pasta.jpg" width="238" height="213" style="float:right"/>
```

5. If you save and then preview your page in your browser, the image of the pasta should now be on the right.
Exercise 4: Inline CSS Style to Override another CSS Style Defined Earlier <h1>

In a previous exercise, we made all occurrences of <h1> and <h2> obey a specific style. In this exercise, we would like our Mission Statement <h1> to also be centered, underlined, and larger. We will use inline styles to override the style in the <head> section of our page.

1. If closed, open the file “CSS_Overview.html”.
2. Make sure view is set to “Split”.
3. Locate the code that has <h1> tags around the text “Mission Statement”.
4. Within the opening <h1> tag, edit the code to look as follows:

```html
<h1 style="text-align:center;text-decoration:underline;font-size:xx-large">Mission Statement</h1>
```

5. Click below the split in the Design window.
   Your Mission Statement text should have updated to resemble the image to the right.
Class styles allow you to create independent styles that you can apply to almost anywhere in your page as many times as desired. They are defined either up in the <head> section of each page or on a style sheet. They are applied by inserting the name of the style within almost any HTML tag.

### Defining .Class Styles

- When defined in the <head> section, all styles you are creating must be within the opening and closing <style> tags.
- Class style names always begin with a period when defining them.
- Definitions are enclosed in French braces.

In the example to the right, we have created a Class style called “.noticeme” that makes text bold and italic.

```html
<style type="text/css">
    .noticeme
        {font-weight: bold;
         font-style: italic;}
</style>
```

The basic syntax is:

```
.stylename {attribute:property; attribute:property; attribute:property;}
```

### Applying .Class Styles

.Class style can be applied from almost any opening HTML tag but <span> is often used because <span> has very few properties of its own. It merely specifies a beginning and ending point. The basic syntax for applying a class style is:

```
<opening html tag class="stylename">
```

Note that when applied, the period in front of the style name is not included.

```html
<title>
<h2>Our Kitchen</h2>
<p>When people learn how we prepare each Mama’s recipe from scratch</p>, they immediately understand something about why the products taste Mma’s is just a large kitchen where the staff begins each day
```

When applying styles in Expression’s <i>Design</i> view:

- Clicking but not highlighting text and then applying a class style typically applies the style by inserting it into the opening <p> tag around the text and affects the entire paragraph.
- Highlighting text typically creates the <span> tag around the text you highlighted and affects only the highlighted text.
Exercise 5: Creating a .Class Style (.noticeme)

In this exercise, we will create the class style “.noticeme” described on the previous page. It will make any text we decide to apply it to bold and italic.

Defining the Style
1. If closed, open the file “CSS_Overview.html”.
2. Make sure view is set to “Split” and the cursor is below the split in the Design area.
3. Click the “Manage Styles” tab.
4. Click “New Style”.
5. In “Selector”, name the style: .noticeme
6. Click the “Font” category.
7. Set “font-weight” to “bold”.
8. Set “font-style” to “italic”.
9. Click “OK”.

Applying the Style
A class style can be applied from within almost any html tag but <span> is often used because it specifies a beginning and ending point has not attributes of its own. In Expressions Web, to apply a .Class style using span tags:

1. In Design view, highlight some text.
2. Click the “Apply Styles” tab.
3. Click the style you wish to apply (.noticeme in this exercise).

If you look at the word(s) you applied the style to in code view, you will see the Expressions inserted the <span> </span> tags around the word(s) you highlighted and within the opening span tag, the “noticeme” class is applied using: class=“noticeme”

The syntax for applying a class style from within an HTML tag is:

<opening html tag class=“Name of style”> content you wish to affect </closing html tag>

Notice that a period is not used in the name of the class style when applying as it is when defining it.
**Exercise 6: .Class Style to Indent a Quote**

In this exercise, we wish to create a .Class style that will indent a paragraph on both sides, place a line above and below it, and change the font color and emphasis to italic.

**Defining the Style**

1. If closed, open the file “CSS_Overview.html”.
2. Make sure view is set to “Split” and the cursor is below the split in the Design area.
3. Click the “Manage Styles” tab.
4. Click “New Style”.
5. Name the style: .Quotes
6. Click the “Font” Category.
7. Make the font settings shown in the image to the right.
8. Click the “Block” category.
9. Set “text-align” to “center”.

This will center our text. Note that if you set a width in the position category, text-align-center won’t center.
10. Click the “Border” category.
11. Make the settings shown in the image to the right. They will place a 3 pixel green border above and below our text.

12. Click the “Box” category.
13. Make the settings shown in the image to the right.

*Padding* is an internal margin between the quote text and the line above and below the quote text. *Margin* is an external margin between the sides of the quote text and the page as well as the distance between the top and bottom lines and the non-quoted text.

14. Click “OK”.

**Applying the Style**

We could simply highlight our quoted text and click the name of our new style to apply it. Expressions would place `<span>` tags around our text and apply the style; however, being that our quote already has paragraph tags around it (i.e. `<p>` & `</p>`) AND we wish to apply our style to the entire paragraph, all we really need to do is click anywhere within the paragraph and click the name of our style.

Expressions will apply the code using the opening `<p>` tag rather than creating additional `<span>` tags. This is a little cleaner than also using the `<span>` tags within the `<p>` tags.

1. In Design view, click anywhere within the text we are formatting as a quote but don’t highlight.
2. Click the “Apply Styles” tab.
3. Click the “Quotes” style.

The paragraph you clicked should now be indented as shown at the beginning of this exercise. Note that the code has the style applied from the opening `<p>` tag:
#ID Styles

**#ID Selectors** always begin with a pound “#” when defining them and like Class Selectors, they are defined within the head section or a style sheet and are applied from within an opening HTML tag. Unlike Class Selectors, **ID Selectors** can be used only once per page (although many browsers don’t enforce this). **ID Selectors** are most often used when the page has been sectioned off with <div> tags to create “containers”. For example, I wish to give a container called “LeftColumn” a tan background color.

Definition of an ID Selector named “LeftColumn” in the <head> area:

```html
<style>
  #LeftColumn {background-color:#FFEDB7}
</style>
```

Applying an ID style in the body section from within an opening HTML tag (<div> in this example). Note that the # sign is not used when applying the style as it is when defining it.

```html
<div id="LeftColumn"> page content  </div>
```

Anything between that opening (<div>) and closing (</div>) tags will have a tan background color. We introduce this now because the next few exercises use ID styles with div tags.
WRAPPING TEXT AROUND A FLOATING <DIV> CONTAINER

Just as you can float an image left or right and wrap text around it, you can also wrap text around content placed in a <div> container. A <div> container is just a set of div tags that you have placed content within and have applied a style to float it and set its width or height. (A div tag is an HTML tag with very view properties of its own.)

For example, the <div> container below contains text and uses an inline style to make the container 100px wide, float right, and have a blue background. A mock-up of the style is shown in the image to the right.

```
<div style="float:right; background:blue; width:100px">
  A mock-up of the style is shown in the image to the right.
</div>
```

Browser Window Resized

One advantage of floating div containers is that if the end user resizes their browser window, the container will move back into the window to be seen and the text around it will adjust its wrapping. On the downside, your page design may no longer look as you intended.

Note that a <div> container can also contain images, hyperlinks, lines, as well as more div containers.
Floating Multiple <div> Containers
When placing <div> containers side-by-side, you must be very careful that there is enough room across the page to fit their width dimensions or else one will end up below the other. The illustrations below cover how multiple containers behave when they are floated right.

Using Multiple Floated Containers
The image above shows two floated <div> containers. Both are set to 180 pixels wide and floated right.

Multiple Floated containers - Browser Window Resized
The image above shown to the right, the end user has resized the browser window to make it narrower. Because there is no longer enough room to display both containers side-by-side, they now appear vertically. Again, the advantage is that the end user can still read them but the disadvantage is that content may not be appearing in the layout you intended.

TIP
If you have multiple <div> containers that have the same properties, then define them using a single .Class style rather than multiple an ID Styles.
Exercise 6: Floated <div> Container & Descendant Selectors

In this example, we wish to create the sidebar shown in the image below. This is simply a <div> container floated to the right. We will use HTML selectors to format the <h3> tags but only when they are found within the #motto styles. This is known as a "descendant" selector.

Our Motto

Our Kitchen  <h3>

When people learn how we prepare each Mama's recipe from scratch, they immediately understand something about why the products taste, well, homemade. Mama's is just a large kitchen where the staff begins each day with the most basic of tasks...hand chopping fresh onions and carrots...building soup stock from fresh vegetables and proofing doughs for pizza crusts and pot pies.

Our Philosophy  <h3>

The sauces perfected, the fillings just right, we begin the fun of the real handwork. Each burrito is hand rolled, each pizza crust hand stretched and topped, and broccoli is hand placed in the pot once the kids sprinkle cheese and enchiladas are trussed like birds. All items are placed in each order Minestroni...

#motto

We will name the <div> "motto" and the style "#motto". This style will control the background and text color within the container as well as make it float to the right so text will wrap around it.

#motto img

(descendant selector)
This style will center any image found within the "motto" container.

#motto h3

(descendant selector)
This style will change how text using the <h3> tag will behave when found within the "motto" container.

#motto a

(descendant selector)
This style will change how our hyperlinks look but only when found within the "motto" container.

#motto a:hover

(descendant selector)
This style will change how hyperlinks look when the mouse is hovered over them but only when found within the "motto" container.
Part A: Creating the Content

In this section we will start a new HTML page and copy text and an image into it.

2. Click "Save" and name it "Quotes.htm".
3. In Code view, title the page "Mama's Quotes".

4. Right click & drag "Today's Quotes.txt" from the "Images" folder into the Design window.
5. Select "Insert File".
6. Select "Normal paragraphs with line breaks".
7. Click "OK".

The text should now be in the body area of your page.

8. Drag the "Mama.jpg" file from the "Images" folder into the Design window between "Our Kitchen" and "Today's Quote".
9. Give the image an Alternative name of "Mama" and click "OK".
10. Click "Save".
Part B: Creating and Naming the <div> Container: id="motto"

Any content that is to be within the right floating sidebar must be between opening and closing <div> tags.

1. Within Code view, create the following code between "Our Kitchen" and the image of Mama:

```html
<div id="motto"></div>
```

You must now cut and paste the closing </div> tag to the bottom of our container.

2. In Code view, highlight the closing </div> tag.

3. Press Control + X to cut it.
4. In Code view, click just below "View More Quotes" and press Control + V to paste.

Your code should now look similar to the image to the right. Note that the line numbers are not important and do not have to match what you have.

```
<body>
<p>Our Kitchen</p>
<div id="motto"></div>
<p><img alt="Mama" height="87" src="Images/Mama.jpg" /></p>
<p>Today's Quote</p>
Don't walk in front of me, I may not follow.
Don't walk behind me, I may not lead.
Just walk beside me and be my friend.

--Albert Camus

View More Quotes</div>
```

```
<p>When people learn how we prepare each Mama's recipes, they understand something about why the products are homemade. Mama's is just a large kitchen where the house is built. It's so much like a church. Our Kitchen</p>
<div id="motto"></div>
```

```
<p>Don't walk in front of me, I may not follow.<br />
Don't walk behind me, I may not lead.<br />
Just walk beside me and be my friend.<p>
</p>
<p>--Albert Camus</p>
<p>View More Quotes</p>
</p>
```

```
<p>When people learn how we prepare each Mama's recipes, they understand something about why the products are homemade. Mama's is just a large kitchen where the house is built. It's so much like a church. Our Kitchen</p>
```

```
</div>
```
Part C: Creating the #motto Style & Floating the Container

We will now create the #motto style.

1. Click "Manage Styles" then "New Style:"
2. In Selector, type: #motto
   (ID styles start with # and are case sensitive.)

Set Background Color
3. Click the "Background" category.
4. Set "Background-color" to dark blue.
5. Click "Apply" to see what it does.

Set Font Color
6. Click the "Font" category.
7. Set "Color" to white.
8. Click "Apply" to see the effect.
Set Width
9. Click the "Position" category.
10. Set "Width" to 180 pixels. (This makes our container 180 pixels wide.)
11. Click "Apply" to see what it does.

Float Right
12. Click the "Layout" category.
13. Set "Float" to "right".
14. Click "Apply" to see the effect.
Set Margin Around the Container and Padding Within
Margin controls the space around the outside of the container and padding controls the spacing between items within the container and the edge of the container.

15. Click the "Box" category.

16. Set "Padding" to 5 pixels. (Same for all.)

17. At "Margin", uncheck "Same for all".
18. Set the top, bottom, and left Margin to 8 pixels.
19. Click "OK" to see the effect finish editing the style.

Our Kitchen
When people learn how we prepare each Mami’s recipe from scratch, they immediately understand something about why the products taste, well, homemade. Mama’s is just a large kitchen where the staff begins each day with the most basic of tasks...hand chopping fresh onions and carrots...building soup stock from fresh vegetables, and proofing doughs for pizza crusts and pot pies.

And yes, even our tofu is made in the traditional Japanese method. While our pots may be large (400 gallons!), the techniques for preparing food are much the same as those used in cooking at home.

Our Philosophy
The goat's perfect, the fillings just right, we begin the fun of the...
Part D: Descendant Selectors - “#motto img” Style to Center the Image

We will use this style to center our image within the <div> container horizontally. We will accomplish this by giving it 30 pixels of margin to its left to slide it over towards the middle. The style uses img as a descendant selector within the #motto div. This means that only images image found within the #motto container will obey this style.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In “Selector”, type: #motto img
4. Click the “Box” Category.
5. Uncheck “Same for all” next to “Margin”.
6. Set “left” to 30 pixels.
7. Click “OK”.

![Modify Style](image)

Today’s Quote
Don’t walk in front of me, I may not follow.
Don’t walk behind me, I may not lead.
Just walk beside me and be my friend.

— Albert Camus

View More Quotes
Part E: .Class Selector – Style to Bold & Center "Today's Quote"
In this section, we will make a style class style that can be applied anywhere. Note that you must highlight the text you wish to apply it to first and then apply the style. It will set text to bold and center it as well.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In "Selector", type: .SidebarTitles
4. Click the "Font" category.
5. Set Font-family to "Ariel".
6. Set Font-weight to "Bold".
7. Click the "Block" category.
8. Set text-align to "Center".
9. Click "OK" to save the style.

“Today’s Quotes” - Applying .Class Styles
Normally, to apply a Class style in Expressions, you highlight your text and then click the name of the style from the “Apply Styles” tab. Expressions then places <span></span> tags around your text in invoking the style. Unfortunately, <span> can be problematic and may not work with “text-align: center”. Therefore, we will enclose “Today’s Quotes” within <p></p> tags and invoke the style from within the opening <p> tag as so: <p class="SidebarTitles">

10. Create the code shown below.
Part F: Creating & Stylizing the Sidebar Hyperlink: “View More Quotes”

Creating the Hyperlink
1. Highlight the text “View More Quotes”.
2. From the menu click “Insert – Hyperlink”.
3. Click “Existing File or Web Page” then “Current Folder”.
4. In Address, type: http://www.quoteland.com
6. Click “OK”.

Creating the “#motto a Style to Change the Color of our Hyperlink
In this section we will change the color of our hyperlinks to yellow by altering the anchor <a> tag but only when it is found within the #motto container.

7. Click the “Manage Styles” tab.
8. Click “New Style”.
9. At “Selector”, type: #motto a
10. Click the “Font” Category.
11. Set “font-style” to “Italic”.
12. Specify a light “color”
13. Set “text-decoration” to “none”.
   (This will remove the automatic underlining of hyperlinks.)
14. Click “OK”.

--Albert Camus
View More Quotes
Creating the “#motto a: hover” Style to Change the Hyperlink Color when Hovered
In this section we will change the color of our hyperlinks to pink when the mouse is hovered over the hyperlink. We will also save ourselves some time by copying and then editing the “#motto a” style created above.

15. Click the “Manage Styles” tab.
16. Right click the “Motto a” style and select “New Style Copy”.

A copy of the #motto a” style will open. We will now edit the copy.

17. Change the Selector to read: #motto a:hover

18. Click the “Font” Category.
20. Click “OK”.

![Image of the CSS editor with #motto a:hover selected and the color set to pink.](image-url)
Relative and Absolute Positioning

Another method of laying out a web page is to use relatively and absolute positioning combined with floating `<div>` containers. There are two types of positioning: `Relative and Absolute`

- **Relative Positioning** – Moves the element relative from where it would have normally appeared on the page.
- **Absolute Positioning** – Places the element a specific distance from the top left corner of the page.

Relative Positioning

Allows you to position an element (i.e. paragraph, image, `<div>` container, heading, etc.), on the page relative from where it would have normally appeared to the edge of the page. You can set its distance from the top, bottom, left or right.

For example, if you wish to move an object 50 pixels to the `right` of where it would normally appear, you would set a relative position of `left` of 50 pixels. In other words, you are adding 50 pixels more between the object’s original position and the left side of the page.

In the example below, “Our Philosophy” is within the `<h1>` tags and relatively positioned as described.

![Relative Positioning Diagram]

- **Relative Bottom**: 50px
- **Relative Right**: 50px
- **Relative Top**: 50px
- **Relative Left**: 50px

In summary, relative positioning:

- Allows you to specify the position of an object using `left`, `right`, `top`, & `bottom`.
- The new position is relative to where the object would have normally appeared.
- It has no effect on the normal document flow. Other objects behave as if it were still in its original position.
Exercise 7: Relative Positioning <h2> Tags Lower
In this example, we will move our paragraph titles down 20pts so they will be closer to the paragraphs they describe. All of our paragraph titles are heading 2 so we will made an HTML selector style called “h2”.

1. Open the file “Relative_Positioning.html”.
2. Click in the Design section of the split.
3. Click the “Manage Styles” tab.
4. Click “New Style”.
5. Name your style: h2
6. Click the “Position” category.
7. Set “Position” to “relative”.
8. Set the “Top” position to 20 pixels.
9. Click “OK”.

<table>
<thead>
<tr>
<th>Modify Style</th>
</tr>
</thead>
<tbody>
<tr>
<td>Selector: h2</td>
</tr>
<tr>
<td>Define in: Current page</td>
</tr>
<tr>
<td></td>
</tr>
<tr>
<td>Category:</td>
</tr>
<tr>
<td>Font:</td>
</tr>
<tr>
<td>Block:</td>
</tr>
<tr>
<td>Background:</td>
</tr>
<tr>
<td>Border:</td>
</tr>
<tr>
<td>Border Radius:</td>
</tr>
<tr>
<td>Layout:</td>
</tr>
<tr>
<td>List:</td>
</tr>
<tr>
<td>Table:</td>
</tr>
</tbody>
</table>

The titles should now be closer to the paragraphs they are describing.

(TIP: If you want less space between the paragraphs, try setting the Top Margin to a negative number such as -20).
Absolute Positioning

Absolute positioning allows you to place objects including <div> containers exactly where you want by positioning them from the upper or lower left corner of the page. The advantage of absolute positioning is that you have much more control over where your page elements are placed and you can overlap elements. The disadvantage is that non-contained elements on your page will run right through your absolutely positioned elements.

Absolutely Positioned Single Element, All Else Free Flowing
The example to the right shows an image that was absolutely positioned to appear 300 pixels from the left and 100 pixels from the top of the page. Because the text on the page is free floating (i.e. not in a container), it will run right through the image. Note that floating elements has no effect when interacting with absolutely positioned elements. When you float an element, it will move out of the normal document flow to get out of the way but when you absolutely position an element, it is taken out of the document flow. (In other words, absolutely positioned elements are invisible to the other elements in your document.)

All Elements Absolutely Positioned
When absolutely positioning an element on your page, it is common to also absolutely position other elements on your page. In the example to the right, both pictures have been absolutely positioned and text has been placed in div containers that that have been sized and also absolutely positioned. By controlling the size and position of the elements, you can prevent them from overlapping.

Disadvantage of Absolutely Position All Elements – Browser Resizing
On the downside, because all elements are absolutely positioned, when the user resizes their browser, the elements will not rearrange themselves in an attempt to stay visible as floated elements can.
Absolute Positioning: Control Stack Order with Z-Index

Absolute positioning allows you to specify the stack order when you have objects that overlap. To place one object above another in the stack, simply give it a larger Z-index number. In the example to the right the images are both absolutely positioned. The image on top has a Z-index of 20 and the image underneath has a Z-index of 10. Note that the actual numbers you use have no other meaning aside from the fact that objects with larger numbers will be on top of objects with smaller numbers. For example, we could have used 2 and 1; however, it is always a good idea to move up in large increments. This will prevent you from having to renumber all of your existing objects should you wish to insert a new object in the middle of the stack.

Summary: Floating and Positioning

Positioning and floating both have their benefits and drawbacks:

Floating Advantages
- Allows you to place an object on the left or right side of the page and other non-positioned objects will adjust to move around it if the browser is resized.
- You can float most objects including <div> containers. Further, you can specify the container’s width and height.

Floating Disadvantages
- Only allows you to place objects on the left or right side of the page.
- Your layout may not be exactly how you want it should the user resize their browser.
- You cannot control overlap of objects that were floated.

Relative Positioning Advantages
- You can set the size of <div> containers and move them relative to where they would have normally appeared in the document.
- You can center an object on the screen by setting its Left and Right margins to “Auto”.
- You can absolutely position objects from within the walls of a relatively positioned <div> container.
- You can float a relatively positioned object.

Relative Positioning Disadvantages
- If you set the position of a relatively positioned object (top, bottom, left or right), other objects will behave as if the object is still in its original position. This can cause text to run through an object.
- You cannot specify the stack order (z-index) of relatively positioned objects.

Absolute Positioning Advantages
- Allows you to position any element anywhere you like on the page or from within a relatively positioned <div> container.
- Allows you to control the stack order of overlapped objects.

Relative Positioning Disadvantages
- Absolute positioning removes the object from the normal document flow. This can cause unwanted overlap with elements whose position is not being controlled.
Combining Positioning with Floating

The best aspects of a floated <div> container is that other objects can wrap around it and should the user resize their browser, the containers will attempt to move to stay in the window. The best aspects of absolutely positioned objects is that you can place the object anywhere you like, overlap them, and control the stack order. In this section, we will look at how to gain both of these advantages. This comes down to one key point:

If you place absolutely positioned objects within a relatively positioned div container, the absolutely positioned objects’ placement will be from within them walls of the <div> container, not the page.

Further, because the relatively positioned <div> was not floated, we can set the left and right margins to “Auto” to center the relatively positioned <div> container on our page. Should the end user adjust their browser width, the relative container will slide left and right to remain centered taking is contents with it.

Note that you can still float a relatively positioned <div> tag so that text wraps around and you can set its height and width as well. You should however avoid setting its position if you intend to float it.
Relative Floated <div> Contains Absolute Objects
In the example to the right, a floated and relatively positioned outer <div> contains two absolutely positioned pictures and an absolutely positioned <div>. Because the absolutely positioned objects are within a relatively positioned <div>, their position is set from the upper left corner of the outer <div>, not the page. And because the outer <div> tag is floated, any non-constrained text will flow around it. Note that as long as you don’t set a relatively positioned <div> container’s position (i.e. Top, Bottom, Left, Right), text will wrap around it properly if you float it. Setting the width and height will not upsetting the wrapping.

Browser Resized
Because the outer <div> is floated, when the user resizes their browser, any non-positioned text around the <div> will move out of the way to make room for the floating <div>. The objects within the outer <div> will move also but their position to each other within the outer <div> remains locked. As with all absolutely positioned objects, if the user makes their browser even smaller, the absolutely positioned objects will get cut off.
Multiple Floated, Relative <div>'s
In this example, we have floated two relative <div> containers. Because their combined widths are less than the current width of the browser, they fit side-by-side. Note also that each of them contains an absolutely positioned picture and an absolutely positioned <div> tag.

Browser Resized
Should the user resize their browser window making it too narrow to display the floated <div> containers side-by-side, they will appear vertically. (Again, this is because they are floated and that is what floated objects do.) Note that the absolutely positioned objects move too when their containers move but stay locked in the same position within their container.
Exercise 8: Nesting Absolute Elements in a Relative Floated Container

In this exercise, we will create a floated, relatively positioned <div> container that contains three absolutely positioned containers.

When you place absolute div’s in a relative div, the dimensions are from within the div rather than the page.

```html
#USCContainer {
    background-color: #FFFFCC;
    border: thin solid #800000;
    padding: 10px;
    margin: 10px;
    width: 300px;
    height: 300px;
    position: relative;
    float: left;}

#LibraryPic {
    position: absolute;
    top: 30px;
    right: 30px;
    z-index: 10;}

#CampusPic {
    position: absolute;
    top: 110px;
    right: 110px;
    z-index: 20;}

#USCText {
    position: absolute;
    top: 250px;
    left: 20px;
    font-family: Arial, Helvetica, sans-serif;
    font-size: 20px;
    color: #800000;
    border-top-style: solid;
    border-bottom-style: solid;
    padding-top: 4px;
    padding-bottom: 4px;}
```
1. Open the file “Absolute_USC.html”.

**Creating the Container Structure**
2. In code view, locate the general area where you want your floated <div> container to appear and create the code structure shown below.

Be sure to nest the absolutely positioned <div> containers within the relatively positioned <div> container (“USCContainer”).

The inner <div> containers (“LibraryPic”, “CampusPic”, and “USCText”) can be in any order within the USCContainer <div> tags.

Using tabs has no effect on the outcome but makes following the structure easier.

**Creating ID Style to Manage the #USCContainer**
3. Click below the split in Design view.
4. Click the “Manage Styles” tab.
5. Click “New Style”.
6. Name the style: #USCContainer
   *(This style is for our USCContainer <div> which is an ID style so we must precede its style name with a # sign.)*
7. Click the “Background” Category.
8. Select a “background-color”. *(This will color in the background of our container.)*
9. Click the “Position” Category.
10. Set “position” to “relative”. (A <div> container that contains absolutely positioned <div> containers must be relatively positioned.)
11. Set the “width” and “height” of the container to 300 pixels.

12. Click the “Border” Category.
13. Make the border settings shown

14. Click the “Box” Category.
15. Set the “padding” and “margin” for all sides to 10 pixels. (Margin is the space around the outside border of the container and padding is the space between the border and the container’s contents.)

16. Click the “Layout” Category.
17. Set “Float” to “Left”. (Floating left will place the container on the left side of the page and allow any free floating objects outside of our container to wrap to its right. Note that floating works with relatively positioned objects or objects that have not been positioned but will not work with absolutely positioned objects. Further, if you specify a position such as top, bottom, left, or right, wrapping may not occur as you wish.)

18. Click “OK”.

At this point, your page should look like the picture shown to the right.
Placing Content within the Absolutely Positioned Containers

At this point, we could either define the styles for our three absolute containers then place content in them or vice versa. For learning purposes, we will place content within our absolute container first and set the styles later.

In this example, placing content in our containers is easier in code view.

19. Between the USCText <div> tags, type: University of Southern California as shown.
20. Drag the Library.jpg picture off of your folder list and between the LibraryPic <div> tags as shown.
21. Drag the USCCampus_Sm.jpg picture between the CampusPic <div> tags.

If you click below the split in Design view, your page should look similar to the image to the right. Note that you may have to save the page and preview it in your browser for it to render correctly. Also, saving, closing and reopening the page within Expressions Web sometimes helps with the rendering.
Creating the Style for the LibraryPic Container
This section will absolutely position the LibraryPic container to be 30 pixels down from the top of the USCContainer and 30 pixels right from the left side of the USCContainer that contains it. We will also give it a Z-index of 10.

22. Click below the split in the Design window.
23. The style name must match the name of our LibraryPic <div> container and it must start with a # sign because it is an ID style. In selector type: #LibraryPic
24. Click the “Position” Category.
25. Make the settings shown.
26. Click “OK”.

Your page should now look similar to the image to the right. Note that you may have to save the page and preview it in your browser for it to render correctly. Also, saving, closing and reopening the page within Expressions Web sometimes helps with the rendering.
Creating the Style for the CampusPic Container
This section will absolutely position the CampusPic container to be 110 pixels down from the top of the USCCContainer and 110 pixels right from the left side of the USCCContainer that contains it. We will also give it a Z-index of 20 so it will be on top of the LibraryPic container. (When absolutely positioned objects overlap, objects with a larger z-index number are higher in the stack.)

27. Click below the split in the Design window.
28. The style name must match the name of our CampusPic <div> container and it must start with a # sign because it is an ID style. In selector type: #CampusPic
29. Click the “Position” Category.
30. Make the settings shown.
31. Click “OK”.

Your page should now look similar to the image to the right. Note that you may have to save the page and preview it in your browser for it to render correctly. Also, saving, closing and reopening the page within Expressions Web sometimes helps with the rendering.
Creating the Style for the USCText Container
This section will absolutely position the USCText container to be 250 pixels down from the top of the USCContainer and 20 pixels right from the left side of the USCContainer that contains it. Because it won't be overlapping any other objects, specifying a Z-index is not necessary.

32. Click below the split in the Design window.
33. The style name must match the name of our USCText <div> container and it must start with a # sign because it is an ID style. In selector type: #USCText
34. Click the “Font” Category.
35. Make the settings shown.

36. Click the “Border” Category.
37. Make the settings shown.

38. Click the “Position” Category.
39. Make the settings shown.
40. Click “OK”.

That’s it! Your page should now look like the example on the starting page of this exercise.
TAKE HOME EXERCISE – USING MULTIPLE RELATIVE CONTAINERS

In this example, we will overlap images and text to create the web page shown to the right. To accomplish this, each element (i.e. the text, images, background) have all been placed within their own absolutely positioned <div> containers. Overlapping is controlled by setting each container’s z-index. (Highest z-index is one top.) To the right is a snapshot of the finished page and below is a breakdown which shows each div container and its content.

Breakdown of Different Items Used
- USC_Logo_Black.jpg
- USCCampus.jpg
- About_USC_Text.txt

All this div contains an image of the campus. It is absolutely positioned.

All this div contains a maroon background with black top & bottom borders. It is relatively positioned.

University of Southern California
This div contains text and top & bottom borders. It is absolutely positioned from the upper left corner of the “header” div (Purple).

This div contains an image of the USC Logo. The div is absolutely positioned from the upper left corner of the “Header” div (purple).

This div contains the USC Motto. The div is absolutely positioned from the upper left corner of the “Header” div (purple).

Established: 1880
Los Angeles was little more than a frontier town in 1880 when USC first opened its doors to...
<div> Layout Sizing & Z-Index Scheme

This div contains the text “University of Southern California”. It is absolutely positioned at 10px top and 15px left. It has a z-index of 60 so it will appear on top of everything else.

This div contains the USC motto. It is absolutely positioned at 120px top & 15px left. It has a z-index of 50.

This div contains the image of USC. It is absolutely positioned 0 top & 0 left. (The header’s border pushes it down 20px.) Its z-index is 30. Its height is 218px to make it fit precisely in the header div vertically.

This div contains all of the divs which make up the header. It is relatively positioned and contains the maroon background and black top and bottom borders. All of the divs within it are absolutely positioned within it from its upper left corner. The borders are 20px each and occur outside of the div’s dimensions.

This div contains the page content. It is relatively positioned and will appear below the header because it is after the header in the code. It has a z-index of 10. Note that its width is set to 560px rather than 580px because we have added 10px of padding all around. Padding exists outside of the div’s dimensions and background color extends into the padding area.

We wanted to center our page content within the browser window so we enclosed all items on the page within this div. Centering is accomplished by setting the left and right margins to “auto”.

© Marshall School of Business Wayne Wilmeth EW4_WS2_PageLayout.docx 3/19/13 Page 45 of 57
Overall Structure: Relative vs. Absolute Positioning

Relative Positioning
Both the “Header” and “Content” divs are relatively positioned within the all encompassing “Page_Container” div. Because their position is relative, they will not overlap. This forces them to appear one above the other as their sequence in the code dictates.

Note that if you wanted them to be side-by-side, you would use “Float” in conjunction with relatively positioning and if there were room, they would appear side-by-side.

Absolute Positioning
All of the div containers within the “Header” div are absolutely positioned. Their position is from the upper left corner of the “Header” container by specifying a top and left dimension.

It is absolute positioning that allows containers to overlap and it is their z-index which dictates which is on top of which.

Note that the dimensions start inside of the black top border. (i.e. the border extends out of the div’s dimensions.)
Step A: Creating the Container Structure

It is vital that you nest your <div> tags correctly for this to work. Follow the steps in this section to create the page’s <div> structure.

1. Create a new page (File - New – HTML) and save it as “USC_Layout.htm”.
2. Click the “Split” button.
3. Click within “Code” view.
   We will create “Page_Container” and its sub containers of: “Header” and “Content”.
4. Create the div containers shown below.

These are the opening and closing div tags for “Page_Container”.

Both the “Header” and “Content” div containers are nested within the “Page_Container div.

The “Header” container contains more div containers so we will now nest those.

5. Insert the “USC_Motto”, “USC_Text”, “USC_Logo” & “USC_Campus” div containers within the “header” container as shown.

All of these div containers are between the opening and closing div tags of the “Header” container.
Step B: Create the #Header Style

In this section, we will create the style for our “Header” container. It has a maroon background, black top and bottom borders, and is 580px wide by 218px high with a z-index of 20. It is also relatively positioned.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In “Selector”, type: #Header
4. Click the “Background” Category.
5. Select a maroon background.

6. Click the “Border” Category.
7. Uncheck “Same for All” at border-style.
8. Set “border-width” to 20px.
9. Set “border-color” to black.

10. Click the “Position” Category.
11. Set “width” to 580px.
12. Set “height” to 218px.
13. Set “position” to “relative”.
15. Click “OK”.

That’s it for the #Header style. Its code is shown below.

```css
<style type="text/css">
  #Header {
    background-color: #800000;
    border-top-style: solid;
    border-bottom-style: solid;
    border-width: 20px;
    border-color: #000000;
    width: 580px;
    height: 218px;
    z-index: 20;
    position: relative;
  }
</style>
```
Step C: Create the #USC_Campus Style

In this section we will drag the “USCCampus.jpg” file into our #USC_Campus div container and then define the style to position the image to fit precisely between the two bars on top of the #Header container.

1. Drag the file “USCCampus.jpg” off of the file list into the “div#USC_Campus” container located in the Design window.

2. Click the “Manage Styles” tab.
3. Click “New Style”.
4. In “Selector”, type: #USC_Campus
5. Click the “Position” Category.
6. Set “position” to “absolute”.
7. Set “z-index” to 30.
8. Set “height” to: 218 px.
9. Set “top” to 0 px.
10. Set “left” to 0 px.
11. Click “OK”.

That’s it for this style. The code is shown below and the Design window should appear as shown to the right.

```css
#USC_Campus {
    height: 218px;
    top: 0px;
    left: 0px;
    position: absolute;
    z-index: 30;
}
```
Step D: Create the #USC_Logo Style

In this section we will drag the “USC_Logo_Black.jpg” file into our #USC_Logo div container and then define the style to position it to be on the right side of the header.

1. Drag the file “USC_Logo_Black.jpg” off of the file list into the “div#USC_Logo” container located in the Design window.
   Note that it will appear behind the USC_Campus image so you will not be able to see it yet.

2. Click the “Manage Styles” tab.
3. Click “New Style”.
4. In “Selector”, type: #USC_Logo
5. Click the “Position” Category.
6. Set “position” to “absolute”.
7. Set “z-index” to 40.
8. Set “top” to: 60 px.
9. Set “left” to 390 px.
10. Set “left” to 0.
11. Click “OK”.

That’s it for this style. The code is shown below and the Design window should appear as shown to the right.

```css
#USC_Logo { 
  position: absolute; 
  z-index: 40; 
  top: 60px; 
  left: 390px; 
}
```
**Step E: Create the #USC_Text Style**

In this section we will create the style for our text: “University of Southern California”. Because the container for it is behind the USC_Campus container, we will create the style first and type the text later.

These steps will specify the font appearance.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In “Selector”, type: #USC_Text
4. Click the “Font” Category.
5. Set “font-family” to “Arial”.
6. Set “font-size” to 36 px.
7. Set “font-weight” to: “bold”.
8. Select a yellow color.

These steps will position the text. Note we are specifying width and height to control the where the top & bottom borders will appear.

10. Click the “Position” Category.
11. Set “position” to “absolute”.
12. Set “z-index” to 40.
13. Set “width” to 540 px.
14. Set “height” to 40 px.
15. Set “top” to 10 px.
16. Set “left” to 18 px.

These steps will place a border above and below the text. The border’s width & distance from the text was determined by the “height” and “width” above.

17. Click the “Border” Category.
18. At “border-style”, uncheck “Same for all”.
19. Set the “top” and “bottom” borders to “solid”.
20. Set “border-width” to 2 px. (This is how thick the lines are, not its distance from the text.)
21. Set “border-color” to white.
Decrease Letter Spacing
These steps will place the letters closer together by setting letter-spacing to a negative number. Note that “em” is the width of the letter “m” for the given font’s current size.

22. Click the “Block” Category.
23. Set “letter-spacing” to: \(-0.025\) \text{em}

These steps will center the text within the borders above and below it.

24. Click the “Box” Category.
25. At “padding”, uncheck “Same for all”.
26. Set the “right” & “left” padding to 5 px.

27. Click “OK”.
28. Click in the \#USC\_Text container.
29. Type: University of Southern California

```css
#USC_Text {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 36px;
  font-weight: bold;
  color: #FFDF62;
  position: absolute;
  z-index: 40;
  width: 540px;
  height: 40px;
  top: 10px;
  left: 10px;
  border-width: 2px;
  border-color: #FFFFFF;
  border-top-style: solid;
  border-bottom-style: solid;
  letter-spacing: -0.025em;
  padding-right: 5px;
  padding-left: 5px;
}
```

That’s it for this style. You should have the image shown above. The code is shown to the left.
**Step F: Create the #USC Motto Style**

In this section we will create the style for our motto: “Palmam qui meruit ferat. (Let whoever earns the palm bear it.)”. Because the container for it is behind the USC_Campus container, we will create the style first and type the text later.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In “Selector”, type: #USC_Motto

These steps will specify the font appearance.

5. Click the “Font” Category.
6. Set “font-family” to “Arial”.
7. Set “font-size” to 20 px.
8. Set “font-weight” to: “normal”.
9. Set “font-style” to “italic”.
10. Set “color” to white.

These steps will position text.

11. Click the “Position” Category.
12. Set “position” to “absolute”.
13. Set “z-index” to 50.
14. Set “width” to 260 px.
15. Set “height” to 45 px.
16. Set “top” to 120 px.
17. Click the “Block” Category.
18. Set “letter-spacing” to -.025 em.
19. Click “OK”.
20. Click in the #USC_Motto container and type the motto.

That’s it for this section! The code is shown below.

```css
#USC_Motto {
  font-family: Arial, Helvetica, sans-serif;
  font-size: 20px;
  font-weight: normal;
  font-style: italic;
  color: #FFFFFF;
  position: absolute;
  z-index: 50;
  width: 260px;
  height: 45px;
  top: 120px;
  left: 15px;
  letter-spacing: -.025em;
}
```
Step G: Create the #Content Container Style

The text below the header is also in a container and has a green background. We will create it in this section.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In “Selector”, type: #Content
4. Click the “Background” Category.
5. Select a “background-color”.

We will add 10px of padding to give some space between the content and the edge of the div.

6. Click the “Box” Category.
7. Check “Same for all”.
8. Set “padding” to 10px.

This will position and size the div. Note that its width is 20px less than the header width of 580. This is because our padding (10 on each side) adds to the outside of the width and background color extents into the padding.

9. Click the “Position” Category.
10. Set “position” to “relative”.
11. Set “z-index” to 10.
12. Set “width” to 560.
13. Click “OK”.

14. There is a text file called “About_USC_Text.txt” in the “Images” folder which contains some text. If you open it you can copy and paste the text into the #Content container of this page.

```
#Content {
  background-color: #CECE9D;
  padding: 10px;
  position: relative;
  z-index: 10;
  width: 560px;
}
```

The code for the style is shown above.
Step H: Create the #Page_CONTAINER Style
All of our <div> containers are contained within an overall div container called “Page_CONTAINER”. We did this to control the overall width of our page (580px) and to be able to center the page within the browser window by setting the left and right margins of the div to “auto”.

1. Click the “Manage Styles” tab.
2. Click “New Style”.
3. In “Selector”, type: #Page_CONTAINER
4. Click the “Box” Category.
5. At margin, uncheck “Same for all”.
6. Set “right” and “left” both to “auto”.

Note that it won’t be centered yet because we didn’t specify a width.

7. Click the “Position” Category.
8. Set “position” to “relative”.
9. Set “z-index” to 5.
10. Set “width” to 580.
11. Click “OK”. That’s it. You are done! All of the code is shown on the next page.

```css
#Page_CONTAINER {
  margin-right: auto;
  margin-left: auto;
  position: relative;
  z-index: 5;
  width: 580px;
}
```
Style Definition Codes Used In Example

The two columns below show the code for all of the styles we defined in this example. All are in the <header> area.

```
#Header {
    background-color: #800000;
    border-top-style: solid;
    border-bottom-style: solid;
    border-width: 20px;
    border-color: #000000;
    width: 580px;
    height: 218px;
    z-index: 20;
    position: relative;
}

#USC_Campus {
    height: 218px;
    top: 0px;
    left: 0px;
    position: absolute;
    z-index: 30;
}

#USC_Logo {
    position: absolute;
    z-index: 40;
    top: 60px;
    left: 390px;
}

#USC_Text {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 36px;
    font-weight: bold;
    color: #FFDF62;
    position: absolute;
    z-index: 40;
    width: 540px;
    height: 40px;
    top: 10px;
    left: 18px;
    border-width: 2px;
    border-color: #FFFFFF;
    border-top-style: solid;
    border-bottom-style: solid;
    letter-spacing: -.025em;
    padding-right: 5px;
    padding-left: 5px;
}

#USC_Motto {
    font-family: Arial, Helvetica, sans-serif;
    font-size: 20px;
    font-weight: normal;
    font-style: italic;
    color: #FFFFFF;
    position: absolute;
    z-index: 50;
    width: 260px;
    height: 45px;
    top: 120px;
    left: 15px;
    letter-spacing: -.025em;
}

#Content {
    background-color: #CECE9D;
    padding: 10px;
    position: relative;
    z-index: 10;
    width: 560px;
}

#Page_Container {
    margin-right: auto;
    margin-left: auto;
    position: relative;
    z-index: 5;
    width: 580px;
}

```

</style>
Div Structure & Application of Styles Used in Example
The code below shows how the div tags were named and nested. This code is entirely in the <body> section.

<body>
<div id="Page_CONTAINER">
  
  <div id="Header" style="visibility: visible">
    <div id="USC_Motto">Palmam qui meruit ferat. <br/>(Let whoever earns the palm bear it.)</div>
    <div id="USC_Text">University of Southern California</div>
    <div id="USC_Campus">
      <img alt="USC_Campus" src="Images/USCCampus.jpg" width="350" height="218" />
    </div>
    <div id="USC_Logo">
      <img alt="USC Logo" src="Images/USC_Logo_Black.jpg" width="166" height="130" />
    </div>
  </div>
  
  <div id="Content">
    <h2>Established: 1880</h2>
    Los Angeles was little more than a frontier town in 1880 when USC first opened its doors to 53 students and 10 teachers. Today it is a world class research university, the oldest private research university in the West.
  </div>
</div>
</body>