

ITEC447

Web Projects

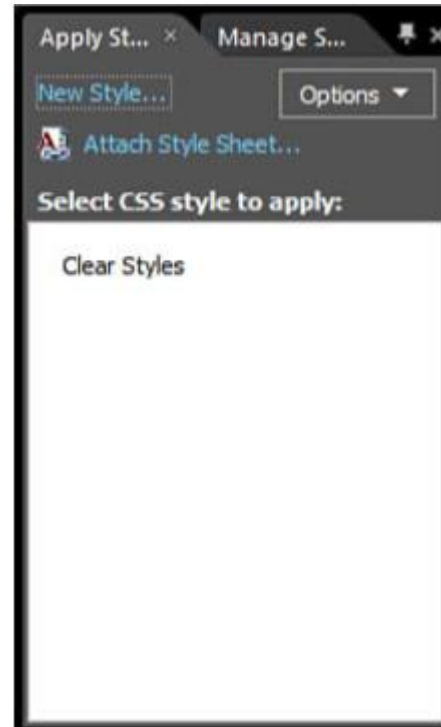
CHAPTER 5 – CSS1

Creating Styles with a Click of Your Mouse

- In this example, you change the style or look of the subheadings in the *default.html* document.
- The idea of this exercise is to show you that changing one style can affect multiple sections within the page.
- If you haven't already done so, open the *default.html* page in *Design view*.

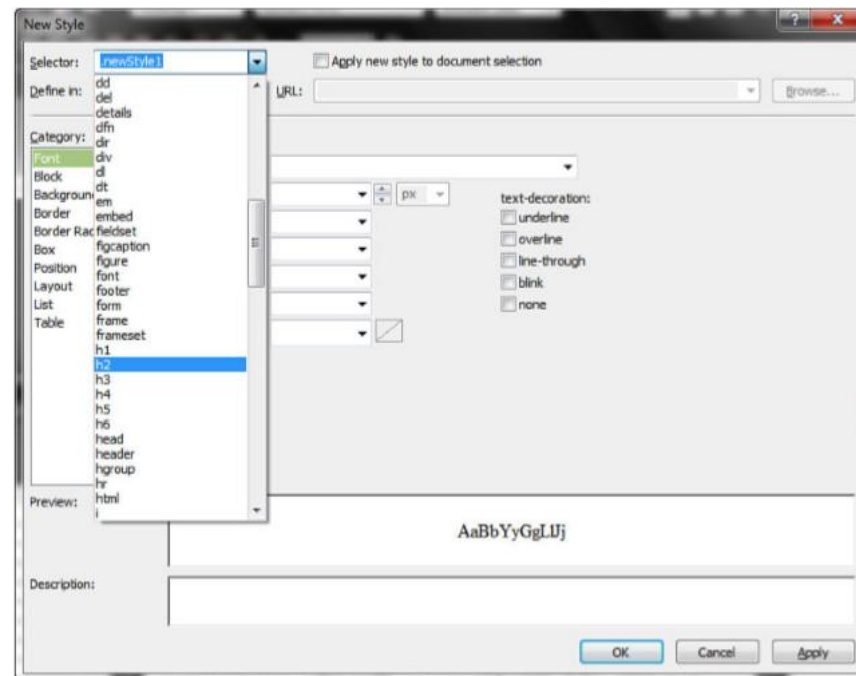
Creating Styles with a Click of Your Mouse Cont.

- With the *default.html* page open, click *New Style* in the *Apply Styles* panel.
- This opens the *New Style* dialog.



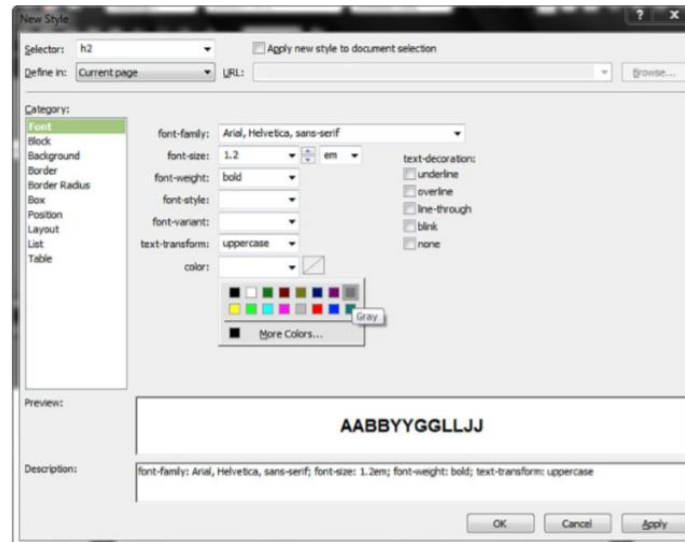
Creating Styles with a Click of Your Mouse Cont.

- In the *New Style* dialog, open the *Selector* drop-down menu and scroll down to *h2*.
- Select *h2* or, alternatively, type *h2* in the Selector bar.
- Doing so means the style you are creating applies to all the text that has Heading 2 as its style.



Creating Styles with a Click of Your Mouse Cont.

- With the Font category selected, change the *font-family* setting to *Arial, Helvetica, sans-serif* using the drop-down menu.
- Set *font-size* to *1.2* and change the *units* to *em*.
- Set *font-weight* to *bold* and *text-transform* to *uppercase*.
- Finally, change the *color* to *gray* using the drop-down menu.
- As you make these changes, you see them in real time in the *Preview* box and you see the code being generated in the *Description* box.
- Click *OK* to apply the changes.



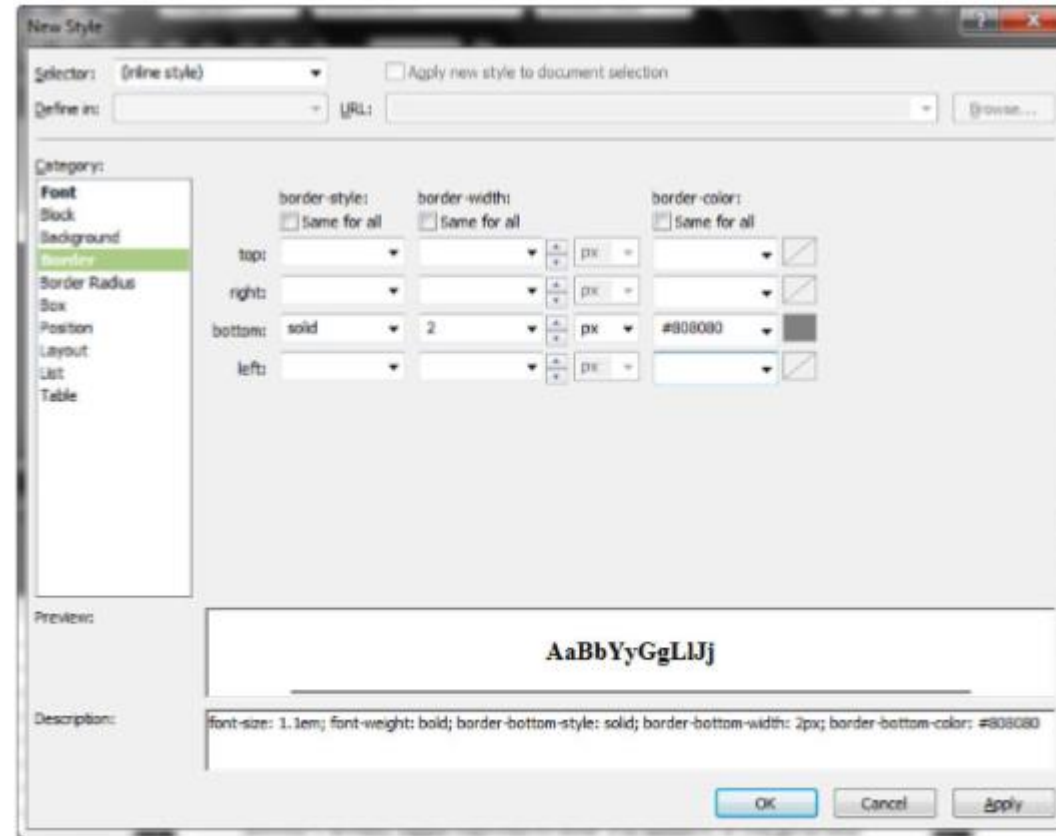
Creating Styles with a Click of Your Mouse Cont.

- By looking through the page in *Design* view, you can see that the subheadings you styled with Heading 2 have changed.
- They are now a different font, all uppercase, and gray in color.
- Now that you've changed the subheadings, you probably want to change the main heading as well.
- To do so, simply click the *New Style* button again and follow the same procedure you used to change the subheadings, but change the *Selector* setting to *h1* instead of *h2*.
- Because *h1* is the primary heading, it should be slightly larger than *h2*, so a size of *1.4em* is appropriate.
- **Note:** You are probably familiar with px (pixel), in (inch), cm (centimeter), mm (millimeter), but pt (point), em (em space; historically the width of the letter m but now the height of the letter m), and ex (x height; the height of the lowercase letter x) are new to most people.

Creating Inline Styles

- The previous example demonstrates how to make styles that affect all the content to which they apply, including new content you add later.
- If you want to make a style change to just one section of the page and do not use the style elsewhere, you create what is called an inline style.
- In Design view, place the cursor anywhere inside the first paragraph to select it.
- Click the *New Style* button in the *Apply Styles* panel as before.
- In the *Selector* drop-down menu, select (*Inline Style*).
- In the *Font* category, set *font-size* to *1.1em* and *font-weight* to *bold*.
- In the *Border* category, *uncheck* the *Same for All* boxes and set the *Border-bottom-style* to *solid*, the *Bottom border-width* to *2px*, and the *Border-bottom-color* to *gray*.
- Click *OK*.

Creating Inline Styles Cont.



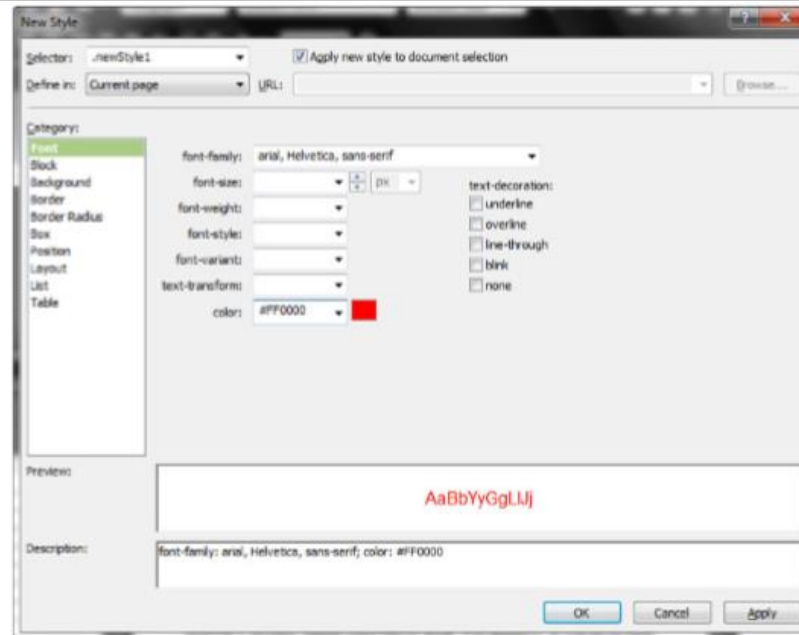
Creating Inline Styles Cont.

- As you can see in Design view, the font in the first paragraph is now bold and a 2pixel-thick line separates it from the rest of the text.
- This style is inline because it affects only one tag or segment of content rather than the whole page.

Styling Small Selections

- In the extreme, you can even set the style of just a small selection of words or even one word or one letter within a word by creating spans within the text and applying styles to them. You can do this in two ways.
- Here's the first method:
- In Design view, select the word *kipple* in the second sentence of the second paragraph.
- Using the tools available from the *Common* toolbar, change the font to *Courier New*, give the word an outside border using the *Outside Borders* button, and change the font color to *green* using the *Font Color* button.
- And here's the alternative method:
- In Design view, select another word in the second paragraph.
- Click *New Style* in the *Apply Styles* panel to open the *New Style* dialog.
- Check the *Apply New Style to Document Selection* box to ensure that the style applies only to the word you selected.

Styling Small Selections Cont.



- Under the *Font* category, change *font-family* to *Arial, Helvetica, sans-serif* and then change *font-color* to *Red*.
- Under the *Border* category, set *border-style* to *solid*, *border-width* to *1px*, and *border-color* to *Black*.

Styling Small Selections Cont.

- These two methods produce two new style classes—*.style1* and *.newStyle1*—that change the appearance of the two words you highlighted.
- The first method is quick and easy if you are making rudimentary changes; it produces a class called *.style1*.
- The second method is more cumbersome but gives you far more flexibility in terms of the final product; it produces a class called *.newStyle1*.
- One illustration of the difference between the two is that when you use the first method, the color of the box is the same as the color of the font.
- In the second method, you can set the color of the box to whatever you want and even set each side of the box to a different color.
- Save the file and test it in your browser to see the result of your style applications so far.

Styling Small Selections Cont.



Setting the Font Family for the Entire Document

- Now you know how to make changes to preset styles, sections, and individual selections of a document.
- But what if you want to define certain attributes for the entire page, such as setting all text to one particular font family unless otherwise specified?
- By default, unstyled HTML text displays in *Times New Roman*.
- You can override the default by applying styles to any tag within the HTML code of the page—even the `<body>` tag that wraps all the content. By creating a body style, you can affect all the content within the `<body>` tag (that is, all the visible content on the page).

Setting the Font Family for the Entire Document Cont.

- Here is how to change all the text in the document to Arial, Helvetica using the `<body>` tag:
- Click *New Style* in the *Apply Styles* panel to open the *New Style* dialog.
- You do not need to select or highlight any portion of the document to create the body style.
- In the *Selector* drop-down menu, select *body* or type *body*.
- Under the *Font* category, change the *font-family* to *Arial, Helvetica, sans-serif*.
- Click *OK*.
- By setting the font-family of the `<body>` tag, you changed to the new font any text that has not had its font-family defined by a custom style.
- In this example, because you created a special style for the word *kipple*, it retains the *Courier* font. The same is true for the headings, although you can't see it because the body font and the heading fonts are the same.

Styling Links (Also Known as Getting Rid of the Blue Underline)

- Click *New Style* in the *Apply Styles* panel to open the *New Style* dialog.
- You do not need to select or highlight any portion of the document to change the link style.
- In the *Selector* drop-down menu, select *a* or type *a*.
- Under the *Font* category, check the *none* option under *text-decoration*. (Even though it is not checked, the default setting for links is underline, so you have to explicitly tell the browser to not apply this option.)
- Click *OK*.
- Now, the underline is gone but the links are still blue.
- Most designers want their links to be a different color; to change the color, you need to modify the style you just created.
- You can do this using the *Modify Style* function.

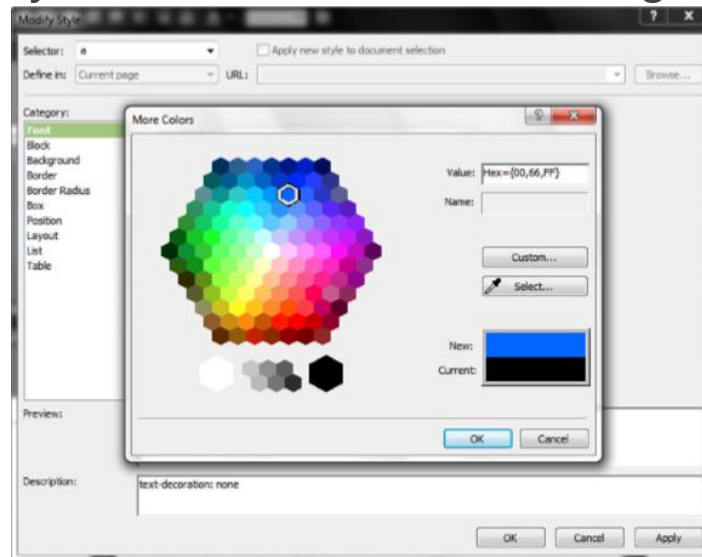
Styling Links (Also Known as Getting Rid of the Blue Underline) Cont.

- To access the styles you have already created, click the *Manage Styles* tab in the *Apply Styles* panel.
- The *Manage Styles* panel gives you a list of all the styles relating to the current document and uses visual aids to tell you what styles are active and what styles are applied to the current selection.
- If you place the cursor on a link, you see that the *a* style is highlighted, telling you that this is the last style in the cascade applied to your selection.



Styling Links (Also Known as Getting Rid of the Blue Underline) Cont.

- To edit the existing style, right-click *a* and select *Modify Style* from the context menu.
- This opens the *Modify Style* dialog, which is the same as the *New Style* dialog.
- Under the *Font* category, change *font-color* to a calmer blue using the *More Colors* option.
- Click *OK*.



- Now all the links are a nicer shade of blue. However, so are the subheadings!

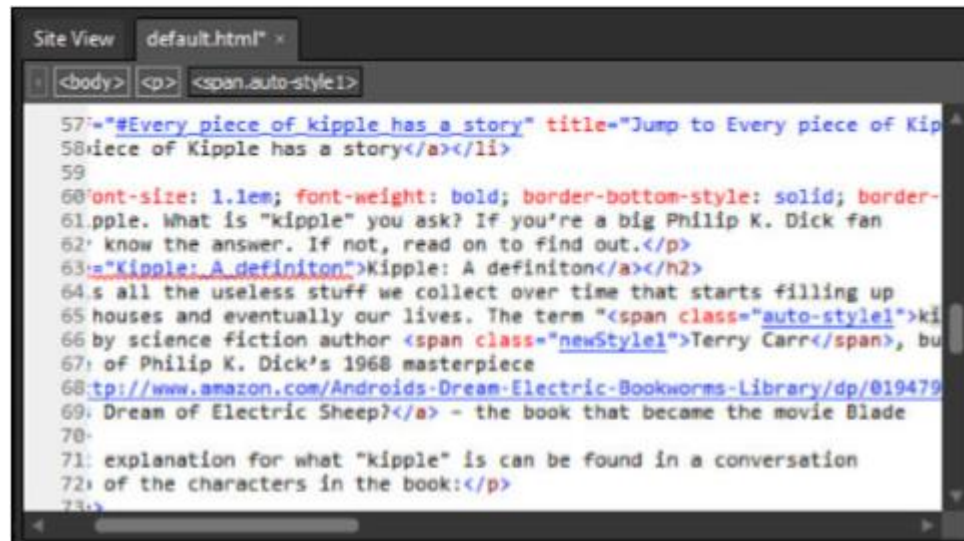
Using the Cascade to Override Styles

- In Hour 5, you set the subheadings as bookmarks using the `<a>` tag, and now the color of the `a` style is overriding the color of the `h2` style.
- This is where the cascade shows its true strength: By using CSS, you can define multiple attributes to the same content, and the browser picks the one that is most relevant based on a set of rules.
- In this case, you want to create a style that applies only to text that is both a subheading and a link.
- Click the *New Style* button in the *Manage Styles* panel to open the *New Style* dialog.
- In the *Selector* bar, type `h2 a`.
- This literally means Heading 2 links.
- In the *Font* category, change *font-color* to *gray* and click *OK*.
- Now the subheadings return to their gray color because the browser picks the most specific style in the cascade. (`h2 a` is more specific than either `h2` or `a`.)

The Quick Tag Tools

- Working with HTML documents, it can be hard to remember, or even figure out, exactly what is going on because most of the content is in tags within tags within tags.
- Trying to navigate through this mishmash of code can be a daunting task for even an experienced developer.
- To fully understand why a certain element or segment of text looks and behaves the way it does, you need to know exactly what tags are applied and in what order.
- Doing this manually is a lot of work.
- But with the *Quick Tag* tools, it is so easy it borders on the absurd.
- To get a complete ordered list of all the tags applied to an element in *Design*, *Split*, or *Code* view, simply place the cursor on that element and look at the top of the view pane.
- There, the *Quick Tag* tools list the entire sequence of tags and give you the option to edit each of them individually.

The Quick Tag Tools Cont.



```
Site View default.html x
<body> <p> <span.auto-style1>
57:="#Every piece of kipple has a story" title="Jump to Every piece of Kip
58:iece of Kipple has a story</a></li>
59
60:ont-size: 1.1em; font-weight: bold; border-bottom-style: solid; border-
61:pple. What is "kipple" you ask? If you're a big Philip K. Dick fan
62: know the answer. If not, read on to find out.</p>
63:="Kipple: A definiton">Kipple: A definiton</a></h2>
64:s all the useless stuff we collect over time that starts filling up
65:houses and eventually our lives. The term "<span class="auto-style1">ki
66:by science fiction author <span class="newStyle1">Terry Carr</span>, bu
67: of Philip K. Dick's 1968 masterpiece
68:tp://www.amazon.com/Androids-Dream-Electric-Bookworms-Library/dp/019479
69: Dream of Electric Sheep?</a> - the book that became the movie Blade
70:
71: explanation for what "kipple" is can be found in a conversation
72: of the characters in the book:</p>
73:

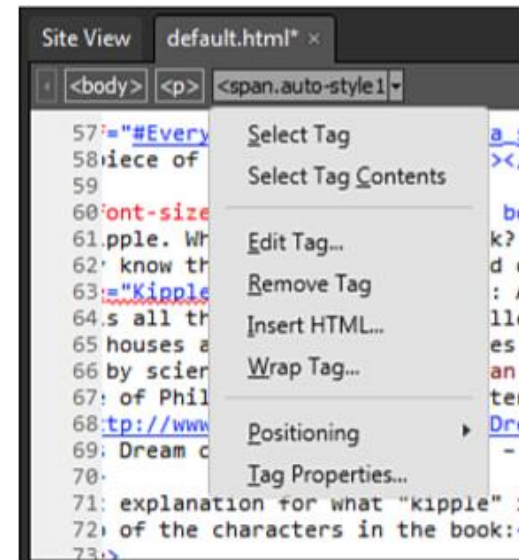
```

The Quick Tag Tools Cont.

- The *Quick Tag* tools also interact with other tools, such as the *Tag and CSS Properties* panels and the *Apply and Manage Styles* panels, to give you a complete picture of what Expression Web 4 is doing to the content.
- The *Quick Tag Selector* is the bar itself.
- After you click something in the page, the *Quick Tag Selector* displays all the applied tags.
- By hovering over each of the tags displayed on the bar, you cause a box in *Design* view to highlight the affected area of that tag.
- From here, you can click any tag in the *Selector* to display its tag or CSS properties or to show what style is applied.
- You can open the *Quick Tag* Tools dropdown menu by clicking the arrow button next to the tag and make changes to the tag without navigating through all the code.

The Quick Tag Tools Cont.

- From here, you can open the *Quick Tag Editor*, which lets you edit the tag by clicking *Edit Tag*, insert new *HTML*, and even *wrap the existing tag* in a new tag.
- You can also *remove the tag* altogether, change the *positioning* of the content, or select just the content or the whole *tag* in *Code* view.
- The *Quick Tag* tools are an important part of the CSS creation and editing process because most tags have attached styles, and you need to know what tags are at the end of the cascade to see what styles apply to a tag and what, if anything, needs to be changed.



The CSS Properties Panel

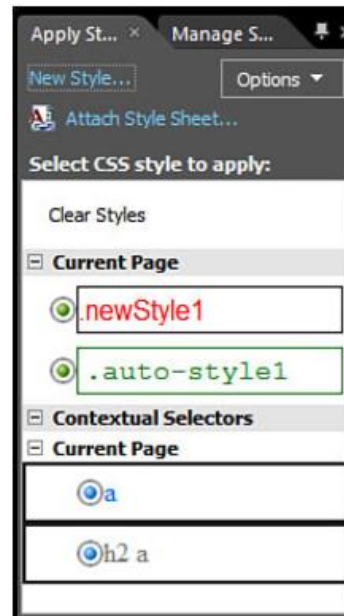
- The *CSS Properties* panel gives you a more detailed breakdown of what styles are applied to the selected elements and exactly what they do.
- The top half of the panel provides a list of all the applied rules or styles (the cascade), and the bottom part lists the attributes applied to the selected rule.
- You can select individual rules by clicking them in the top half of the panel or by clicking the tags in the *Quick Tag Selector*.
- By default, the bottom half of the *CSS Properties* panel displays a list of all the available attributes for the selected style.
- If you want to see only those attributes that have a value, click the *Summary* button.

The CSS Properties Panel Cont.

- The *CSS Properties* panel lets you make direct changes to the selected CSS style by entering new values for all the attributes.
- This is a quick way to make the same kind of changes you made in the *Modify Style* dialog, but it doesn't give you the same kind of trial-and-error environment because all the changes are immediate.
- The *CSS Properties* panel works in direct conjunction with the *Apply and Manage Styles* panels.

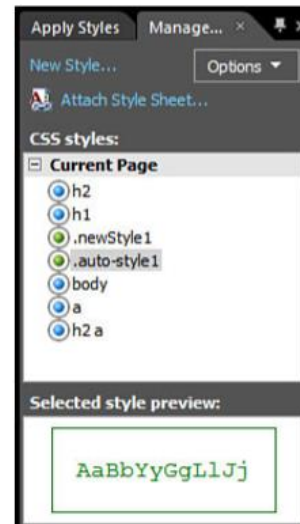
The Apply Styles Panel

- The *Apply Styles* panel, provides a visual representation of the applied and available styles by previewing them in a list.
- The primary function of this panel is to provide an easy way to apply styles to the selected content.
- To do so, simply select the content in *Code* or *Design* view and click the desired style.
- You can also use the *Apply Styles* panel to remove styles from the content and select all the content to which the style is applied on the page.



The Manage Styles Panel

- The *Manage Styles* panel, works in much the same way as the *Apply Styles* panel, with the one major difference that you can't simply click a style to apply it to a page. (You need to right-click it and select *Apply Style* from the context menu.)
- The *Manage Styles* panel provides a complete list of all the styles available in the page and a preview of the selected style.
- As the name suggests, the *Manage Styles* panel is an excellent tool for managing styles both when you want to edit a particular style and also when you start dealing with styles stored in different locations, such as multiple external style sheets.



Using Various CSS Tools to Apply and Change Styles

- Now that you know all the different CSS tools available, it's time to put them to use.
- In this example, you use the various panels to apply styles to content and then change the styles without using the *Modify Style* dialog.

Using Various CSS Tools to Apply and Change Styles Cont.

- In *Design* view, find and highlight the word *homeopape* in the *blockquote*.
- With the *Apply Styles* tab selected in the panel, click *.newStyle1* to apply it to the selected text.
- Click *<span.newStyle1>* in the *Quick Tag Selector* to select the correct tag.
- If *Summary* is active in the *CSS Properties* panel, click the button to deactivate it so that you get a list of all available attributes.
- Pin the *Folder List* panel to the side to get full access to the *CSS Properties* panel.
- Use the *CSS Properties* panel to change *font-color* to *Maroon* by clicking the drop-down menu under *Font and Color*.
- Scroll down to *fontvariant* and set it to *small-caps*.

Using Various CSS Tools to Apply and Change Styles Cont.

- To change the border, scroll up until you find the border attribute and click the + sign to see all the available attributes.
- Delete the current attributes by clicking the border attribute value and pressing the *Delete* key or the *Backspace* key on your keyboard.
- Scroll down to *border-bottom* (you might have to change the layout of your workspace to see the whole name) and set the *Border-bottom-color* to *Maroon*, the *Border-bottom-style* to *dotted*, and the *Border-bottom-width* to *1px*.

CSS Classes—Because Not All Content Should Be Treated Equally

- Until now, you learned how to apply styles to a page using the standard tag selectors such as *p*, *h1*, *h2*, and *a*.
- However, these styles applied to the entire page, so you had to make an inline style to change the style of just one section of the page.
- This is an acceptable solution if the change happens only once, but if you plan to use this special style again somewhere else in the page, this approach quickly becomes cumbersome.
- You need a way to group the content into separate classes so that each section can get its own style, even though the same selectors define them all.
- Enter CSS classes.

Create a Class and Apply It to the Content

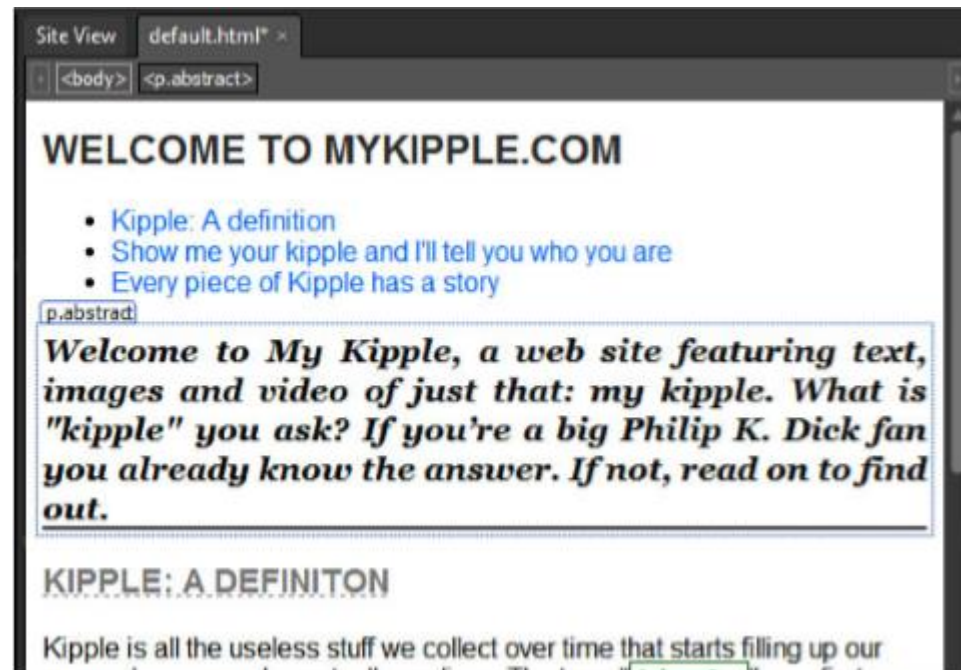
- A CSS class defines a subsection of the content that has its own set of styles.
- An example illustrates this best: Right now, there is no clear separation between the beginning part of the *default.html* page and the rest of the content.
- To remedy this, you can make a class to style this portion of the page:
- With the *default.html* page open in *Design* view, place the cursor inside the first paragraph to select it.
- In the *Apply Styles* task pane, right-click the *inline style* you created in Hour 10 and select *Remove Inline Style* from the context menu.
- This returns the paragraph to its original appearance.
- Click the *New Style* button and change the *Selector* name to *.abstract*.
- The punctuation mark in front of the name defines this style as a class.

Create a Class and Apply It to the Content Cont.

- In the *Font* category, set *font-family* to *Georgia, Times New Roman, Times, serif*; *font-size* to *1.2em*; *font-weight* to *bold*; and *font-style* to *italic*.
- In the *Block* category, set *text-align* to *justify*.
- In the *Border* category, uncheck all the *Same for All* boxes and change the *bottom* values to *solid, 2px, and #000000* (black).
- Click *OK* to create the new style class.
- To apply the new class to an existing element within the page, place the selector on the element (in this case, the first paragraph) and click the *.abstract* class in the *Apply Styles* task pane.

Create a Class and Apply It to the Content Cont.

- When you click the first paragraph after applying the new class, you can see that the p tag in the *Quick Tag Selector* has changed to include the new class.
- It now reads `<p.abstract>`.



Create a Class and Apply It to the Content Cont.

- Using the method described here to apply a new class results in the class being applied to the last tag in the chain of the selected items.
- This means that when you have grouped objects such as lists, you need to pick which tags you want to apply the class to.
- If you click one of the list objects at the top of the page and apply the class, it affects only the selected list item.
- If you highlight the entire list or select the `` tag from the *Quick Tag Selector*, Expression Web 4 applies the class to the list as a whole.

Using CSS Classes to Center an Image

- To properly center non-text content with standards-based code, you need to use CSS.
- However, although you want the option to center your images and other content, you don't want to center every image.
- Making a class to center content is the perfect solution to this problem.
- Before you start, replace the current *myDesk.html* file with the fresh one from the lesson files for this hour.
- You should do this because, when you inserted and changed the properties for the images in Hours 6 and 7, you created a series of styles.
- This new file has no styles and gives you a fresh start.

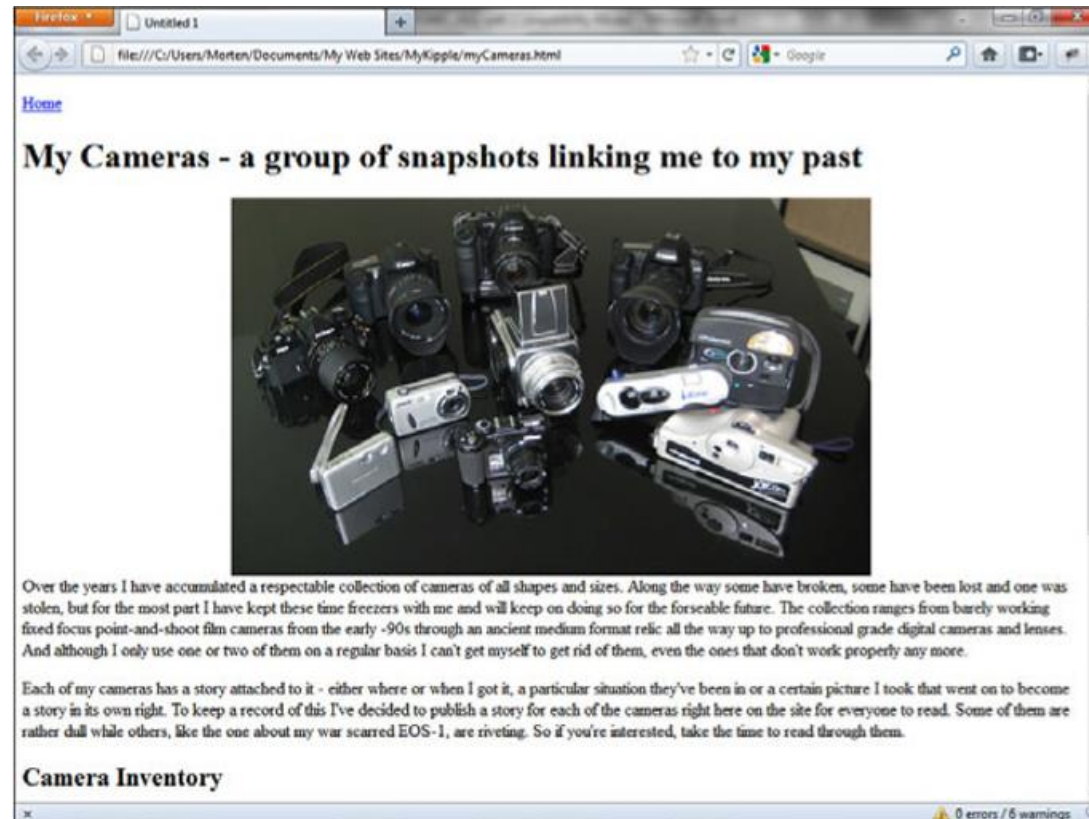
Using CSS Classes to Center an Image

Cont.

- With the *myDesk.html* page open in *Design* view, click the *New Style* button and change the *Selector* name to *.alignCenter*.
- In the *Box* category, uncheck the *Margin: Same for All* box and set *right* and *left* to *auto*. (Leave top and bottom empty.)
- In the *Layout* category, set *display* to *block*.
- This tells the browser that whatever content this class is applied to is to appear as a separate block or box independent of the rest of the content (that is, on its own line).
- Click *OK* to create the new class.
- Click the image of the *Kenny* and click the new *.alignCenter* class to apply it.

Using CSS Classes to Center an Image Cont.

- Save the page and test it in your browser, and you see that with the new `.alignCenter` class applied, the image centers itself on its own line in the page.



Using CSS Classes to Center an Image

Cont.

- The clever thing about using the method to set the left and right margins to auto is that you leave it to the browser to decide where the center of the page is by telling it that the two side margins are to be identical.
- You can create similar classes for *.alignLeft* and *.alignRight* by setting the *display* attribute under the *Layout* category to *inline* (to keep the image on the same line as the text) and setting the *float* attribute to *left* for *.alignLeft* and *right* for *.alignRight*.
- That way, you don't have to use the *Picture Properties* dialog to position your images, but you can apply classes to them individually instead.

Creating a Div and Placing It Around Content

- To understand when and how you would use divs to wrap content, you apply the *.abstract* class to all the content before the first subheading in *default.html*.
- As you saw in the previous example, adding the *.abstract* class to individual sections of the page causes Expression Web 4 to treat each section as a separate entity.
- Now, you want to create a box that contains both the first paragraph and the list above it and treat them as a single entity.
- You use the *Toolbox* panel to assist you in the next task.
- The *Toolbox* should be visible on the top-right side of the workspace.
- If it is not, you can activate it by clicking *Panels* from the main menu and selecting *Toolbox*.

Creating a Div and Placing It Around Content Cont.

- With the *default.html* page open in *Design* view, drag and drop a `<div>` instance (found under *HTML, Tags* in the *Toolbox* task pane) into the page and place it in the empty space between the first heading and the list.
- This should create a new empty horizontal box directly under the heading .



Creating a Div and Placing It Around Content Cont.

- To move the content into the *div*, simply highlight the first paragraph and drag and drop it into the *div*.
- For layout purposes, which will make sense later, you want the list to appear underneath the first paragraph, so select it and drag and drop it inside the *div* underneath the text.
- Now, the first paragraph and the list are both contained within the new div, and when you place your cursor on either, the *Quick Tag Selector* shows that the *div* comes before any of the other tags in the cascade.



Creating a Div and Placing It Around Content Cont.

- Now that we have separated some of the content from the rest of the page, it is time to make that content appear separated visually and in the code.
- To do this you use a different kind of style element called an ID.
- In addition to style classes, you also have style IDs.
- The ID differs little from the class—so little, in fact, that many wonder why it exists at all.

Introducing ID—Class’s Almost Identical Twin

- The ID works in the same way as the class: You can apply attributes to it, apply it to any tag, and create custom styles that appear only within divs that belong to this ID. The only difference between the class and the ID is that whereas you can use the class many times throughout a page, you can use the ID only once. (Or rather, if you want your page to remain compliant with web standards, you can use an ID only once per page—most browsers allow the repeated usage of the same ID in a page even though it’s technically a breach of the standards.)
- So what is the point of using IDs or having them at all? From a designer and developer standpoint, the ID is a great tool for separating content and making the code more readable. As an example, a common practice when designing blogs is to use IDs to define the main sections of the page and classes to define components that repeat several times within these sections. For example, the front page of a blog may have an ID called content that holds all the articles, and each article is kept in a class called post. For someone looking at the code, it is far easier to understand what is going on in large pages if the developer lays out the code this way.

Creating a Sidebar Using an ID

- To make the page layout more interesting, let's make the new div you just created into a sidebar that appears on the left side of the screen.
- To do this, you assign it an ID called `#sidebar` and then style that ID to make it float to the left.
- Click the *New Style* button in the *Apply Styles* panel to open the *New Style* dialog.
- Set the *Selector* to `#sidebar`. (The `#` symbol prefix tells the browser that this is an ID.)
- Under *Background*, set the *background-color* to `#CCCCCC` (a light gray) using the *More Colors* swatch or by inserting the *hex* value manually.
- Under *Border*, leave the *Same for All* boxes checked and set the *borders* to *solid*, *px*, and `#808080` (a darker gray).
- Under *Position*, set the *width* to `250px`.
- By default, the *width* of a div box is `100%`. This sets it manually to a fixed size.
- Under *Layout*, set *float* to *left*.
- This pushes the box to the far left, letting the remaining text float around it, as you saw with the images earlier.
- Click *OK* to create the new ID.

Creating a Sidebar Using an ID Cont.

- To apply the new ID, select the div by clicking the `<div>` box in the *Quick Tag Selector* and then click the new `#sidebar` ID in the *Apply Styles* panel.
- When the `#sidebar` ID is applied to the div, the browser creates a gray box around the content and shifts it to the left.

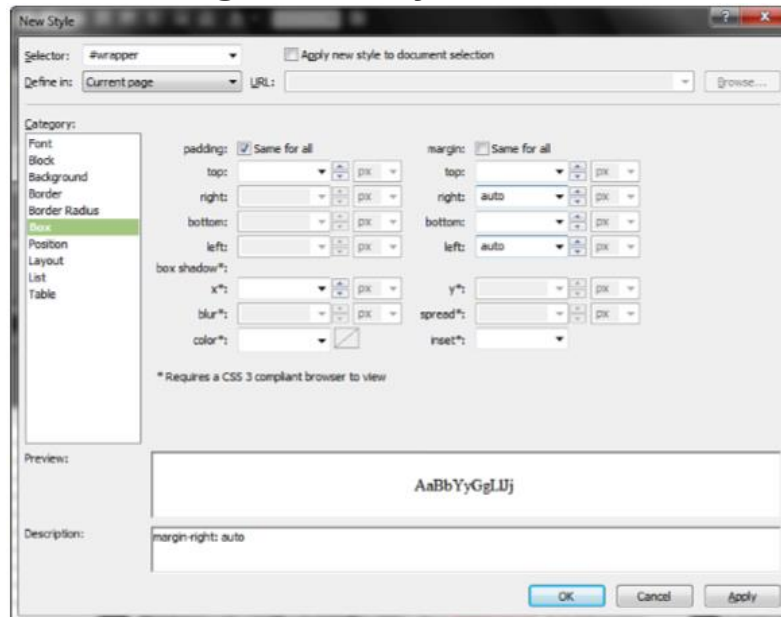


Using an ID to Center the Page

- In the past, a common way to center the content on a page was to put it in a one-cell table and center the table using *text-align*.
- This is not an ideal solution because by putting the content inside a table, you are inadvertently restricting the options for future layout changes and fancy styling.
- Even so, the table idea is a good one; it's just using the wrong type of box.
- If you paid close attention to the earlier sections of this hour, you might already have figured out how to do this using only CSS.

Using an ID to Center the Page Cont.

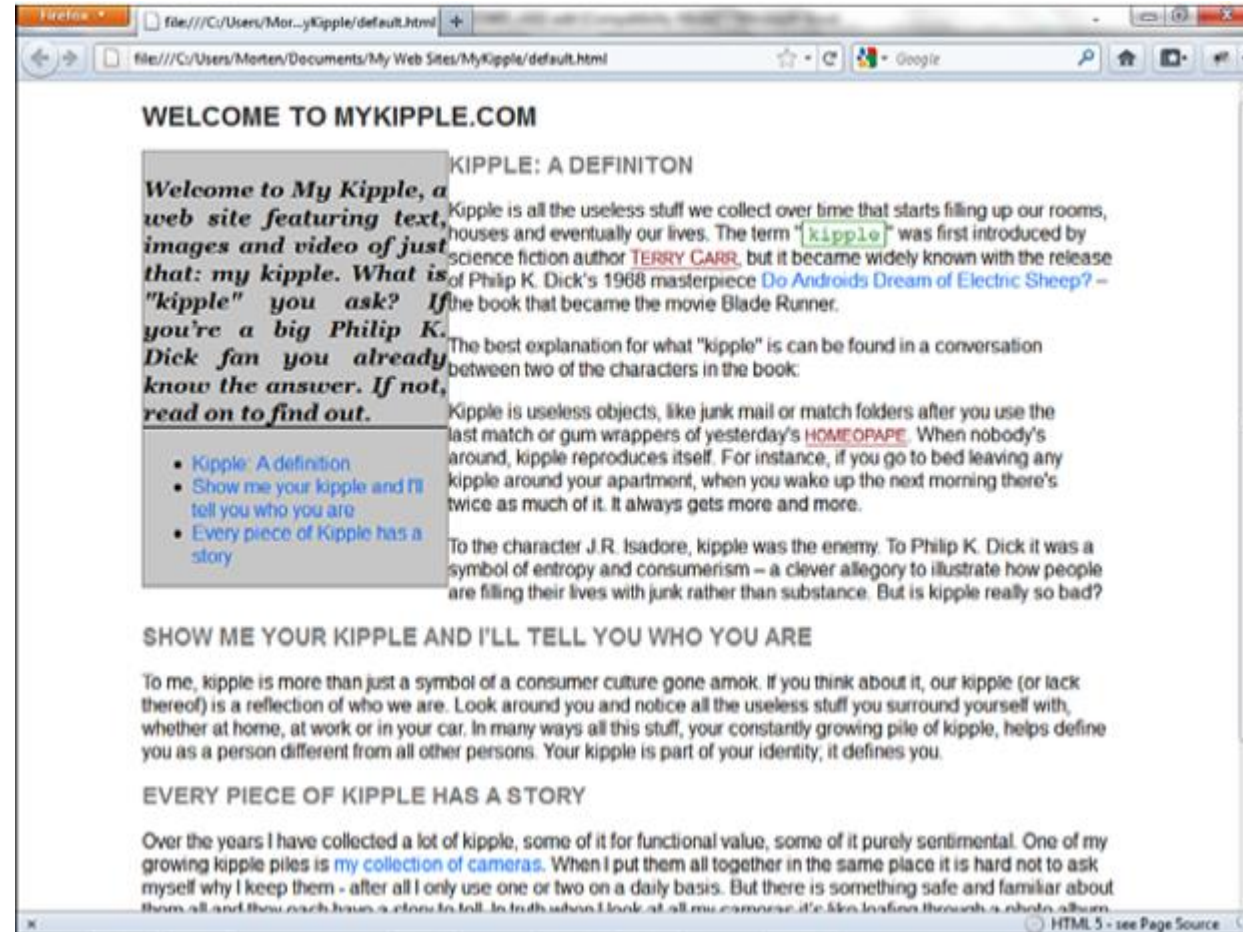
- Go to *Code* view and drag and drop a `<div>` instance found under *Tags* in the *Toolbox* task pane into the page directly before the line that reads `<h1>Welcome to MyKipple.com</h1>`.
- Go back to *Design* view and a new box appears at the top of the page.
- 2. Create a new style and give it the *Selector* name `#wrapper`.
- 3. In the *Box* category, uncheck the *Margin: Same for All* box and set *left and right* to *auto*.
- Leave *top* and *bottom* blank.



Using an ID to Center the Page Cont.

- In the *Position* category, set *width* to *800px*.
- This will be the total width of the content on the page.
- Click *OK* to create the new ID.
- In *Design* view, highlight all the content underneath the new *div*, including the sidebar, and then drag and drop it into the *div* you just created at the top of the page.
- Select the *div* tag belonging to the new div from the *Quick Tag Selector* bar and click the new *#wrapper* ID in the *Apply Styles* task pane to apply the ID.
- The tag changes to `<div#wrapper>`.
- Save and then press *F12* to preview the page in your browser.
- The content of the page should now be restricted to the center of the page and remain so even if you resize your browser window.

Using an ID to Center the Page Cont.



Using an ID to Center the Page Cont.

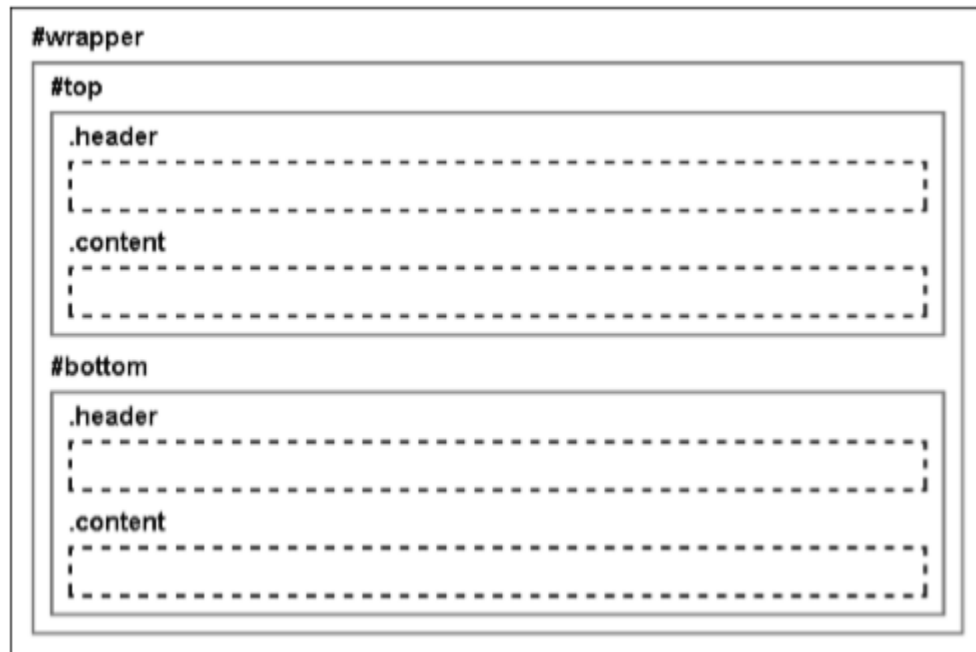
- When you apply this ID to your div, Expression Web 4 reduces the *width* of the div to *800px* and tells the browser to place the div within two equally wide margins: one on the left and one on the right.
- Naturally, this results in the div box appearing in the middle of the screen.
- To position the content to the left or right of the screen, simply remove the two margin attributes and set *float* to *left* or *right* instead.

Creating Custom Styles Within IDs and Classes

- When you have content that is contained within a div that has an ID or class, you can create custom styles that affect only the tags within that class.
- You do so by making the tags a sub-element of the class.
- To do so, create a new style but give the tag a prefix in the form of the class name.
- For example, you can make a custom version of the *.abstract* class that applies only to content within the *#sidebar* div.
- To do so, create a new style and give it the selector name *#sidebar .abstract*.
- In the *Font* category, set *font-size* to *1em* and *font-weight* to *normal*.
- When you click *OK* to create the new style, you see that attributes from both of the *.abstract* classes are applied to the content but that the attributes from the *#sidebar .abstract* class have preference.
- That is because the more specific style is further down the cascade and closer to the content.
- You can apply this technique to any standard tag, whether it is a heading, paragraph, link, blockquote, or something else.

Classes Within Classes: Micromanaging the Content

- In the earlier example, you saw that you can create special styles for content within IDs and classes.
- There is no limit to how far you can take this technique by applying multiple IDs, classes, and tags within each other.



- In this figure, multiple IDs and classes divide different parts of the content.
- By understanding how to properly name your style selectors, you can micromanage the content within these IDs and classes for a highly customized look.
- You do so by creating selector names that have the relevant IDs, classes, and tags listed with spaces between them.

Classes Within Classes: Micromanaging the Content Cont.

- Here are some examples of different *selector* names:
- *p* styles all paragraphs on the page, both inside and outside the IDs and classes.
- *#wrapper p* styles all paragraphs within the wrapper ID.
- *#wrapper #top p* and *#top p* style paragraphs within the top ID only.
- *.header p* styles all paragraphs within the header class regardless of ID.
- *#wrapper #top .header p* and *#top .header p* style paragraphs within the header class inside the top ID only.

Using Classes to Control IDs

- To see just how flexible the tag, class, and ID structure is, consider this trick used by professional designers for quick-and-easy prototyping: Right now, the sidebar floats to the right because the ID contains a float variable.
- However, you can also use a class and apply it to the ID to do this!
- Earlier in this hour, you created two alignment classes called *.alignLeft* and *.alignRight* in the *myDesk.html* page.
- Now, create the two classes in *default.html* and give them the following attributes:
- For *.alignLeft*, set *margin-right* to *10px* and *float* to *left*.
- For *.alignRight*, set *margin-left* to *10px* and *float* to *right*.

Using Classes to Control Ids Cont.

- Next, open the *Modify Style* dialog for the *#sidebar* ID by right-clicking the style in the *Apply Styles* panel and selecting *Modify style* from the pop-up menu.
- In the dialog, go to the *Layout* category and remove the *float:left* attribute.
- Click *OK* to save the change, and the text should no longer wrap around the sidebar.
- Place your cursor anywhere inside the sidebar box and select the `<div#sidebar>` box in the *Quick Tag Selector* to select the whole div.
- Then, go to the *Apply Styles* panel and click the *.alignLeft* style to apply it.
- With the application, the sidebar floats to the left with a nice 10px margin as a buffer against the other content.
- Without making any changes, click the *.alignRight* style instead and, as if by magic, the sidebar jumps to the right with the text floating to the left.
- This is because Expression Web 4 won't let you apply two styles to the same div, so it overwrites the last one you applied.
- This way, you can quickly see which layout you like better. And this trick doesn't just apply to the sidebar—you can do the exact same thing with images and other elements on the page.

Pseudoclasses

- In addition to tags, classes, and IDs, HTML supports something called pseudoclasses.
- These specialized versions of selectors come into play when the user interacts with the page; that is, when the user hovers over or clicks content or a link.
- There are five such pseudoclasses, all of which are normally used in conjunction with the `<a>` tag:
- `:active` refers to an element that is currently active. For example, a link during the time the user is holding the mouse button down and clicking it.
- `:focus` refers to an element that currently has the input focus, meaning that it can receive keyboard or mouse input. To understand focus, think of an input table with the current cell highlighted—that cell has the focus. When you press the Tab button, the focus changes to the next cell.
- `:hover` refers to an element being hovered over by the mouse pointer.
- `:link` refers specifically to an element that is an unvisited hyperlink. Unlike the preceding pseudoclasses, `:link` applies to the `a` tag only.
- `:visited` refers to a link that has already been visited. Like the `:link` pseudoclass, `:visited` only applies to the `a` tag.

Use Pseudoclasses to Style Links

- If you do not define an a style, browsers will style hyperlink so that *:link* is set to *blue*, *:active* is set to *red*, and *:visited* is set to *purple*.
- If you define only an a style, it overrides all the default settings and the link appears the same regardless of what the user does.
- To give the visitor a visual guide to what she is doing, it is a good idea to style the main pseudoclasses for links within your page.
- To use pseudoclasses, all you need to do is attach them directly after the tags in the selector name.

Use Pseudoclasses to Style Links Cont.

- With *default.html* open in *Design* view, create a new style.
- In the *Selector* area, use the drop-down menu to find *a:active* or type *a:active*.
- In the *Font* category, set the *color* to *red* (#FF0000).
- Click *OK* to finalize the new style.
- Create a new style and give it the selector name *a:hover*.
- In the *Font* category, check the *underline* box under *text-decoration*.
- Click *OK* to finalize the new style.
- Create a new style and give it the *selector* name *a:visited*.
- In the *Font* category, set the *color* to *gray* (#808080).
- Click *OK* to finalize the new style.

Use Pseudoclasses to Style Links Cont.

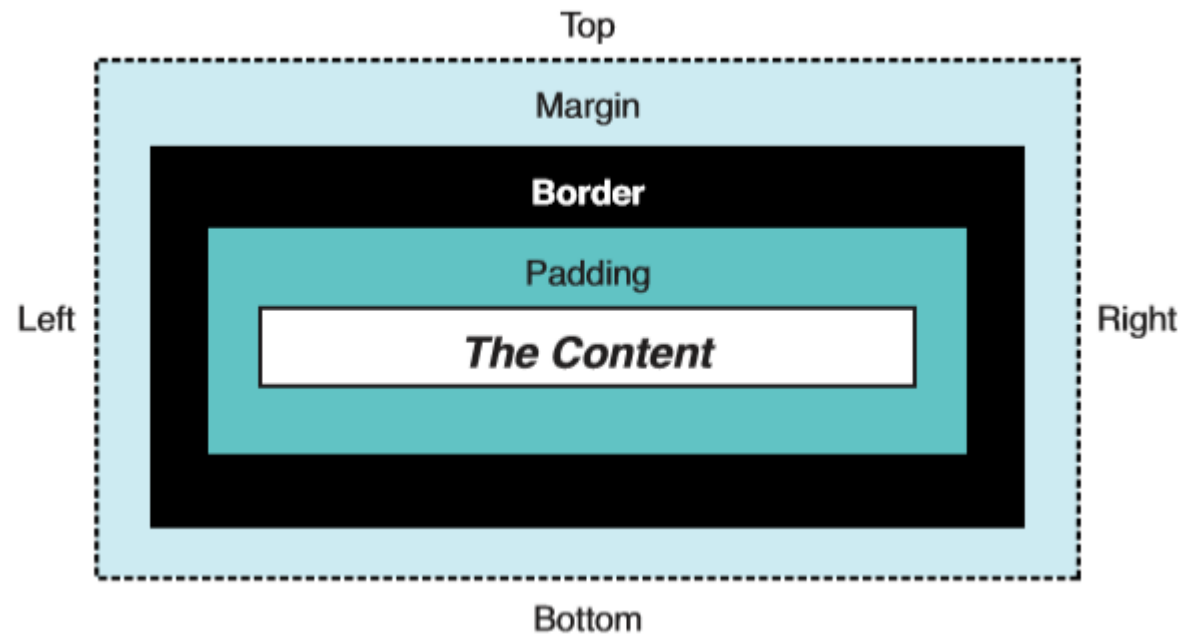
- You will not see any changes in *Design* view, but when you save and preview the page in your browser, you will see that the links on the page are blue when they have not been visited and are not being hovered over.
- They have an underline when they are being hovered over, turn red when you press and hold them, and turn gray if they have already been visited.
- Note that because you attached the pseudoclasses to the general `a` style, they are applied to all the instances of the `<a>` tag in the page, including the subheadings that work as bookmarks.
- And because the pseudoclasses are more specific, they override the `h2 a` link as well.
- If you want the `h2` links to have separate pseudoclasses from the other links on the page, all you have to do is create a new style with a *selector* name, such as `h2 a:hover`.

Use Pseudoclasses to Style Links Cont.

- For even more advanced control, you can combine several pseudoclasses by stacking them.
- As an example, right now when you hover over a link, it retains the current color and displays an underline regardless of whether you visited it before.
- By creating a separate style with the *selector* name `a:visited:hover`, you can display hovered-over visited links in a different color.
- Just as with tags, classes, and IDs, you can attach any styling attribute to pseudoclasses.

Understanding the Box Model

- Previously in this hour, you learned that when you attach tags to your content, Expression Web 4 creates an invisible box around the content.
- To understand how the content behaves and how you can style it, you need a firm understanding of the box model.



Understanding the Box Model Cont.

- To get a better understanding of what the box model is and how you use it, let's take a closer look at the *#sidebar* ID you created in *default.html*.
- To do so, rightclick the *#sidebar* ID in the *Apply* or *Manage Styles* task pane and then select *Modify Style* to open the *Modify Style* task pane.
- All content wrapped inside tags has four main areas.
- In the center is the content itself, and surrounding the content, is the padding.
- The padding is the “breathing space” that separates the content from the next area, the border.
- The padding retains the same background color or image as the content.
- The border is the outer edge of the box.
- It can be given any color, be solid, or have a number of different textures.
- Outside the border is the margin.
- The margin works as the buffer area between the outer edges of the box (the border) and the other content on the page.
- The margin is transparent and you cannot give it a distinct color.

Understanding the Box Model Cont.

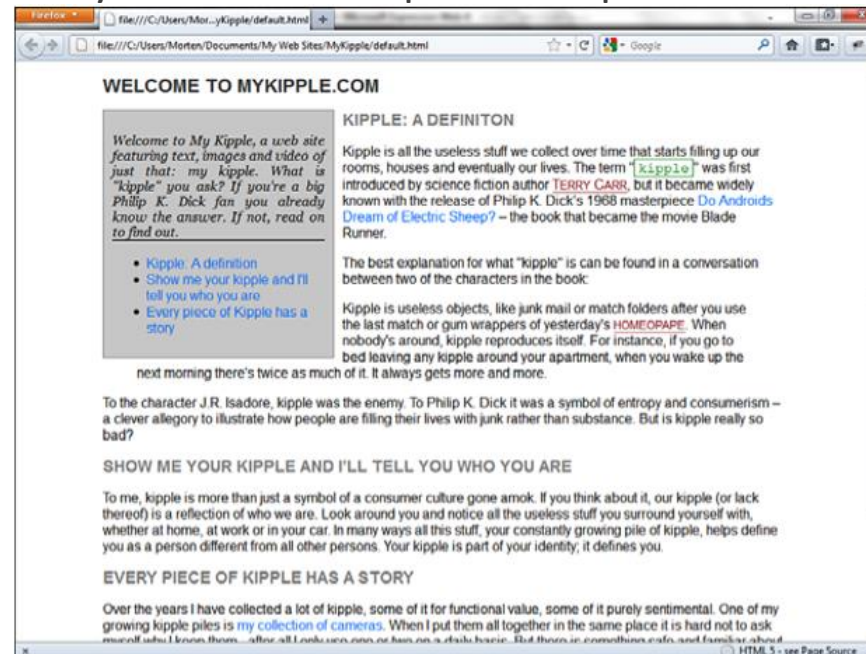
- You can set the values of each of the four sides of the padding, border, and margin independently or in groups from the *New* and *Modify Style* panels.
- To set all four sides of the padding, border, or margin, check the corresponding *Same for All* box and enter the desired value in the first box.
- To set the values for each side independently, uncheck the *Same for All* box and set each value.
- If you leave any values empty, the default value applies.
- The default value is usually *0px*.
- **Note:** The tricky part about the box model is the calculation of width and height. Generally, the width and height of any boxed element are equal to the distance from side to side or top to bottom of the content area before the padding is applied. The thickness of the padding, borders, and margins add to the total width and height of the box. This means that if you create a div with a width of 800px, as you did earlier, and give it a border of 2px on each side and a padding of 10px on each side, the total width of your div becomes 824px. As a result, if you want to keep the total width of your div at 800px, you need to subtract both your border width and your padding width and set the width of your div to 776px. It's not rocket science, but if you forget this little piece of information, you could easily end up with content that doesn't fit and not understand why.

Using the Box Model to Style Content

- Now that you know how the box model works, you can use it to create layout elements that are far more functional than tables. In this example, you change the appearance of the sidebar *default.html* page by changing the #sidebar ID style.
- With *default.html* open in *Design* view, right-click the #sidebar ID and select *Modify Style* to open the *Modify Style* dialog.
- Right now, there is no space between the edge of the div box and the text, so the text appears attached to the left wall.
- To solve this problem, change the padding of the div: Under the *Box* category, check the *Padding: Same for All* box and set *padding* on all four sides to *10px*.
- This creates some breathing space between the inner edge of the box (the content) and the borders.
- You can use the *Preview* box to make sure the space is created correctly.
- Click *OK* to apply the changes to the style.

Using the Box Model to Style Content Cont.

- When you save and preview the page in your browser, you can see that the breathing space inside and around the sidebar (previously added with the alignment class) gives the entire page a much nicer layout.
- Because the outer #sidebar ID has a specific width defined and you have now added padding, you have to reduce the width attribute if you want to keep it at 250px width.



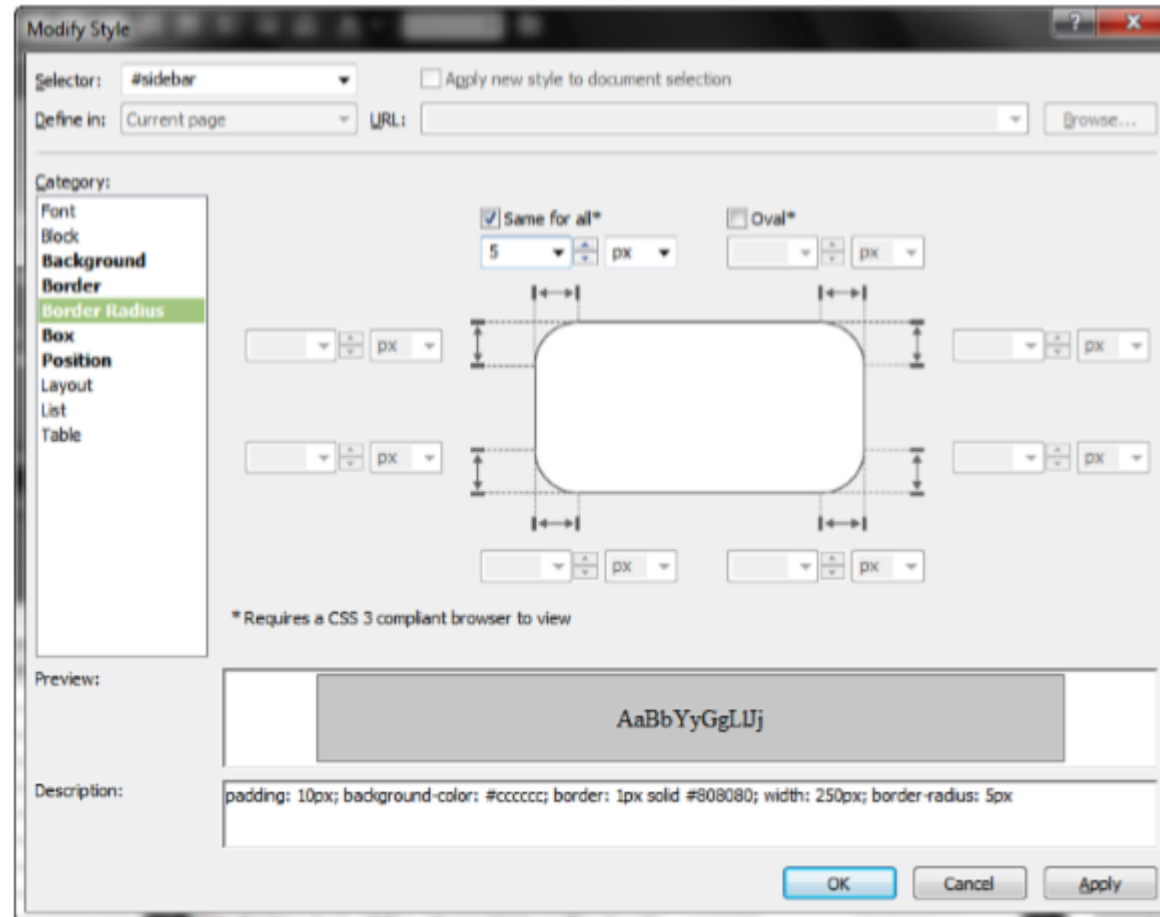
CSS3 Rounded Corners with Border Radius

- Rounded corners have always been a desired design attribute that was hard to maintain in web design.
- With the introduction of CSS3, this is no longer the case.
- CSS3 allows us to create advanced rounded corners using only code, and it works quite well in newer browsers.
- As an added bonus, if the browser does not support CSS3 rounded corners, the corners will just be regular squares without any fuss.

CSS3 Rounded Corners with Border Radius Cont.

- Open the *Manage Style* panel for the *#sidebar* ID and select *Border Radius* from the *Category* list.
- This opens the new *Border Radius* dialog.
- From here you can set a consistent border radius for all four corners or for each individual corner and you can also use the *Oval* option to set the height and width of each corner.
- The value you input in the fields defines the radius of the corner circle, so the higher the number, the rounder the corner gets.
- Set the *radius* to *5px*, and click *OK* to apply the change.

CSS3 Rounded Corners with Border Radius Cont.



CSS3 Rounded Corners with Border Radius Cont.

- Note that in *Design* view, nothing changed.
- This is because Expression Web 4's *Design* view does not support advanced CSS3 features such as *Border Radius*.
- That doesn't mean your change didn't take effect—it just means you can't preview it in the application.
- To see your rounded corners, you have to preview the page in a modern browser.