CHAPTER 3

Starting Off Right: Configuring Expression Web

In the last chapter, you created a local website using the default settings. In this chapter, you will change the default settings in Site Settings and Page Editor Options to improve your workflow. You'll configure Expression Web to make your website easier to maintain by using classes that improve workflow, instead of inline styles and program defaults.

Using the Tools Menu

Like the Format menu, the Tools menu has many useful options (see Figure 3-1). You will probably need to set some of the options here only once, but they are significant. They are Application Options and Page Editor Options, which are important enough to warrant a detailed discussion of the options on their tabs. Like others we've discussed, this menu can easily be divided into sections. You'll start with the general tools you find in most Microsoft programs.

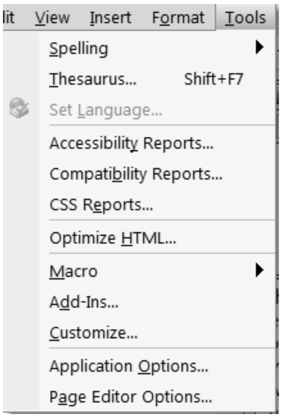


Figure 3-1. *The Tools menu is where you set Expression Web default options, as well as a few content and reporting tools.*

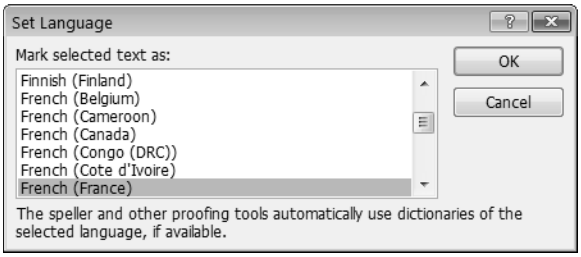
General Tools Section

The general tools are as follows (the first two are shared with Office 2007 and will be installed even if Office 2007 is not present):

- *Spelling*: This option has a submenu that allows you to run the spell-checker and set spelling options, which are self-explanatory.
- *Thesaurus*: This option is useful only if you are writing content.
- *Set Language*: All of your web pages should have default language settings. However, sometimes you will use more than one language on a page. Use the Set Language option to indicate you have changed the language within a page. For example, you might have the French phrase *c'est la vie* in the middle of your content. Normally, to tell the browser and any screen reader or search engine about the language change, you need to wrap the text in a span with the new language identified. In my example, the code to change the language would be as follows:

  ```
  <span lang="fr"> c'est la vie. </span>
  ```

  By highlighting the text in Design view and using Tools ➤ Set Language, you can choose the language without needing to write the language code yourself, as shown in Figure 3-2.



Mark selected text as:

Finnish (Finland)
French (Belgium)
French (Cameroon)
French (Canada)
French (Congo (DRC))
French (Cote d'Ivoire)
French (France)

The speller and other proofing tools automatically use dictionaries of the selected language, if available.

Figure 3-2. *Changing the language of a section of text is as simple as choosing the correct language from the scrolling text box.*

Reports Section

This section is where you can run reports that let you know what problems, if any, there are on your page. Here, you will just see what reports are available. In Chapter 12, you will be using the Accessibility Report. The reports are as follows:

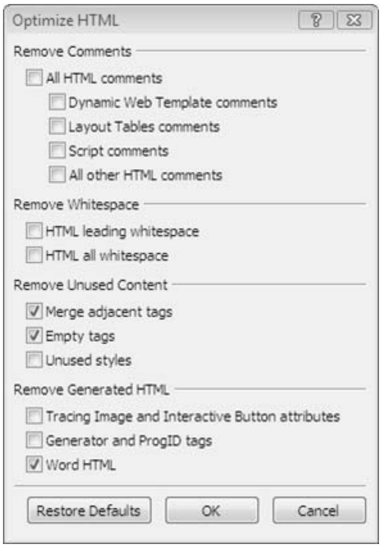
- *Accessibility*: Checks for Section 508 and Web Accessibility Initiative (WAI) compliance issues with the page. Please note that while the checker can provide a basis for determining whether the page is accessible or not, *no* automated checker can determine that accessibility has been achieved.

- *Compatibility Checker*: This report is more or less a local validator for well-formed HTML/XHTML and CSS code. Since you should always have a `doctype` declared on your pages, I recommend using the Checkbox control to base the check on the `doctype` specified in the page. You can use this checker as you are creating pages, and the result should be the same as you get by using the W3C validator, which you should also run once you have published your pages.
- *CSS Reports*: This option lets you determine what styles are or are not used within your site as well as checking for errors. Sometimes when you have been working on a site, styles accumulate that you end up not using. Many people prefer to remove these unused styles to both tidy up their stylesheets and reduce the page weight.

Notice that you have the ability to run checks on the entire site or on selected pages.

Optimize HTML Selection

The options in the Optimize HTML dialog box, shown in Figure 3-3, are both useful *and* dangerous. Like in the Format ➤ Remove Formatting selection, you can clean up legacy code as well as data from Word files and other imported files with `<font>` tags. These useful options are located in the Remove Unused Content and Remove Generated HTML sections. If you have content imported from Word or legacy FrontPage sites, I recommend using all the options in these two sections. If you have an external stylesheet applied, detach the stylesheet before removing Unused Styles.


Optimize HTML

Remove Comments
- ☐ All HTML comments
  - ☐ Dynamic Web Template comments
  - ☐ Layout Tables comments
  - ☐ Script comments
  - ☐ All other HTML comments

Remove Whitespace
- ☐ HTML leading whitespace
- ☐ HTML all whitespace

Remove Unused Content
- ✓ Merge adjacent tags
- ✓ Empty tags
- ☐ Unused styles

Remove Generated HTML
- ☐ Tracing Image and Interactive Button attributes
- ☐ Generator and ProgID tags
- ✓ Word HTML

Figure 3-3. *The Optimize HTML dialog box is one way to clean up imported code or legacy sites.*

OPTIMIZATION OPTIONS NOT TO USE

I do not recommend using the Remove Whitespaces options. If you use them, your code will run together, making it very hard to troubleshoot or even read. Some people feel that removing whitespace will protect their code, which is not true since most web editors easily reformat the code.

Tip In Expression Web, simply right-click while you're in Code view, and select Reformat HTML to restore text indenting.

Another reason some people remove whitespace is to eliminate a few bytes of page weight. You will not reduce the weight enough to make it worth the time it takes to reformat the code when you need to work on the page again.

The other section I don't recommend using, or at least using with extreme caution, is the Remove Comments section. HTML comments are something that I encourage you to use, since on many occasions, you might not look at a web page for long periods of time and work on other projects in between. When you return to your page, you might not remember why you did something. HTML comments will remind you why you wrote the code you did. Many designers use comments to identify structural parts of the page. Script comments can serve the same function, but if you are using third-party scripts, they might require you to leave attribution comments in the code. Older browsers or those that do not have JavaScript enabled will keep the text of the script from being displayed in the browser. For these reasons, I recommend that you leave comments in your code.

Program Management

Expression Web is highly extensible. Add-ons are available from Microsoft and third-party add-on makers that will give your site more features. Some of the add-ons might be downloaded for free and others purchased. Add-ons available at the time of this writing include PayPal buttons, shopping carts, form tools, Flash players, task pane managers—and more are becoming available all the time. The tools in the program management section allow you to create and use add-ins:

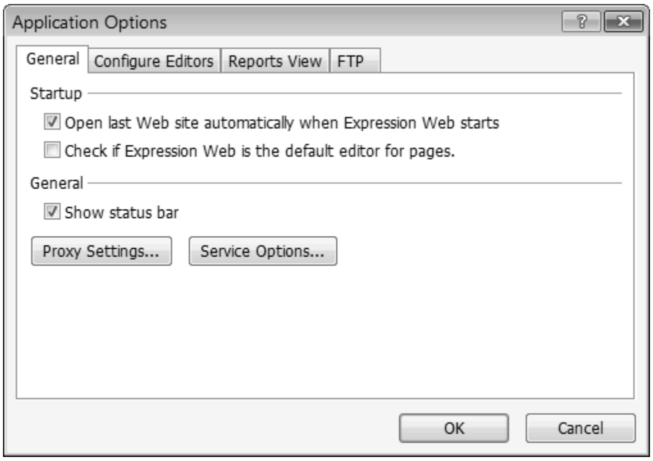
- *Macros*: The fly-out menu here allows you to run macros you have created or launch the Visual Basic Editor or Microsoft Script Editor.
- *Add-Ins*: Third-party tools and applications you can use in Expression Web can be accessed here.
- *Customize*: This lets you customize the Format and other toolbars.

Preferences Section

In this section of the Tools menu, you set defaults for how Expression Web will start up, handle different file types, interact with other programs used in web design such as graphics programs, and handle coding options.

Application Options

Clicking Application Options launches the dialog box shown in Figure 3-4.


Application Options

General | Configure Editors | Reports View | FTP

Startup

☑ Open last Web site automatically when Expression Web starts

☐ Check if Expression Web is the default editor for pages.

General

☑ Show status bar

Figure 3-4. *By default, Expression Web will make itself your HTML editor, and it will automatically open the last website you worked on when it is launched.*

The General, Reports View, and FTP Tabs

The settings under the General, Reports View, and FTP tabs rarely need to be changed from the defaults. However, you will need to use an external graphics editor, since Expression Web has very limited image-editing capabilities, and when you double-click a graphics file, Expression Web will open Microsoft Paint to edit images. If you do not have a graphics program, such as Expression Design, Photoshop, or Fireworks, you can download Paint.NET from `http://www.getpaint.net/index.html`. Although this is not as full featured as the others named, it is a good free graphics program. Later in this chapter, you will be adding a third-party graphics editor and setting the editor you add as the default.

The Configure Editors Tab

Figure 3-5 shows the Configure Editors tab. Notice that Expression Web correctly picked up that my default graphics editor is Fireworks. If you have only one graphics editor and it is correctly identified, you can skip the remainder of this section.

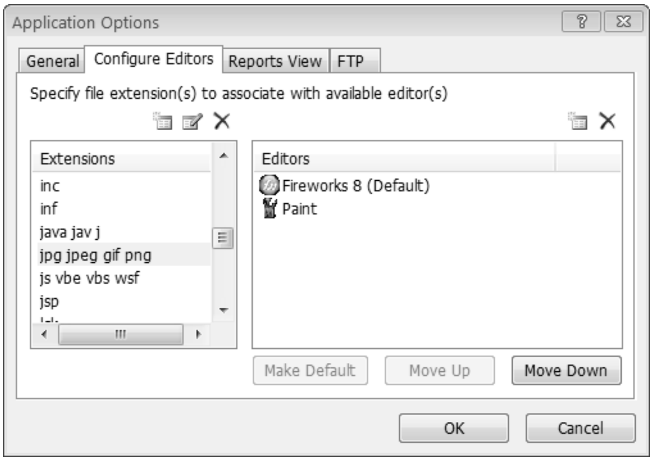


Figure 3-5. *By selecting a file extension from the left-side list, you can check what will happen when you double-click a file with that extension in Expression Web.*

If you have more than one editor or if the editor you'd like to use is not marked as the default, follow these steps to select the editor:

1. To begin adding a new editor, locate the file extension that you want to change to a different editor; in this case, select the entry for "jpg, jpeg, gif, png" from the Extensions section of the dialog box (on the left side), as shown in Figure 3-6.

■ **Note** If you want to use different editors for each of these extensions, you could delete each default group and add each extension back, either individually or grouped to fit your preferences.

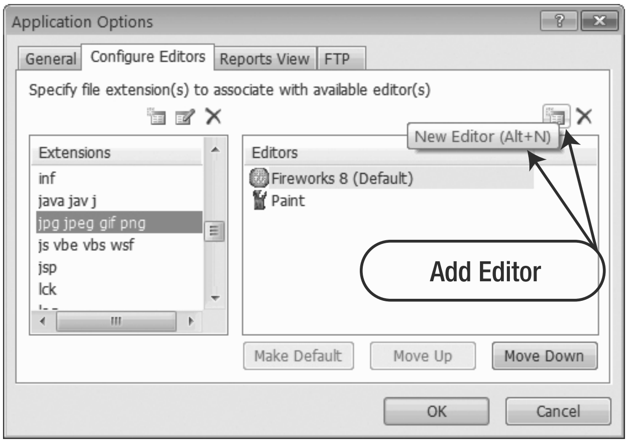
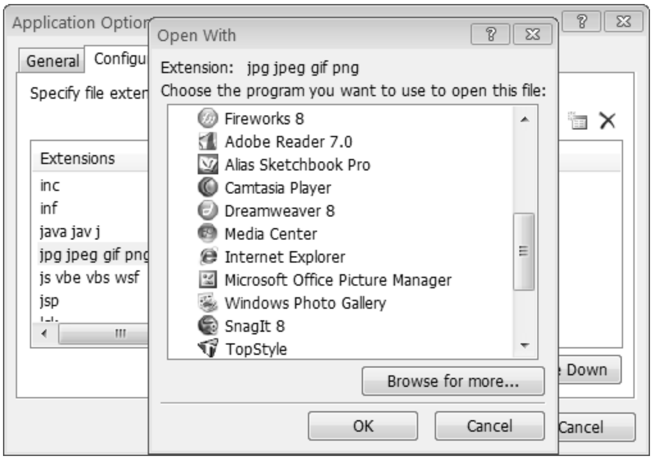


Figure 3-6. *To add an editor, click the file icon button.*

2. Next, select the folder icon above the Editors section, or if you prefer keyboard shortcuts, press Alt+N, as shown in Figure 3-6, to launch the Open With dialog box (shown in Figure 3-7).



Open With

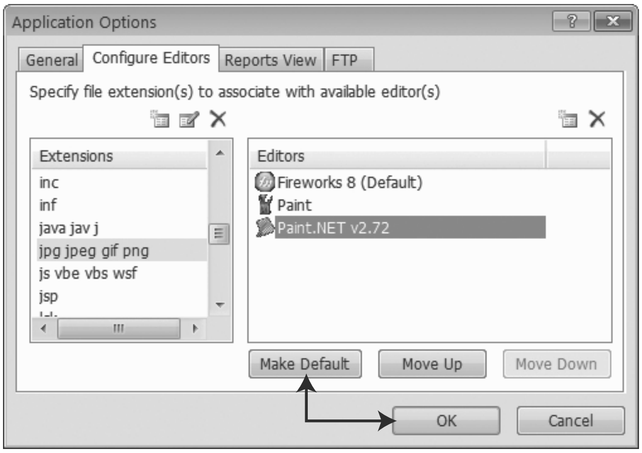
Extension: jpg jpeg gif png

Choose the program you want to use to open this file:

- Fireworks 8
- Adobe Reader 7.0
- Alias Sketchbook Pro
- Camtasia Player
- Dreamweaver 8
- Media Center
- Internet Explorer
- Microsoft Office Picture Manager
- Windows Photo Gallery
- SnagIt 8
- TopStyle

Figure 3-7. *Select the program you want to add.*

3. In the Open With dialog box, you can choose which editor you want to use. If the program is listed in the Open With dialog box, select the program and click OK. The editor I want to add, Paint.NET, is not listed, which means I will need to use click the "Browse for more" button to locate its `.exe` file. This will add the program to your editor list. If you want to set it as your default, click the Make Default button, as shown in Figure 3-8before you close the dialog box with the OK button.



Specify file extension(s) to associate with available editor(s)

Extensions
inc
inf
java jav j
jpg jpeg gif png
js vbe vbs wsf
jsp

Editors
Fireworks 8 (Default)
Paint
Paint.NET v2.72

Figure 3-8. *Make sure that you use the OK button to save your changes.*

For any file extension with more than one editor listed, Expression Web will give you the option to open the page with all of the programs on the list, as shown in Figure 3-9.

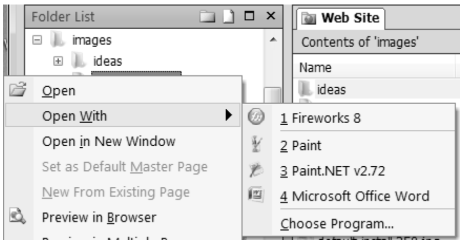


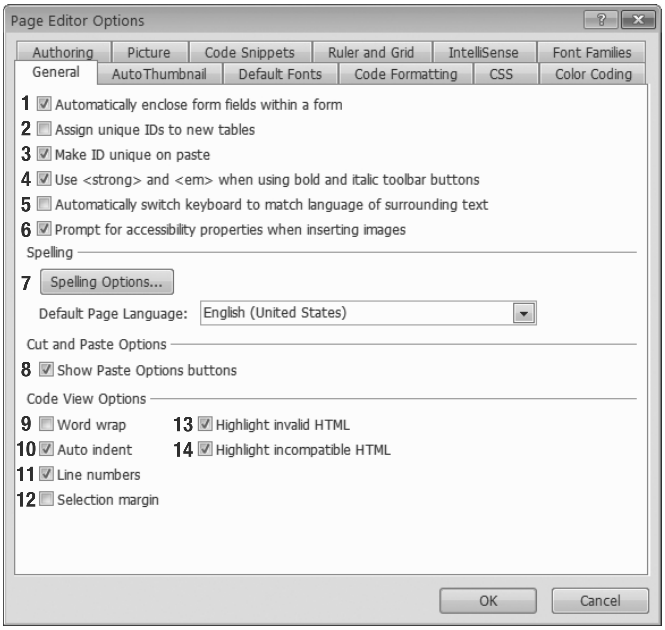
Figure 3-9. *Right-clicking Open With shows all editors associated with the file extension.*

Page Editor Options

Although I know all this configuration information might seem boring, there are bits that will make a difference in how your pages render. Here, you will be looking at the Page Editor Options. Some of the tabs will be quickly explained but have no real changes to be made; however, there are a few where you should pay attention to the settings. The ones you will be focusing on are General, CSS, and Font Families. I will touch on a few options under the Authoring, IntelliSense, and Code Formatting tabs as well. The other tabs will be grouped together.

The General Tab

We will quickly go through the defaults that can be selected under the General tab, which are numbered in Figure 3-10, and where appropriate I will explain the effects of the choices you might make.


Page Editor Options
Authoring | Picture | Code Snippets | Ruler and Grid | IntelliSense | Font Families
General | AutoThumbnail | Default Fonts | Code Formatting | CSS | Color Coding
OK
Cancel

1 ☑ Automatically enclose form fields within a form
2 ☐ Assign unique IDs to new tables
3 ☑ Make ID unique on paste
4 ☑ Use <strong> and <em> when using bold and italic toolbar buttons
5 ☐ Automatically switch keyboard to match language of surrounding text
6 ☑ Prompt for accessibility properties when inserting images

Spelling

7 Spelling Options...

Default Page Language: English (United States)

Cut and Paste Options

8 ☑ Show Paste Options buttons

Code View Options

9 ☐ Word wrap
10 ☑ Auto indent
11 ☑ Line numbers
12 ☐ Selection margin
13 ☑ Highlight invalid HTML
14 ☑ Highlight incompatible HTML

Figure 3-10. *Page Editor Options is where you set defaults for all of your Expression Web sites.*

The options on the General tab have been given numbers to make cross-referencing them with their explanations easier:

1. *Automatically enclose form fields within a form*: Leave this checked.

2. *Assign unique IDs to new tables*: If you use only data tables, checking this would be worthwhile, but if you use layout tables, you do not need and probably don't want an ID automatically assigned.

3. *Make ID unique on paste*: Since there can be only one ID per page, this is probably a good item to leave checked, but you might prefer to assign IDs yourself.

4. *Use <strong> and <em> when using bold and italic toolbar buttons*: You should leave this one checked, because `<strong>` and `<em>` have semantic meaning for screen reader users. They might also help search engines determine what to pay attention to on your page as long as you do not overuse them.

5. *Automatically switch keyboard to match language of surrounding text*: This is almost the opposite of the Site Settings Language tab you looked at in Chapter 1. If you use multiple languages on your site *and* are knowledgeable about the keyboard layout native to that language or country, then you might want to check this box. For all others, I recommend leaving it unchecked.

6. *Prompt for accessibility properties when inserting images*: This is new to Expression Web and something I am glad to see, since `alt` attributes are not only required for accessibility but also for all valid versions of XHTML and HTML 4.01. Do not remove the check from this check box, or it will haunt you.

7. *Spelling Options*: Clicking this button opens a dialog box with spelling options. I do not recommend selecting the check box to hide spelling errors, since errors that should have been caught by a spell-checker might cause your site to lose credibility.

8. *Show Paste Options buttons*: This option will allow you to decide whether to bring formatting when you are pasting text.

9. *Word wrap*: This is for the Code view, since text wraps automatically in Design view under HTML specifications. This setting is a personal preference; there is no right or wrong choice.

10. *Auto indent*: This option is also applied to the Code view and is another personal preference~~o~~ne that will make it easier for you to move from one section of the code to another. I recommend leaving it checked.

11. *Selection margin*: This adds a margin to the left side of your code to make it easier to work with blocks of text in Code view.

12. *Highlight invalid HTML*: I recommend keeping this checked, since it makes finding errors much easier. This is particularly useful when you are working on a web page that is not valid and moving it to current standards.

13. *Highlight incompatible HTML*: This will alert you to code that is not correct for your `doctype`, such as elements that improperly closed. I can think of no reason to disable this option.

The AutoThumbnail Tab

The next tab is AutoThumbnail, which allows you to make small thumbnails of your images and link them to the larger version in one operation. I recommend you make your thumbnail images in a dedicated graphics editor where you have full control of your optimization settings, but if you are just putting together a prototype, the thumbnail capabilities of Expression Web are adequate.

The Picture Tab

The defaults here affect only images you paste into your pages by copying and pasting them into an open page or by importing a Word or other file with images embedded in it. They do not affect images you import into your local website.

As with the AutoThumbnail tab, converting pictures from one format to another is best done in a full-fledged image editor. You do have limited control over the quality of your converted image from the Picture File Types dialog box using the File Type Settings, but this method of editing images is not the best one to use.

The Ruler and Grid Tab

There might be cases when you want to align elements on a page using a line or grid. The settings on this tab allow you to display grid lines using the spacing you choose. Remember that web pages are measured in pixels (px), which are the units of measurement for a display, not inches, centimeters, or points, which are units of measurement for print media and should not be used on the Web.

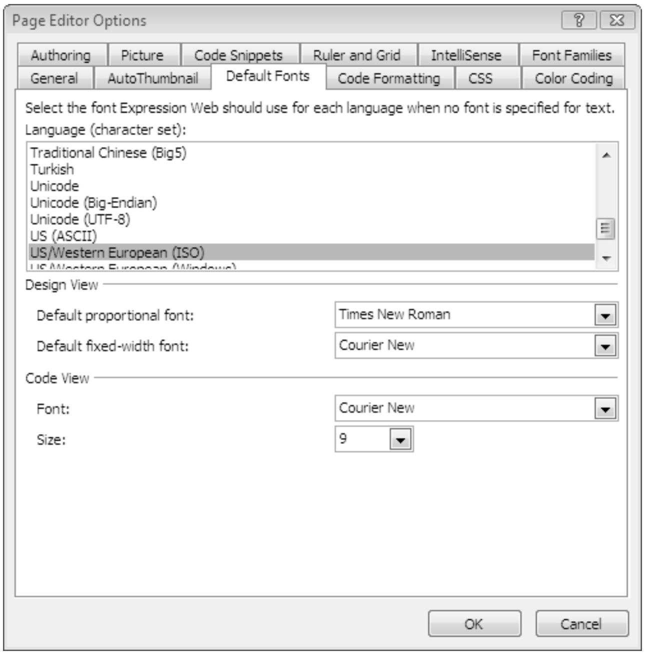
The Default Fonts Tab

The top half of the Default Fonts tab, shown in Figure 3-11, duplicates the Language tab of the Site Settings menu. You should set both of these to match one another. Expression Web follows the W3C recommendation of using UTF-8 as the default. Unfortunately, Expression Web adds hidden characters called Byte Order Marks (BOMs) to the top of each page, which can cause problems with some web servers and all PHP pages.

If you are using English or another western European language, I recommend selecting US/Western European (ISO), as shown in Figure 3-12, to prevent the BOM characters from being added to the page. If you are using Cyrillic, Arabic, Hebrew, or an Asian language, use the language-specific encoding, UTF-8 or UTF-16.

The bottom half of the Default Fonts tab will allow you to set the default fonts that are used inside Expression Web.

■ **Note** Changing the default from Times New Roman for Design view will not change the browser default when the page is viewed. Changes made here will affect pages only when viewed in Design view. I recommend leaving the defaults as they are set, since Times New Roman is the default for Internet Explorer and Firefox. To change the fonts used by the browser (and in Design view), you will need to use CSS.

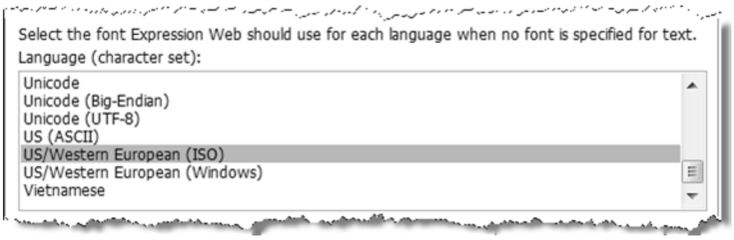


Select the font Expression Web should use for each language when no font is specified for text.
Language (character set):
Traditional Chinese (Big5)
Turkish
Unicode
Unicode (Big-Endian)
Unicode (UTF-8)
US (ASCII)
US/Western European (ISO)

Design View
Default proportional font: Times New Roman
Default fixed-width font: Courier New

Code View
Font: Courier New
Size: 9

Figure 3-11. *The character set tells the browser what set of alphanumeric and symbol characters will be used on the page.*



Select the font Expression Web should use for each language when no font is specified for text.
Language (character set):
Unicode
Unicode (Big-Endian)
Unicode (UTF-8)
US (ASCII)
US/Western European (ISO)
US/Western European (Windows)
Vietnamese

Figure 3-12. *In most cases, US/Western European (ISO) will be your best choice for the default Web Expression character set.*

If you want to change the font displayed in Design view, use a font family in your CSS body definition. That way, what you see in Design view will match what the visitor sees on the Web.

The Code view settings, on the other hand, will not affect how your code is displayed on the Web, and you should set those for your own comfort. Users with high-resolution screens might find 9-point text too small for comfort, while others might prefer a smaller font so they can see more at a time.

The Code Formatting Tab

There are a lot of settings you can change on the Code Formatting tab. The defaults will work well for most people, but if you have a code display preference, you can configure Expression

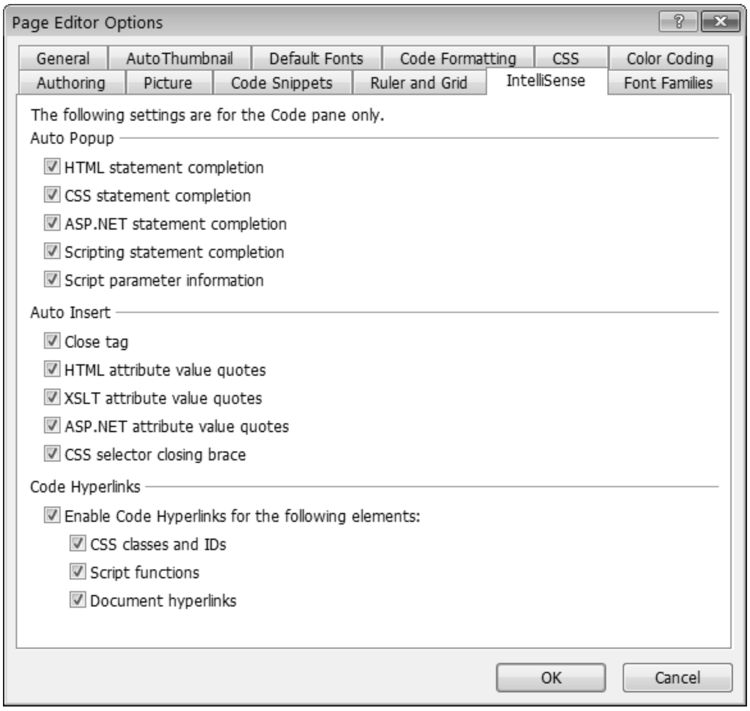
Web to match it. It is possible to get very granular in your display settings, down to specifying which attributes will contain a line break or a specific number of spaces and where those will be located. Unless you have a particularly strong preference, I suggest leaving the defaults.

The Color Coding Tab

Using color to distinguish elements, attributes, values, selectors, properties, and scripts will make it easier to spot problems caused by improper editing as well as missing elements. For most people, the defaults do not need to be changed, but if you prefer a specific color scheme, this is where you change it.

The IntelliSense Tab

Also referred to by some as code completion, IntelliSense is one of the most useful features Expression Web has when you are working in Code view. I recommend checking the boxes for all options, as shown in Figure 3-13, since selecting code as you type and putting in the proper quotes will prevent code errors from creeping in.



The following settings are for the Code pane only.

Auto Popup
- ☑ HTML statement completion
- ☑ CSS statement completion
- ☑ ASP.NET statement completion
- ☑ Scripting statement completion
- ☑ Script parameter information

Auto Insert
- ☑ Close tag
- ☑ HTML attribute value quotes
- ☑ XSLT attribute value quotes
- ☑ ASP.NET attribute value quotes
- ☑ CSS selector closing brace

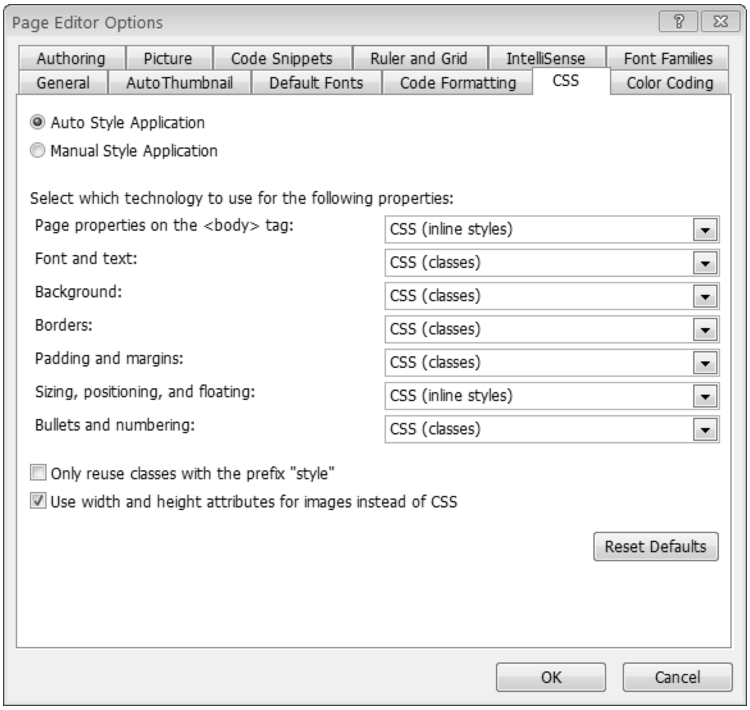
Code Hyperlinks
- ☑ Enable Code Hyperlinks for the following elements:
  - ☑ CSS classes and IDs
  - ☑ Script functions
  - ☑ Document hyperlinks

Figure 3-13. *IntelliSense can help you prevent typos.*

The only drawback to IntelliSense is that it does not support scripting languages other than ASP.NET 2.0. There might be add-ons available in the future to add support for other scripting languages.

The CSS Tab

It is not possible to discuss the CSS tab, shown in Figure 3-14, without discussing CSS. In this section, there are several exercises and sidebars to help you make sense of the options you will be choosing here. Consider this an appetizer before you start using the CSS-related task panes in creating your pages.



(● Auto Style Application)
Manual Style Application

Select which technology to use for the following properties:

Page properties on the <body> tag:	CSS (inline styles)
Font and text:	CSS (classes)
Background:	CSS (classes)
Borders:	CSS (classes)
Padding and margins:	CSS (classes)
Sizing, positioning, and floating:	CSS (inline styles)
Bullets and numbering:	CSS (classes)

☐ Only reuse classes with the prefix "style"
[✓ Use width and height attributes for images instead of CSS]

Figure 3-14. *A few changes to the CSS defaults will make your work flow better.*

CSS BASICS

Recall that CSS stands for Cascading Style Sheets; these stylesheets contain a series of styles and definitions (we will be working with CSS more extensively in Chapter 5). Each style consists of three components:

- *Selector*: This is typically an HTML element, a class, or an ID.
- *Property*: This is the presentation item you are trying to define.
- *Value*: This is the property definition. In the following examples, you will see one of each of the types of selections and an example of its use.

The general syntax of a CSS style follows:

```
selector {property1: value;}
```

Continues

You can have more than one property in a style:

```
selector
{
  property1: value;
  property2: value;
}
```

Selectors

There are three types of selectors:

- *HTML element*: The style definition will affect all instances of the specified element on pages that use this style; for example, to set the color of all `<h3>` elements, you would use this definition:

  ```
  h3 {color: #666666;}
  ```

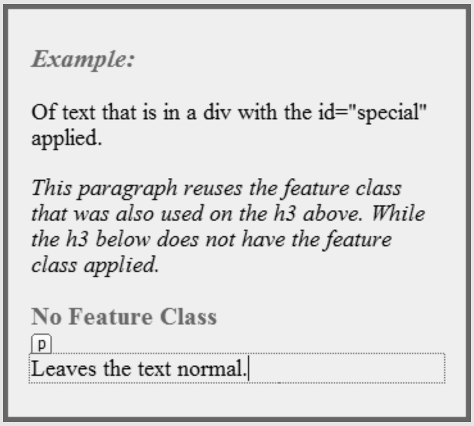
- *Class*: This style will apply to any element that has the class applied and might be used more than once on a page:

  ```
  .feature {font-style: italic;}
  ```

- *ID*: Styles assigned with an ID can be used only once per page, so you should reserve them for structural elements:

  ```
  #special {
    background-color: #efefef;
    border: medium #666666 solid;
    padding: .5em 1em;
    margin: 1em 6em;
  }
  ```

The following image shows what the browser would display with a `<div>` with `id="special"`, an `<h3>` defined as in the previous code snippet, and the `feature` class applied to the paragraph element.



Example:

Of text that is in a div with the id="special" applied.

This paragraph reuses the feature class that was also used on the h3 above. While the h3 below does not have the feature class applied.

No Feature Class

p

Leaves the text normal.

A `<div>` is a block-level element with no semantic meaning that is normally used for sections. Think "page divisions." Though not used in this example, a `<span>` is an inline element used as a hook to apply a style or to provide for a tooltip or script ID.

In all, six property elements are used in this sidebar's example:

- `color`
- `background-color`
- `border`
- `font-style`
- `margin`
- `padding`

CSS Shorthand

Expression Web gives you the option to write each CSS property individually or to use what is called CSS shorthand, where multiple properties and values are written on one line. The `border` style in this sidebar uses shorthand. There are three separate values for border: `size`, `type`, and `color`. Without any sort of shorthand, that one line of CSS would be written as follows:

```
border-top-style: solid;
border-top-width: medium;
border-top-color: #666666 ;
border-right-style: solid;
border-right-width: medium;
border-right-color: #666666 ;
border-bottom-style: solid;
border-bottom-width: medium;
border-bottom-color: #666666 ;
border-left-style: solid;
border-left-width: medium;
border-left-color: #666666 ;
```

While some people prefer the expanded format while learning CSS, using shorthand will make your stylesheet more compact, faster-loading, and, in most cases, easier to read and edit.

We will go into the specifics of properties and values later in the book when you are working with the CSS task panes.

In the CSS tab, you will leave the Auto Style Application setting selected, since this is the best choice for most users. This setting tells Expression Web to look at what you are doing and apply any matching style linked to your page. If there are no matching styles, then Expression Web will create a new style to match your font, border, background, layout, and so forth.

By default, the majority of the CSS that Expression Web writes will be written as styles in the `<head>` section of the page. However, there are two types of styles that are written inline, as shown in Figure 3-14. My recommendation is that you change those two settings.

All of the options should be set to "CSS (classes)" with the exception of the "Page properties on the <body> tag" option, which does not offer the option to use classes since it is an HTML element. Otherwise, you lose many of the benefits of stylesheets, that is, the ability to make one change that affects all the pages on your site. The two settings Microsoft has chosen to make inline follow:

- The first inline setting is the "Page properties on the <body> tag" option, where your options are the "CSS (inline styles)" option and the "CSS (rules)" option. Since a `<body>` tag will be used only once on the page, you would not need a class. In some circumstances, you might want to use an ID to select a different default for certain pages in your site when an ID would be appropriate to do that, but for your normal defaults, redefining the `<body>` element is the best choice.
- The other option that is set to inline styles is the "Sizing, positioning, and floating" option.

Leave the defaults, the "CSS (inline styles)" setting, until you are directed to change them for the exercises in this chapter.

TYPES OF STYLES

Styles can be written and applied using one of three methods:

- *Inline styles*: These styles apply only at the location where the style is written and are similar to the old HTML 3.2 `<font>` element:

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

- *Style block*: The style is written in the `<head>` section of the page and will apply to every selector on the page:

```
<style type="text-css">
p  {
    font-family: "Times New Roman", Times, serif;
    color: navy;
}
</style>
```

The style is applied to every `<p>` element on the page by default.

- *External stylesheet*: Styles are in a separate document with the `.css` extension attached to the page either by using a relative link

```
<link rel="stylesheet" type="text/css" href=" site.css" />
```

or by importing it into the page:

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

  <!--
  @import url("site.css");
  -->
</style>
```

Using external stylesheets will keep the style definitions the same throughout your site. `<head>` section styles can be used to change the appearance on just one page. Inline styles will allow you to override both the `<head>` section style and any style in an external stylesheet for one instance on your page. Since it is more difficult to locate and change inline styles on your pages, they should be used sparingly.

Creating Styles

Although you will be creating styles in this book using the new styles method, you can also create styles by using the buttons on the Format toolbar. When you use the Format toolbar to create a style, Expression Web will automatically name the styles with class names of `.style1`, `.style2`, `.style3`, and so forth. The check box you see near the bottom of the CSS tab lets you choose whether to limit the automatic application of styles to those created by Expression Web or to include those you create or rename. My recommendation is to leave this box checked.

To avoid creating separate classes for each image on your page, leave the check in the "Use width and height attributes for images instead of CSS" box. This will also help your page load properly. Have you have ever visited a web page where the page rearranged itself after all the images loaded? It did that because the height and width of the image were not in the HTML, and the browser did not know how much space to reserve when laying out the page. In that case, the browser has to reflow the page, because the image and stylesheet usually load after the HTML. Exercise 3-1 shows the effect of changing the CSS Tab Styles setting, Exercise 3-2 shows the effect of changing the Body setting to Rule, and Exercise 3-3 shows how to change styles for the positioning of your website.

Exercise 3-1. Seeing the Effects of Changing the CSS Tab Styles Setting

In this exercise, you will compare the effects of the default settings of Expression Web with the recommended changes to the effects of the CSS styles that are written as you create them in Expression Web.

1. Download the `exercise.txt` file from `http://foundationsofexpressionweb.com/exercises/chapter3/ch3example1.txt` or the `chapter3.zip` file containing all the Chapter 3 exercise files, and save a copy to your project web. Or type the following lines into Code view and save them as `exercise.html`:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

```
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
  <title>CSS Test</title>
</head>
<body>
  <div>
      <h1>CSS Test Page</h1>
  </div>
  <div>
    <p>Menu Section</p>
    <ul>
      <li><a href="#">Link 1</a></li>
      <li><a href="#">Link 2</a></li>
      <li><a href="#">Link 3</a></li>
      <li><a href="#">Link 4</a></li>
    </ul>
  </div>
  <div>
    <h2>Content</h2>
    <p>
      We will be using this basic page for many of our exercises so you should
      archive a copy of this file to use for your exercises in exactly this
      format.
    </p>
    <p>If you wish you can add more paragraphs to the content section and
bring in some images from your computer. While I could provide you using
images (there are some images the zip file) and more content but using
images and content of your choice will make the exercises more realistic.</p>
    <h3>Next Bit</h3>
    <p>
      Some more paragraphs and maybe even breaking up content into more lists
      will appear in this section.
    </p>
  </div>
  <div>Footer, copyright and stuff</div>
</body>
</html>
```

■ **Note** You can insert an image of your choice using File ➤ Insert ➤ Picture ➤ From File. If you downloaded this file from the website, you might use one of the images in the zip file.

You will be using this file in multiple exercises, so make sure you make a new copy before you begin each of the exercises that follow.

The `exercise.html` page consists of four sections, or `<div>`s, with content in each `<div>`. Before you continue with the exercises, save a copy of your `exercise.txt` or `exercise.html` file as `exercise1a.html`. Please use Split view for all the exercises in this book unless directed to do otherwise.

The Effects of Using Inline Body Styles

The best way to understand why changing CSS defaults will help you make your website easier to maintain is to first see what the defaults do by creating styles using those defaults.

1. Right-click anywhere in Design view and select Page Properties, or select File ➤ Properties to launch the Page Properties dialog box.
2. Set the page background color and text color from the drop-down menu on the Formatting tab (see Figure 3-15). Make sure that you actually select a body color and a text color. Contrary to what many people believe, not all browsers use white as their default background color. Some versions of Netscape, for instance, use a light gray. You might optionally add a background image.
3. Click the OK button.

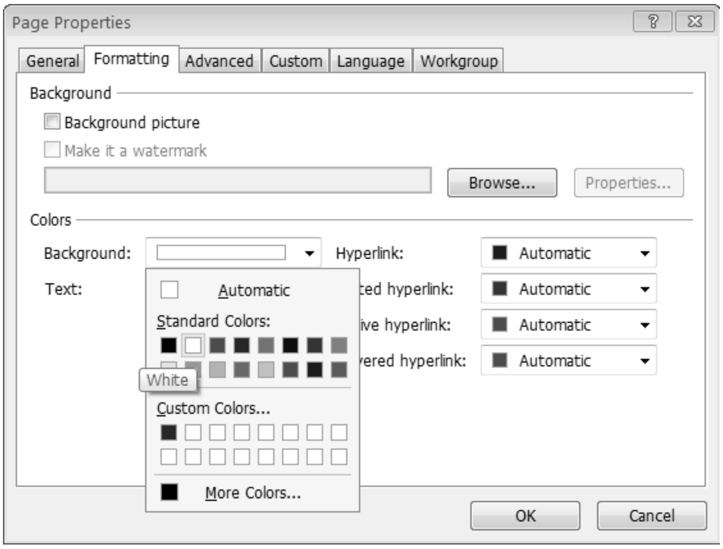


Figure 3-15. *To quickly set basic page-level formatting from the Formatting tab, select your background and text colors.*

4. Now look at the `<body>` element in the code section of Split view.

Note Select the `<body>` element from the Quick Link bar. The `<body>` element will be highlighted in Code view.

You should see an inline style that looks like this:

```
<body style="color: #000000; background-color: #FFFFFF">
```

5. Next, return to the Page Properties dialog box, and choose "Hyperlink state colors." You should see something like the following code written by Expression Web:

```
<style type="text/css">
a {
  color: #000080;
}
a:visited {
  color: #000080;
}
a:hover {
  color: #CC0000;
}
a:active {
  color: #000080;
}
</style>
```

These are the styles written in the `<head>` section of the document and will apply to all links and their states, which are called pseudo-classes. Preview your `exercise.html` page in the browser of your choice, and move your mouse over the links to see the rollover effects caused by the `hover` pseudo-class.

6. Select the `<body>` element again, and from the Format bar, click the Borders button to add a border to your web page, as shown in Figure 3-16.

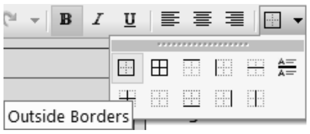


Figure 3-16. *Select Outside Borders from the Format bar's Borders drop-down.*

You will see that you have created a style instead of adding it to the inline style in the `<body>` element:

```
.style1 {
  border-style: solid;
  border-width: 1px;
}
</style>
</head>
<body style="color: #000000; background-color: #ffffff;">
```

PSEUDO-CLASSES

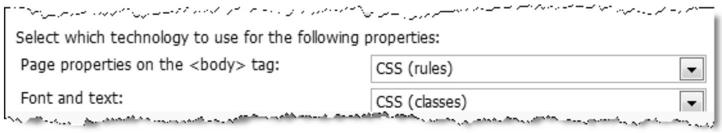
Pseudo-classes apply a style based on an element's state such as `link`, `visited`, `hover`, and `active`. These are what gives your links a different color when you have visited the page to which the link leads. The pseudo-class of `hover` provides the rollover effect when you mouse over the link when you change the color, background color, or image; add or remove an underline; or add other touches. For your CSS-based rollovers to work, your pseudo-classes must be in the order listed; some people use the "LoVe HAte" mnemonic as a memory aid:

- `link`
- `visited`
- `hover`
- `active`

Exercise 3-2. Seeing the Effect of Changing Body to Rule

Expression Web uses CSS (rules) when referring to using the HTML element as your selector. In this exercise, you'll see how styles are written when the `<body>` element is used as the selector in the CSS tab.

1. Begin by saving another copy of `exercise.txt` as `exercise1b.html`.
2. With the new page open in Expression Web, go to Tools ➤ Page Editor Options ➤ CSS tab, and change the body's setting to the "CSS (rules)" option, as shown in Figure 3-17.



Select which technology to use for the following properties:
Page properties on the <body> tag: CSS (rules)
Font and text: CSS (classes)

Figure 3-17. *There should be no "CSS (inline)" in your CSS tab settings now.*

3. Next, add the same page properties and border to the `<body>` element.

Note When applying the `border` property to the `<body>` element, there is a bug. What you see is a body style with the background color and text color but a separate `style1` for the border, which was applied to a `<div>` instead of to the `<body>` element. In the first exercise, the style was correctly created and applied to the `<body>` element, but in the second exercise no style should have been created, and the `border` property with its associated values should have been added to the `<body>` element.

---

4. Compare the code between the `exercise1a.html` and `exercise1b.html`. What you should see follows:

```
body {
  color: #333333;
  background-color: #FFFFFF;
  border-style: solid;
  border-width: 1px;
}
```

The code you should have seen could easily be transferred to an external stylesheet to use on all subsequent pages to make maintaining your site easier. This is the code that would be created if you used one of the CSS task panes or the Styles dialog box to create your CSS instead of using a combination of the Page Properties settings and the Format toolbar. While the Page Properties settings are useful for basic, quick formatting of your page styles, it is best not to use the Format toolbar on the `<body>` element.

Exercise 3-3. Changing Styles for Positioning

Next, you will see the effect of making the recommended change of "Sizing, positioning, and floating" from "CSS (inline)" to "CSS (classes)" for a single page, while leaving the default of the "CSS(inline)" setting. Perform the following steps:

1. Select `h1` in Design view. Then, from the Quick Tag bar, select the `<div>` element that precedes it in the list, which will show the boundaries of the `<div>` element in the design windows.
2. Drag the bottom of the highlighted area (using the bottom-center round bullet shown Figure 3-18) so that the `<div>` is approximately 75px high.

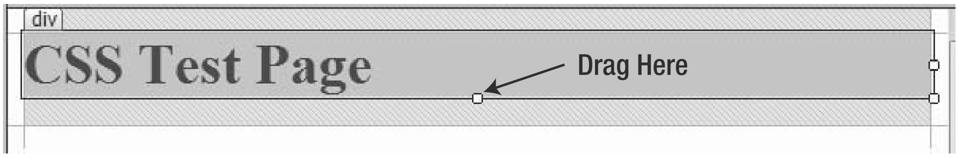
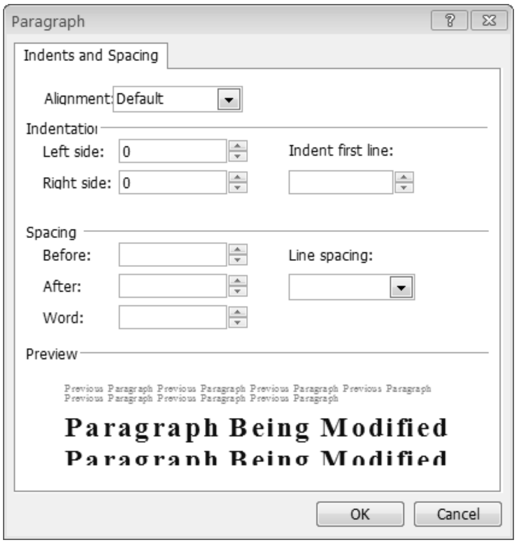


Figure 3-18. *Make sure that you drag the round bullet to change the height or width of a highlighted section.*

3. Select the `h1` again, and apply a font family and color from the Format toolbar. You might also change the text size if you want; for now, use only relative sizes such as xx-large or small.
4. Right-click or use the drop-down arrow on `h1.style1` in the Quick Tag bar.
5. In this dialog box, select Tag Properties to launch the Paragraph dialog box shown in Figure 3-19, and add a left-side indentation.



Alignment: Default

Indentation

Left side: 0

Right side: 0

Indent first line:

Spacing

Before:

After:

Word:

Line spacing:

Preview

Previous Paragraph Previous Paragraph Previous Paragraph Previous Paragraph Previous Paragraph Previous Paragraph Previous Paragraph

Paragraph Being Modified Paragraph Being Modified

Figure 3-19. *By default, the indentation is 0px. You can type in the number of pixels you want to indent the h1 element.*

6. Go to the next `<div>`, click somewhere inside it, and choose the `<div>` element from the Quick Tag bar again.
7. Right-click the arrow to launch the drop-down box for the `<div>`, select Positioning and then “position: absolute” (see Figure 3-20).

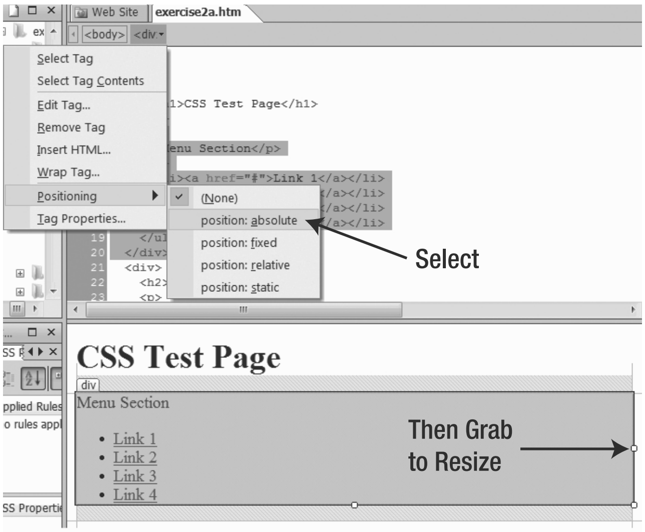


Figure 3-20. *Choose the “position: absolute” option*

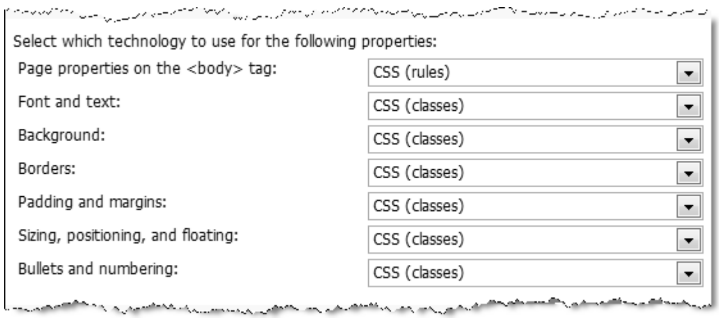
8. Next, in Design view, drag the right side of the positioning box until the width is 170px wide using the right-side dot.

■**Note** It is very easy to set both a height and a width when you are dragging an element to size it. If you create a height or width you do not want specified, simply delete it in Code view.

---

9. Move to the main content `<div>`, and repeat steps 6 to 8 to apply absolute positioning to your main content `<div>`. This time, drag the right side to 190px, or whatever looks to you to be sufficient space for the left-side menu.
10. For now, move the `<div>` that contains your footer inside the container `<div>`, and change the font size to a smaller font size using the Format toolbar.

    This is starting to look like a real page, isn't it? I do not suggest you use this layout as it is written by Expression Web, since the way you created it does not give you the most robust layout (and you will be creating robust layouts using CSS later in this book). As you can see, using inline styles would make it hard to transfer this page to an external stylesheet, which is the point of this exercise.
11. Save this page as `exercise2a.html`.
12. Now, select Tools ➤ Page Editor Options ➤ CSS tab, and change all of the remaining properties to "CSS (classes)" (you should have already changed the body styles to "CSS (rules)"). Your CSS tab options should now look like Figure 3-21.



Select which technology to use for the following properties:

Page properties on the <body> tag:	CSS (rules)
Font and text:	CSS (classes)
Background:	CSS (classes)
Borders:	CSS (classes)
Padding and margins:	CSS (classes)
Sizing, positioning, and floating:	CSS (classes)
Bullets and numbering:	CSS (classes)

Figure 3-21. *Your CSS tab style settings should look be all "CSS (classes)" or "CSS (rules)" after this exercise.*

13. Now that you have changed the sizing, positioning, and floating from inline styles to classes, repeat this exercise with a fresh copy of your `exercise.html` file saved as `exercise2b.html`, and compare the resulting code to `exercise2a.html`. When you are creating multiple web pages with the same layout, having all the presentation elements in one `<style>` block will make site maintenance much easier, because you need to make style changes in only one place, and they will affect all pages linked to those styles.

The Font Families Tab

If you use the Font drop-down on the Format menu in the earlier exercise, you will see that the first choices on the drop-down list are font groups, as shown in Figure 3-22.

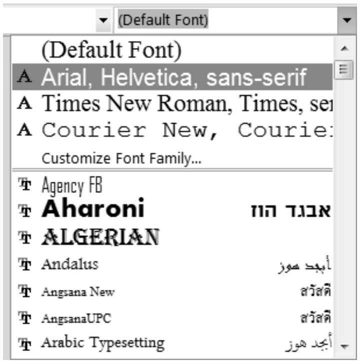


Figure 3-22. *Font drop-down choices*

Tip You should always use a font family group and not just a single font.

The Font Families tab allows you to add more font family groups to the drop-down list. Whether you access this tab from Tools ➤ Page Editor Options ➤ Font Families or by using the Customize Font Family link on the Format drop-down, you will end up at the tab shown in Figure 3-23.

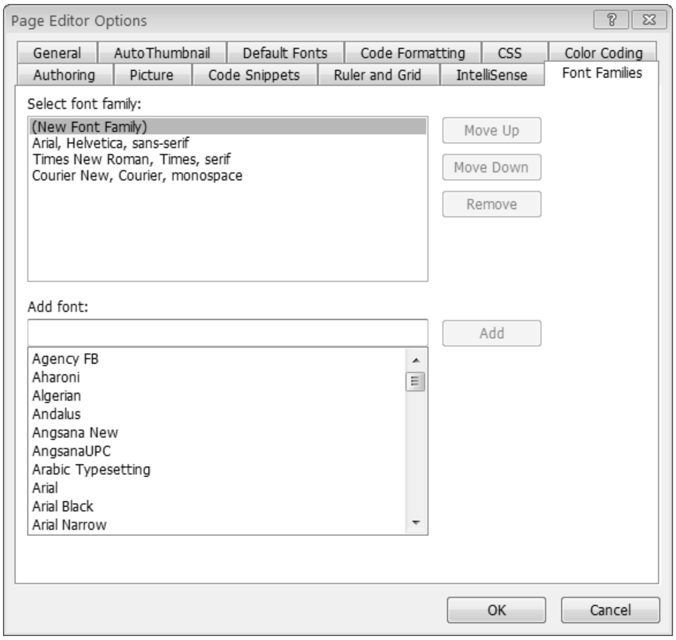
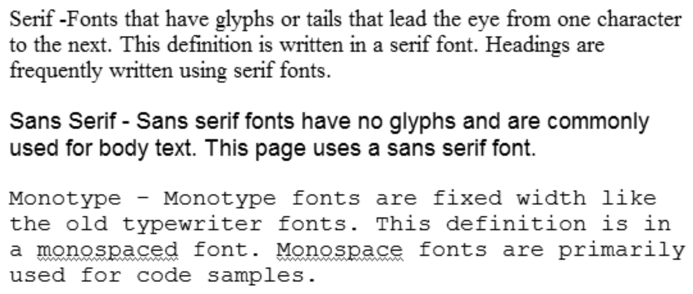


Figure 3-23. *This is where you create your own font family groups.*

A Word About Fonts

Before you begin creating custom font groups, here is a little background information on the fonts for the Web. There are three major families of fonts that are commonly used on the Web; examples in Figure 3-24 use the default font for each font family on Windows computers.



Serif -Fonts that have glyphs or tails that lead the eye from one character to the next. This definition is written in a serif font. Headings are frequently written using serif fonts.

Sans Serif - Sans serif fonts have no glyphs and are commonly used for body text. This page uses a sans serif font.

Monotype - Monotype fonts are fixed width like the old typewriter fonts. This definition is in a monospaced font. Monospace fonts are primarily used for code samples.

Figure 3-24. *These are the three most commonly used font families on the Web.*

■**Note** *Monospace* and *monotype* are used interchangeably depending on whether the person comes from a print background.

There are also two other font family types, both of which are rarely used on the Web: cursive and fantasy. One reason you rarely see either of these on the Web is there is not a standard default font for either type. Some browsers will default to a serif or sans serif font, which means browser testing is critical if you are considering using either of them.

The browser default is usually a serif font such as Times New Roman at font size medium. Over the years, there have been many studies on the use of fonts for print. The consensus has been that serif fonts like Times New Roman are easiest for the majority of people to read at speed. This has also been found to be true on the screen with one major caveat—font size. Because computer screens are all lower resolution than print, the glyphs of serif fonts break up at small sizes.

For this reason, you should not use serif fonts below the equivalent of 13px, CSS size small, or .95em. If you use small font sizes on your website, you should use sans serif fonts. Above the equivalent of 18px, serif fonts generally look best. In-between sizes are a matter of personal preference.

■**Note** An *em* on the Web is the size of character according to the visitor's computer settings.

Font families should always be specified in a group starting with the font you would most like the visitor to see on a Windows machine or a Mac, followed by commonly available alternatives and ending in a generic font family. It is important to realize that visitors must have the font installed on their computer for the browser to use it. That means fancy and/or unusual fonts should be reserved for use in logos or other graphics.

Common font-family Values

Common values for the CSS `font-family` setting follow, split up by font family type.

These are the serif fonts:

- `font-family: "Times New Roman", Times, serif;`: The first parameter is a Windows font, the second is a Mac font, and the last is a generic font family and will be used if neither of the other two fonts is available. This is the font family group for serif that ships with Expression Web.
- `font-family: Georgia, "Times New Roman", Times, serif;`: Georgia is a serif font developed for the Web by Microsoft. It is considered by many to be more attractive and easier to read than the various Times fonts. The original Times font was created as a compressed but easy-to-read font for newspapers. Mac OS X 10.4+ users must have Microsoft Office installed to use Georgia.

■ **Note** If a font name consists of more than one word such as Times New Roman, the font name must be enclosed in quotation marks, for example, `"Times New Roman"`, `"Monotype Corsiva"`, and `"Courier New"`.

These are the sans serif fonts:

- `Arial, Helvetica, sans-serif`: This is probably the most commonly used `font-family` declaration on the Web, whether it starts with Helvetica or Arial. This is the default sans serif font group in Expression Web.

  Most people begin this group with Arial, but I prefer to begin with Helvetica, since it is a Mac default that many Mac users believe is more attractive than Arial, which is also a common Mac font. This is one case where the Windows and Mac fonts bear the same name.
- `Tahoma, Helvetica, Arial, sans-serif`: Like Georgia, Tahoma is a Microsoft font developed for use on computer screens.
- `Verdana, Helvetica, Arial, sans-serif`: Verdana is another font developed to be easy to read on the screen and is a favorite of designers. However, I do not recommend using it, since Microsoft removed the ability to download its web fonts for Mac and Linux users. The reason I recommend not using it should be clear from the relative size differences between the two following entries. The first uses Verdana, and the second uses Arial, which is the font most commonly used by web browsers as the `sans-serif` default:

  Now is the time for all good men to come to the aid of their country.

  Now is the time for all good men to come to the aid of their country.

These are the monospace fonts:

- `"Courier New", Courier, monospace`: As with Times New Roman, Courier New is the Windows version, and Courier is the Mac version. This is really the only commonly used monospace declaration, and it is the one that ships with Expression Web.

Font Guidance

Let the following guidelines help you make sure the fonts you choose are the fonts your visitor sees:

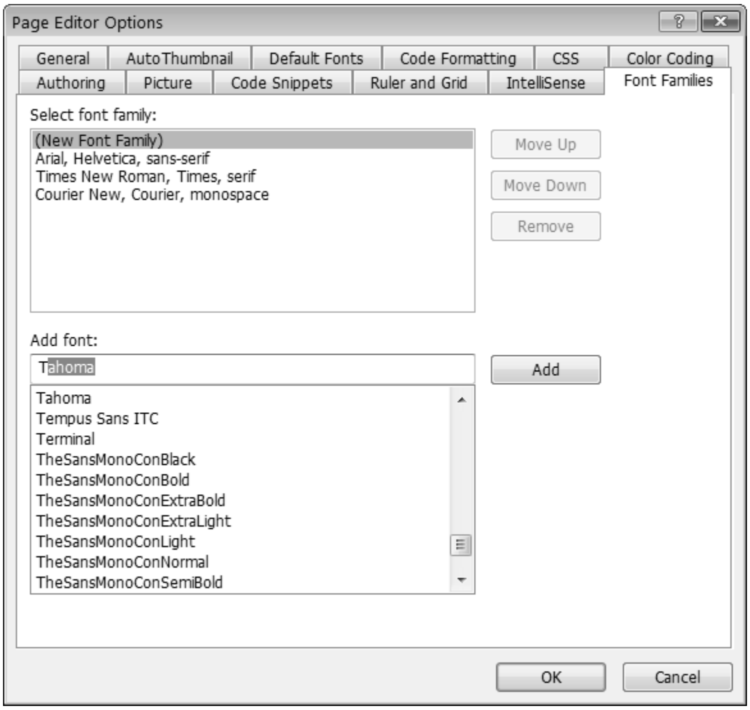
- Best practice is to list multiple choices for the font you want the visitor to see.
- Separate the fonts with commas.
- If a font name is more than one word, you must use quotes around the name. `"Times New Roman"` and `"Courier New"` are examples of multiple-word font names.
- Always include a generic font family as the last entry in your font value list.
- Remember that just because you have a font installed does not mean your visitors will have that font and see it when they visit your page.

Exercise 3-4 shows how to create a font family group.

Exercise 3-4. Creating a Font Family Group

In this exercise, you will create a font family group starting with a common but not universal font and ending with a generic font family.

1. To begin a new group, select New Font Family from the “Select font family” list box.
2. Next, move down to the “Add font” box, and either begin typing the name of the font or scroll down the list. As you type, the box will perform an incremental search through the fonts installed on your computer.
3. Scroll down the list box until you find the name of the font you want to use, as shown in Figure 3-25.
4. Once you have chosen your font, click the Add button. You must add each font before you can add the next font for your family group.
5. Make sure the font group you want to add more fonts to is selected before you add the next font.
6. Repeat steps 2 through 5 for each font you want to add.
7. When your new font list is completed, click one of the other font families to lock your new group in place. If you used the OK button, you would have closed the Page Editor Options, so you will need to reopen it to make any additional changes.



Select font family:

(New Font Family)
Arial, Helvetica, sans-serif
Times New Roman, Times, serif
Courier New, Courier, monospace

Add font:

Tahoma

Tahoma
Tempus Sans ITC
Terminal
TheSansMonoConBlack
TheSansMonoConBold
TheSansMonoConExtraBold
TheSansMonoConExtraLight
TheSansMonoConLight
TheSansMonoConNormal
TheSansMonoConSemiBold

Figure 3-25. *Don't forget to click Add after you select each font.*

Note If you want to include a font that is not installed in your computer, such as the Mac Helvetica font, you will need to enter the name. Make sure you have the font name spelled correctly.

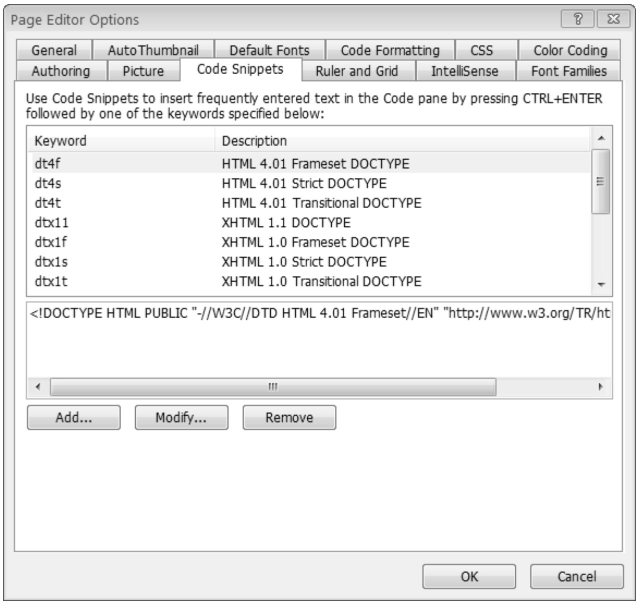
The Code Snippets Tab

The Code Snippets tab contains a handy collection of items you can quickly insert into Code view to perform specific functions or to replace one bit of code with another. As I've previously explained, Expression Web uses UFT-8as its default character set. When you created your first HTML page, UTF-8was applied by default. Despite the changes you are making to the default language and character set settings, if you use one of the website or page templates that ships with Expression Web, those pages will have UTF-8as the character set. One of the easiest ways to change from UTF-8to ISO-889-1 is by using a code snippet. Since Expression Web does not ship with a meta-character-set snippet, you will create one in Exercise 3-5.

Exercise 3-5. Creating a Code Snippet

If you find yourself using the same line of code frequently, you should turn it into a code snippet using the steps in this exercise.

1. To create a code snippet that will allow you to easily replace an existing UTF-8 character set, use Tools ➤ Options to open the Page Editor Options, and select the Code Snippets tab.
2. Use the Add button shown in Figure 3-26 to begin your new snippet.



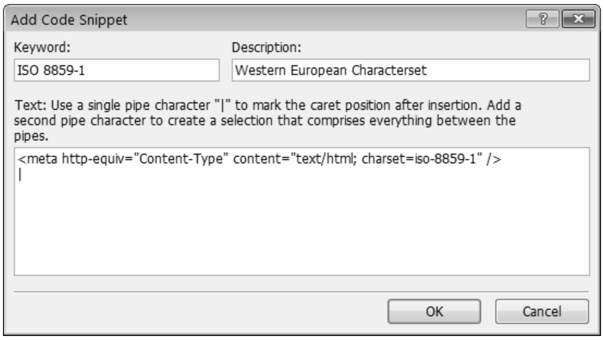
Use Code Snippets to insert frequently entered text in the Code pane by pressing CTRL+ENTER followed by one of the keywords specified below:

Keyword	Description
dt4f	HTML 4.01 Frameset DOCTYPE
dt4s	HTML 4.01 Strict DOCTYPE
dt4t	HTML 4.01 Transitional DOCTYPE
dtx11	XHTML 1.1 DOCTYPE
dtx1f	XHTML 1.0 Frameset DOCTYPE
dtx1s	XHTML 1.0 Strict DOCTYPE
dtx1t	XHTML 1.0 Transitional DOCTYPE

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Frameset//EN" "http://www.w3.org/TR/ht
```

Figure 3-26. `doctype`*s make up the majority of the code snippets that ship with Expression Web.*

3. When the Add Code Snippet dialog box shown in Figure 3-27 appears, fill in the text boxes with the appropriate entries.



Keyword: ISO 8859-1

Description: Western European Characterset

Text: Use a single pipe character "|" to mark the caret position after insertion. Add a second pipe character to create a selection that comprises everything between the pipes.

```
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
|
```

Figure 3-27. *Make sure all the text boxes are filled in before you click OK.*

4. In the Keyword box, type the name you want to appear in the code snippet drop-down box. I used the ISO number.
5. In the Description box, provide a description of what the code snippet will do.
6. In the Text box, type **<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />** with a pipe [|] before and after the code snippet to mark where the text begins and where the cursor should be after insertion.
7. When you are satisfied, click the OK button.

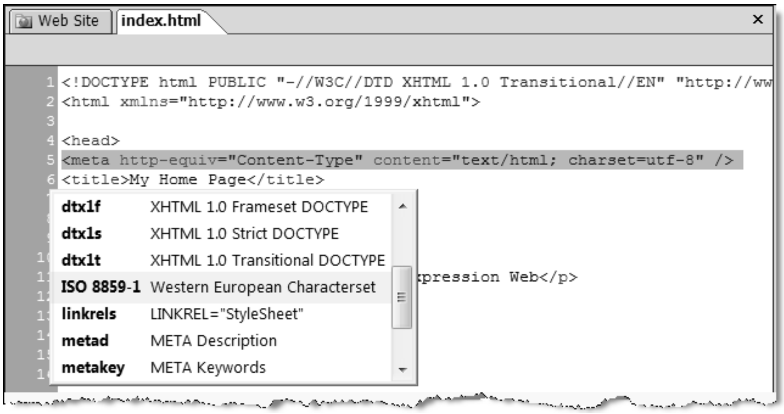
Congratulations, you have successfully created your first code snippet. You will be creating and using more code snippets in Chapter 10.

Now that you have created a code snippet, it is time to use the code snippet you just created (see Exercise 3-6).

Exercise 3-6. Using Code Snippets

Before you can insert a code snippet, you must be in Code view or the code half of Split view. Begin with opening the `index.html` page you created in Chapter 2.

1. Press Ctrl+Enter at the location you want to insert the code snippet, which launches the code snippet drop-down, as shown in Figure 3-28. In this case, you want to replace the UTF-8 character set `<meta>` tag with an ISO-8859-1 `<meta>` tag.



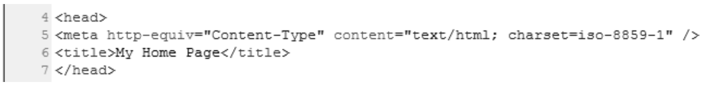
```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://ww
<html xmlns="http://www.w3.org/1999/xhtml">

<head>
<meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
<title>My Home Page</title>
```

dtx1f	XHTML 1.0 Frameset DOCTYPE
dtx1s	XHTML 1.0 Strict DOCTYPE
dtx1t	XHTML 1.0 Transitional DOCTYPE
ISO 8859-1	Western European Characterset
linkrels	LINKREL="StyleSheet"
metad	META Description
metakey	META Keywords

Figure 3-28. *Right-click in Code view to view available code snippets.*

2. Scroll down until you see the code snippet you want to use highlighted, as shown in Figure 3-28, and double-click it. The drop-down will close, and your code snippet will be inserted. The result will look like Figure 3-29.



```
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>My Home Page</title>
</head>
```

Figure 3-29. *Using a code snippet to insert code that you will use frequently helps prevent syntax or typing errors.*

3. Make sure you place the code snippet where you want it to appear. The character set `<meta>` tag should be the first item after the opening `<head>` element, as shown earlier in Figure 3-29.

Using the Site Menu

The first group of options under the Site menu, shown in Figure 3-30, provides you with information about your site that will open in the Web Site tab in the main Document window or in a reports window under the main document.

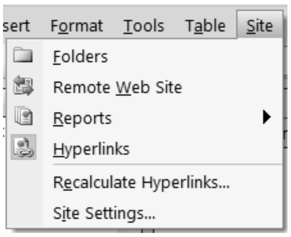


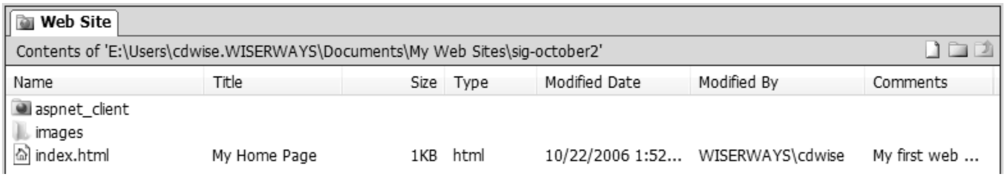
Figure 3-30. *The Site menu provides you with quick links to your site management tools.*

Folders

The Folders item gives you a seven-column, detailed view, as shown in Figure 3-31; the columns follow:

- *Name*: This is the file name, the same as in the folder view.
- *Title*: This is your HTML `<title>` element, and it is particularly useful in sites without good, strong file names.
- *Size*: This shows how big the file is, exclusive of images or included content.
- *Type*: Example types are `.html`, `.aspx`, `.css`, image types, and so forth. This column is useful for sorting files.
- *Modified Date*: This column tells when the file was last updated.

- *Modified By*: If you are working in a team environment, this can be very important, as long as you know the user ID of the people who work on the Web.
- *Comments*: This column contains notes you can add via the right-click Properties Summary tab. This is one way of making notes available for yourself or team members without putting comments in your HTML.



Contents of 'E:\Users\cdwise.WISERWAYS\Documents\My Web Sites\sig-october2'

Name	Title	Size	Type	Modified Date	Modified By	Comments
aspnet_client						
images						
index.html	My Home Page	1KB	html	10/22/2006 1:52...	WISERWAYS\cdwise	My first web ...

Figure 3-31. *The Folders option in the Site menu provides you with details about your pages.*

Remote Web Site

This item opens the publishing interface in the Document window.

Reports

This does exactly what it says and provides you with a list of different reports and sorting options such as the "Recently added pages" option. These reports will be more useful when you have a multipage site. In Chapter 12, I will be covering the available reports in more detail.

Hyperlinks

This item creates a link map of pages linked from the home page, which can be expanded to display a link diagram of your site.

Recalculate Hyperlinks

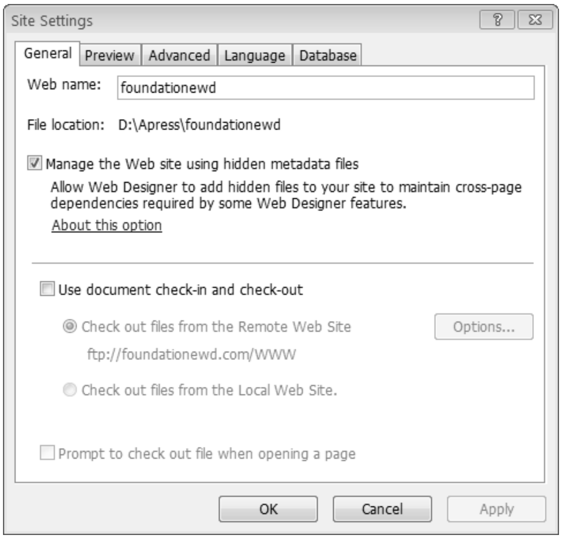
This item will update the metainformation if your local or remote site information gets corrupted.

Site Settings

The settings chosen in this section can have a major impact on site management functions and site testing.

The General Tab

Unless you are working in a team environment, the default settings on the General tab, shown in Figure 3-32, should be left alone. If you disable metafiles, your links will no longer be updated when you move files, and changes to your DWT (see Chapter 11) will not propagate to pages that use the DWT when changes are made.



Web name: foundationewd

File location: D:\Apress\foundationewd

[✓ Manage the Web site using hidden metadata files]

Allow Web Designer to add hidden files to your site to maintain cross-page dependencies required by some Web Designer features.

Use document check-in and check-out

[✓ Check out files from the Remote Web Site]

ftp://foundationewd.com/WWW

Check out files from the Local Web Site.

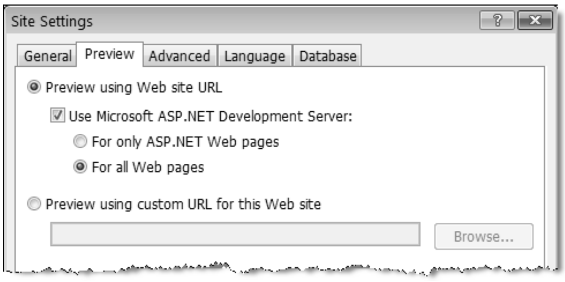
Prompt to check out file when opening a page

Figure 3-32. *Leave the check mark in the "Mange the Web site using hidden meta data files" box, or you will lose access to most of the site management features of Expression Web.*

The Preview Tab

In Chapter 2, you learned the importance of previewing web pages in real web browsers. Expression Web has a small built-in web server called Cassini used for testing ASP.NET 2.0 server behaviors. On the Preview tab, shown in Figure 3-33, you have the option to preview all pages using Cassini web server, which is what I recommend.

■ **Note** The Cassini web server does not support any server-side language other than ASP.NET 2.0. If you are using another server-side language, such as PHP or Classic ASP, you will not be able to see the output of your server-side code without publishing your pages.



[✓ Preview using Web site URL]

[✓ Use Microsoft ASP.NET Development Server:]

For only ASP.NET Web pages

[✓ For all Web pages]

Preview using custom URL for this Web site

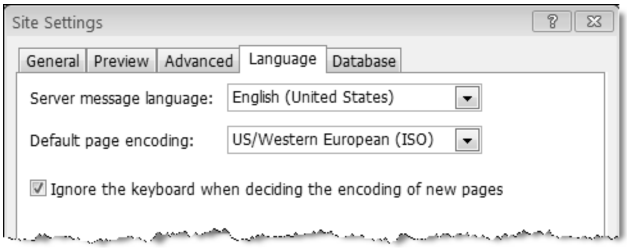
Figure 3-33. *I recommend changing this setting from the default of using the development server for ASP.NET pages only to using it for all web pages.*

The Advanced and Database Tabs

I recommend not changing the defaults on either of these tabs.

The Language Tab

Like the other language settings, this one should be set to US/Western European (ISO). There is a bug that appears in some classes—content that is imported will have `<span lang="en">text</span>` because of a character set or language mismatch between the originating program or operating system and your settings. If you get unnecessary language spans, remove the check mark shown in Figure 3-34, or replace it if the check mark is missing.



Server message language: English (United States)

Default page encoding: US/Western European (ISO)

[✓] Ignore the keyboard when deciding the encoding of new pages

Figure 3-34. *You might need to remove the check mark from the "Ignore the keyboard when deciding the encoding of new pages" section.*

Summary

In this chapter, you have learned how a one-time setting of Expression Web configuration options will help you to create websites that are easier to maintain and work well in a variety of situations. The menus you have yet to explore—Tables, Data View, and Task Pane—will be used on individual pages and are not applicable to your entire website. As such, it is better to address these toolbars as you use them over the course of this book.

In the following chapter, you will be creating web pages using structural markup with headings, lists, tables, and hyperlinks.