"The Missing Manual series is simply the most intelligent and usable series of guidebooks..."
—KEVIN KELLY, CO-FOUNDER OF *WIRED*

Dreamweaver CS5.5

the missing manual®

The book that should have been in the box®

O'REILLY®

David Sawyer McFarland

Table of Contents

CHAPTER
13

Forms

A website is a great way to broadcast a message, announce a new product, post late-breaking news, or just rant about the state of the world. But all that's *one-way* communication, which you may find a bit limiting. You may be curious to get some feedback from your audience. Or you may want to build your business by selling your product online, and you need a way to gather vital stats from customers. If you want to *receive* information as well as deliver it, it's time to add *forms* to your web design repertoire (see Figure 13-1 for a simple example). Whatever type of information you need to collect on your site, Dreamweaver's *form objects* make the task easy.

Form Basics

A form begins and ends with the HTML <form> tag. The opening tag (<form>) indicates the beginning of the form, and sets its properties; the closing tag (</form>), of course, marks the form's end.

You add different objects between these tags to deck out your pages with the elements your visitors interact with—radio buttons, text fields, and pull-down menus are just a few ways you can gather input. It's perfectly OK to include other HTML elements inside a form, too. In fact, your visitors would be lost if you didn't also add (and format) text that explains each element's purpose. And if you don't use a table or Cascading Style Sheets to lay out a form in an organized way, it can quickly become an unreadable mess (see the box on page 535).

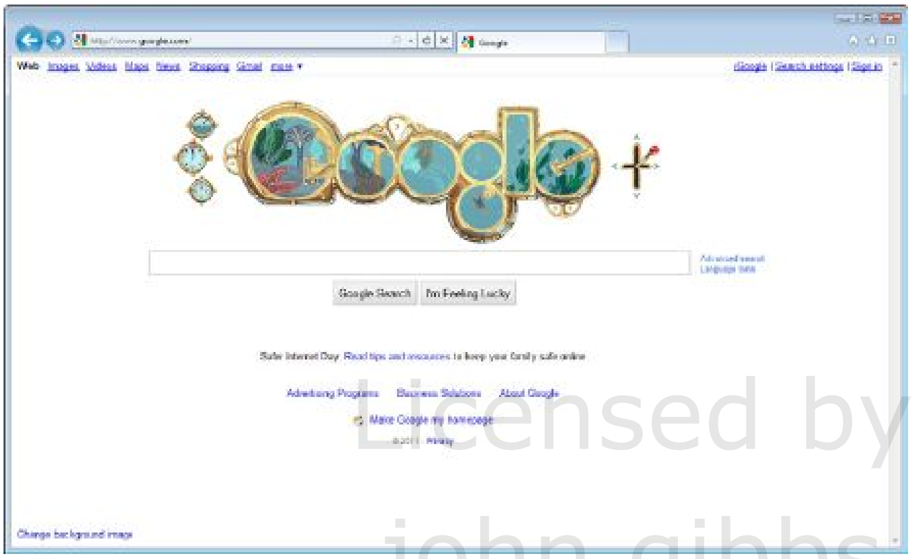


Figure 13-1: *A form can be as simple as a single empty text box (a field) and a button, or as complex as a 100-question survey composed of fill-in-the-blank and multiple-choice questions.*

Every form element, whether it's a text field or a checkbox, has a *name* and a *value*. You supply the name, which should reflect the information you're trying to collect. For example, if you want visitors to type their email addresses into a text field, you might name that field *email*. The value, on the other hand, is what the visitors actually type in—the text they enter into a text field, for example, or the selections they make from a pull-down menu.

After your visitors fill out a form and click the Submit button to transmit their responses, the browser transmits each form element as a name/value pair like this: *email=bob@bobville.com*. Submitting both pieces of information helps the program that processes the form figure out what the input means. After all, without a name, a value of "39" doesn't mean much (39 what? Potatoes, steps, days until Christmas?). The name/value pair (*age=39*) provides context for your visitor's input.

The Code Backstage

Creating a form is just the first step in collecting information from your visitors. You also need to *connect* the form to a program that actually *does* something with the information. The program may simply take the data from the form and email it to you. But it could also do something as complex as contacting a bank, processing a credit card payment, creating an invoice item, or notifying a shipping department to deliver a poster of Justin Bieber to someone in Nova Scotia.

A form is pretty useless without a form-processing program on the other end of things—running on your web server. These information-crunching programs come in a variety of languages—Perl, C, C#, Visual Basic, VBScript, Java, ColdFusion

Markup Language, PHP, Rails—and may be part of a dedicated application server, like Adobe's ColdFusion Server or Microsoft's .NET technology.

Writing the necessary behind-the-scenes processing software can be complex, but the concepts behind the forms themselves are straightforward:

1. **First, someone fills out a form on your website and clicks the Submit button (or Search, Buy, or whatever you actually label the button that transmits information).**
2. **Next, the browser transmits the form data over the Internet to a processing program on your web server.**
3. **The form-processing program collects the data and does something with it—whatever you and the programmer decide it should do. It could, for example, send the data off as an email to you, search a vast database of information, or store the information in a database.**
4. **Finally, the server returns a page to the browser, which your visitor sees. It may be a standard web page with a message like "Thanks for the info," or a page the program generates on the fly that includes information like a detailed invoice or the results of a search.**

So how do you create the processing half of the forms equation if you're not a programmer? You can use Dreamweaver, of course. Part Six of this book describes Dreamweaver's dynamic web-building tools for creating pages that use information collected from forms. If your web server accommodates PHP, for example, Dreamweaver can create form-processing programs for you. If you're part of a company's web development team, you may already have programmers on staff who can create the processing program.

Furthermore, even if your web hosting company doesn't tolerate any of the application servers that work with Dreamweaver, they probably offer free form-processing programs as part of their services. Contact your host and ask; most companies provide basic instructions on how to use these programs.

If you feel adventurous, many form-processing programs are available free on the Web. For a thorough sampling, see the CGI Resource Index at *http://cgi.resourceindex.com*. Using these free programs can be tricky, however, because you need to download the appropriate program and install it on your web server—something not every web host allows.

Lastly, you can use a form-processing service like Wufoo (*http://wufoo.com*), which handles all the complicated parts of collecting and storing information from forms and provides tools for retrieving that information in a variety of formats.

Creating a Form

In Dreamweaver, you can build forms with one-click ease using the Insert panel's Forms category (see Figure 13-2).

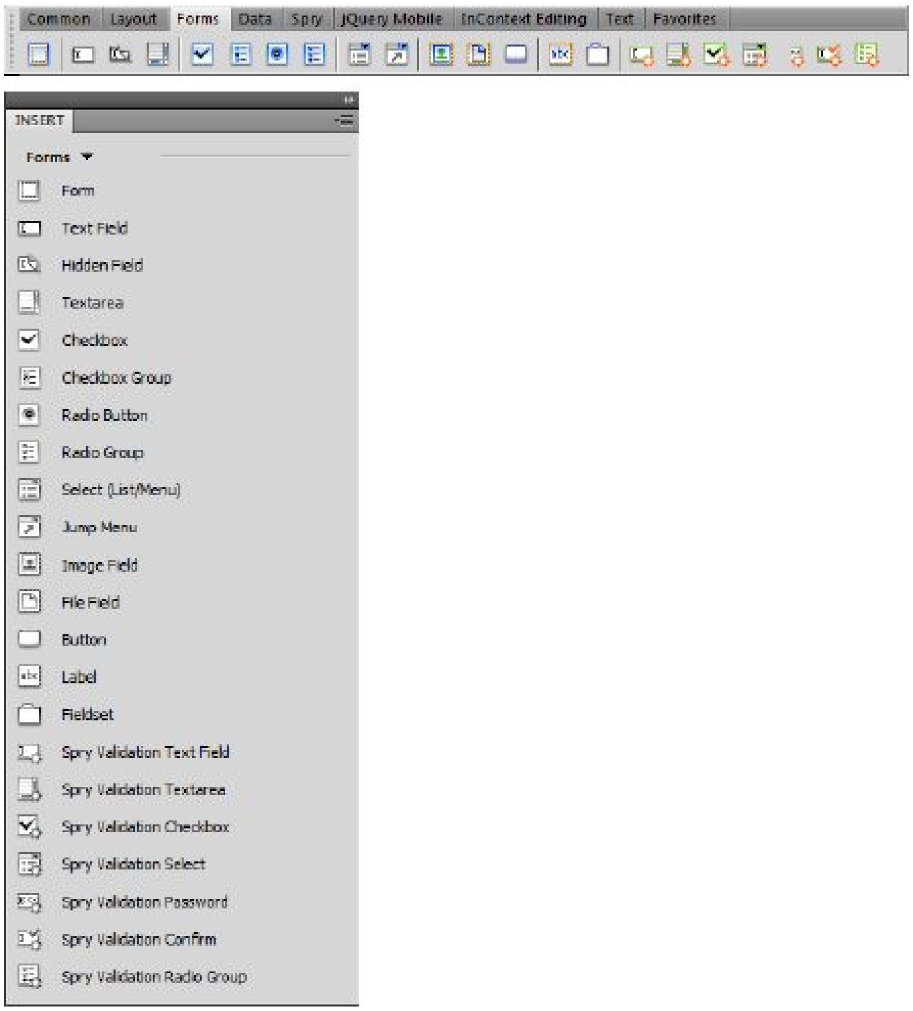


Figure 13-2: *Choosing the "Classic" view from the Workspace Switcher, as described in the Note on page 37, turns the insert panel into a toolbar that runs along the top of the Dreamweaver workspace (top). (You can also just drag the Insert panel up to the top of the screen to turn it into a toolbar.)*

If you prefer the Insert panel where it normally appears (bottom), select the Forms category to get one-click access to all the different form elements—buttons, text fields, checkboxes, and more. Since the Forms category has so many buttons, you might want to turn off the labels that appear next to each icon as described on page 26. That way, you can see all the form buttons without using your monitor's entire height.

To begin, you need to insert a <form> tag on your web page to indicate the boundaries of the form:

1. **In the document window, click the location where you want to insert the form.**

   You might decide to place it after a paragraph of introductory text, for example, or into a <div> tag that holds the page's main content.

Note: If you plan to use an HTML table to organize a form's fields, insert the form first, and then insert the table inside the <form> tag.

2. **On the Insert panel, select the Forms category.**

   The tab reveals 22 form-building tools.

3. **Click the Form icon (the very first square in the 22-icon list).**

   If you're a menu-driven person, choose Insert→Form→Form.

   Either way, a red, dashed-line rectangle appears in the document window, indicating the form's boundaries. (If you don't see it, choose View→Visual Aids→Invisible Elements.) The top line represents the opening <form> tag; the bottom represents the closing tag. Make sure you always insert form objects, like buttons and menus, *inside* these lines. Otherwise, Dreamweaver thinks you're trying to create a second form on the page. (It's perfectly valid to include more than one form per page [as long as you don't try to insert a form inside *another* form], but your visitor can submit only one form—and its data—at a time.)

Tip: An even faster way to insert a <form> tag is to bypass step 3 and just insert a form element–like a text field or radio button; Dreamweaver asks if you want to add the <form> tag at the same time.

   Since you can place so many other HTML elements inside a form, you'll often find it easier to insert the form first, and add tables, graphics, text, and form objects later.

4. **If it isn't already selected, click the dotted red line to select the form.**

   This step not only selects the form, but it highlights everything inside the red lines, too. The Property inspector displays the "Form ID" label in the upper-left corner, as shown in Figure 13-3.

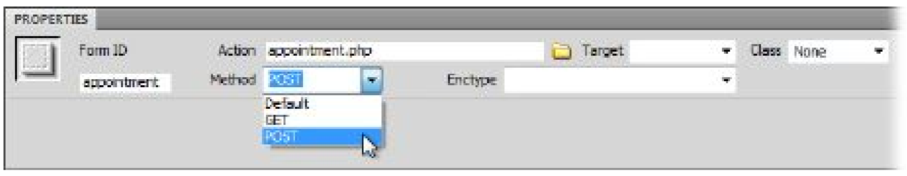


Figure 13-3: *Unless you're creating a search form, you'll generally want to use the POST method of sending data to the server. Step 7 has the details.*

5. **If you like, type a name for your form into the "Form ID" field.**

   This step is optional. Dreamweaver supplies a generic ID name—*form1*—but you don't need to name a form for it to work. A name is useful if you use the JavaScript or the Spry form validation tools discussed below because they both interact with the form or its fields, and the tools need a way to uniquely identify each form. But you don't have to change the name Dreamweaver supplies, and the name doesn't appear anywhere on the page, so you can leave the default name in place if you wish.

6. **Into the Action field, type a URL, or select a file by clicking the tiny folder icon.**

   Your mission here is to specify the location of the program that will process the form. If someone else is responsible for the programming, ask that person what to enter here. It's a standard web address—either an absolute URL (one that begins with *http://*) or the path to the server's form-processing program (see page 167 for more on these different kinds of links). If you use Dreamweaver's dynamic page-building tools, you usually leave this field blank. When you apply a server behavior—the programming code that makes the page "dynamic"—Dreamweaver automatically inserts the correct URL.

   Either way, the file name you add to the Action field *doesn't* end in *.html*. The path might be, for example, *.../cgi-bin/forms.pl*. In this case, .pl extension indicates a program written in the Perl programming language. Other common file extensions for web programs include .cfm (for ColdFusion Markup Language), .aspx (.NET pages), .php (PHP pages), .jsp (Java Server Pages), or .cgi (CGI programs).

7. **Using the Method pop-up menu, specify how you want the browser to transmit the form data to the processing program (see Figure 13-3).**

   Basically, browsers can transmit form data to a web server in either of two ways. You'll use the more common method, called POST, most often. It sends form data in two steps. First, the browser contacts the form-processing program at the URL you specified in the previous step; then, it sends the data to the server. This method gives your data a bit more security, and it can easily handle forms with lots of information.

   The GET method, on the other hand, adds the form data to the destination URL, like this: *http://search.yahoo.com/bin/search?p=dogs*. (Even though the GET method *sends* data, it's named GET because its purpose in life is to *receive* information—such as the results of a search.) The characters following the *?* in the address represent the form data. This code submits a single form field—named *p*, with the value *dogs*—to the server. If a form has lots of fields and accepts lots of user input, a GET URL can become extremely long. Some servers can't handle very long URLs, so don't use the GET method if your forms collect a lot of data.

Note: The GET method has one big benefit: You can *bookmark* it, which is great if you want to save and reuse a common search request for Google, for example, or you want to send someone Google Maps driving directions. That's why search engines use the GET method for form submissions.

8. **If you're using frames, select a Target option.**

   You'll most likely skip this menu. Frames are so 1998 web design, and pose serious problems for web designers and search engines. However, even if you're not using frames, you can choose the "_blank" option to open a new browser window to display the results. (See page 181 for more on the Target property.)

9. **Select an encoding type, if you like.**

   You usually don't have to select anything from the "Enctype" menu. Leaving this box empty is almost always correct, and is the same as selecting the much more long-winded "application/x-www-form-urlencoded" option.

   But if you use the File Field button (see page 535) to let visitors upload files to your site, you should use the "multipart/form-data" option. In fact, Dreamweaver automatically selects this option when you add a File Field to a form. See the box on the opposite page for more info on potential problems with File Field forms.

   You've laid the foundation for your form. Now you're ready to add the input controls—menus, checkboxes, and so on—as described in the next section.

FREQUENTY ASKED QUESTION

Using a Form to Upload Files

I want to let visitors upload photos to my site, but when I include a File Field button in one of my forms, I get an error from the server when I try to submit the form. Why?

To upload files from a web page, you need to do two things: Change the encoding method (see step 9 on the previous page) to "multipart/form-data," and set up your server to receive files. Dreamweaver automatically takes care of the first part: whenever you insert a form field, Dreamweaver changes the form's encoding method to "multipart/form-data."

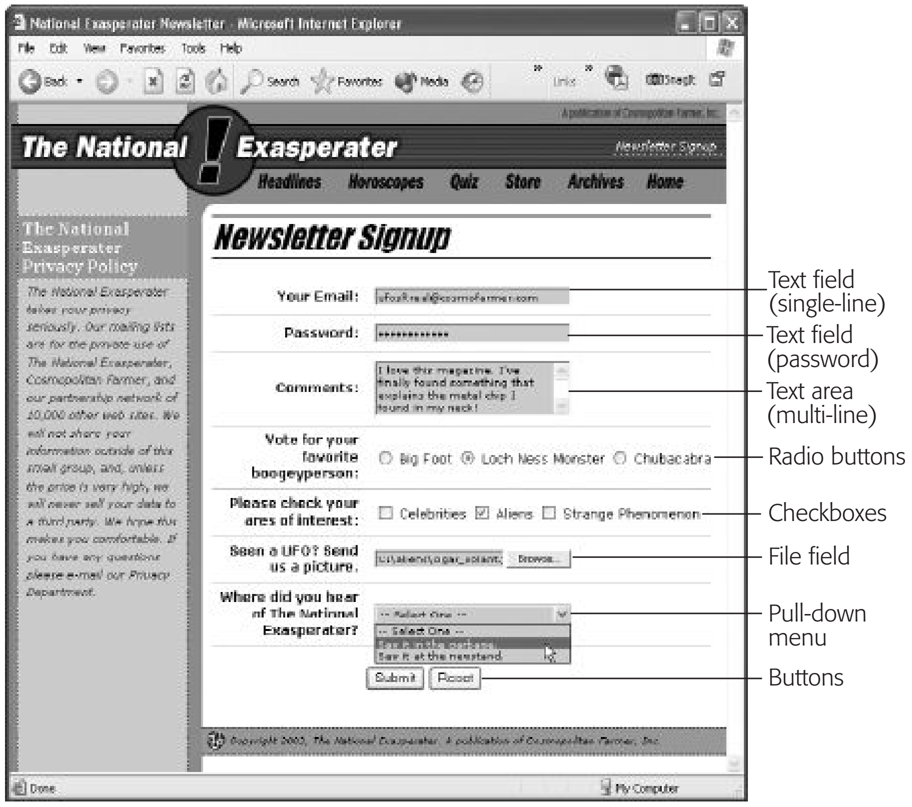
The second part is up to you (or your web hosting company). Many web servers have this option turned off for security reasons. Check to see if your web host lets you use forms to upload files to your server. If it doesn't, find a hosting company that does.

In addition, you have to program the form-processing script to accept data in the "multipart/form-data" format. Since this task is challenging, you might want to enlist some help. The box on page 537 provides several resources for commercial Dreamweaver extensions that can help with this task.

If you decide that's too much trouble and you delete the File Field button, you're still in trouble. Dreamweaver doesn't reset the encoding method to the original "application/x-www-form-urlencoded" setting, so when visitors try to submit the form (even without the File Field), they'll get a nasty error message from the server. You must remedy the situation manually, by selecting the form, and then using the Property inspector to change the encoding method back to "application/x-www-form-urlencoded."

Adding Form Elements

Unless you've never used a computer before, the user interface elements available for HTML forms should look familiar (Figure 13-4): text fields where people can type in information (their names, addresses, phone numbers, and so on); checkboxes for making multiple-choice selections; and menus for picking items from a list. The Insert panel's Forms category lets you create all these elements and more.


National Exasperater Newsletter - Microsoft Internet Explorer

The National ! Exasperater

The National Exasperater Privacy Policy

The National Exasperater takes your privacy seriously. Our mailing lists are for the private use of The National Exasperater, Cosmopolitan Farmer, and our partnership network of 10,000 other web sites. We will not share your information outside of this small group, and, unless the price is very high, we will never sell your data to a third party. We hope this makes you comfortable. If you have any questions please e-mail our Privacy Department.

Newsletter Signup

Your Email: ufos@cosmofarmer.com

Password: *************

Comments: I love this magazine. I've finally found something that explains the metal chip I found in my neck!

Vote for your favorite boogeyperson: ○ Big Foot ◉ Loch Ness Monster ○ Chubacabra

Please check your ares of interest: ☐ Celebrities ☑ Aliens ☐ Strange Phenomenon

Seen a UFO? Send us a picture. C:\aliens\ogar_solant; Browse...

Where did you hear of The National Exasperater? -- Select One -- / -- Select One -- / Saw it at the newstand

Submit Reset

Copyright 2002, The National Exasperater. A publication of Cosmopolitan Farmer, Inc.

Figure 13-4:
Forms collect information using a variety of interface elements, like text boxes, password fields, and pull-down menus. You can even add a File Field to let visitors upload files to your site.

Dreamweaver includes some special form elements, called Spry Validation widgets. They're like the form elements discussed below, but have the added ability to *verify* the contents of a form field, which prevents visitors from submitting a form if they haven't fill it out correctly. The Spry Validation widgets are discussed on page 540.

What All Form Elements Have in Common

Adding form elements to your document always follows the same pattern:

1. **In the document window, insert a form (see page 517).**

   Or, if the page already has a form, click inside its red border.

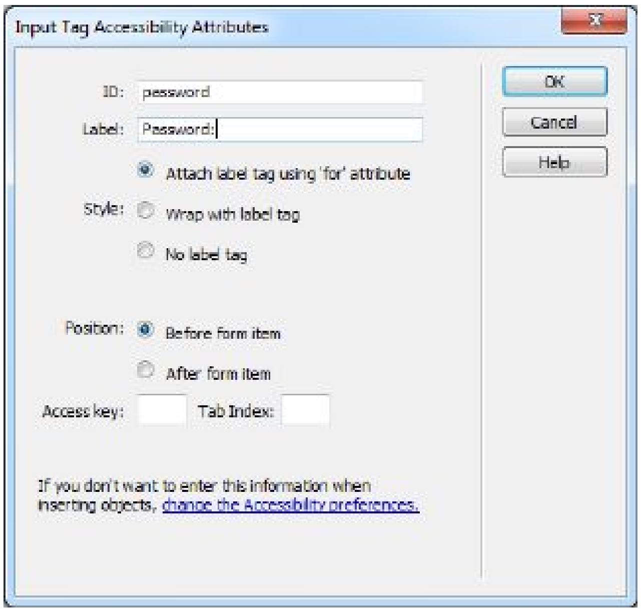
Tip: You can skip step 1 and have Dreamweaver add a form element when you first add the form field to a page. When you insert a field (step 2) and there's no form yet, Dreamweaver asks if you'd like to add the proper <form> tag. Click Yes and Dreamweaver automatically creates the red dotted-line form boundaries (and, behind the scenes, inserts the corresponding <form> tags). You should *always* click the Yes button. A form field that isn't surrounded by the proper form tag doesn't work in all browsers.

2. **In the Insert panel's Forms category, click the appropriate button (see Figure 13-2).**

   Alternatively, use the Insert→Form submenu. You'll soon discover that Dreamweaver represents every form object on the Insert panel by a command on the Insert menu, too (for example, Insert→Form→Text Field).

3. **In the Input Tag Accessibility Attributes window, type an ID (see Figure 13-5).**

   This window serves a couple of functions: It lets you assign an ID (this step) and set a few accessibility options. These options add information and tools for the benefit of those who surf using *assistive technologies*—like screen readers—or those who use their keyboard (rather than their mouse) to jump from form field to form field.


Input Tag Accessibility Attributes

ID: password

Label: Password:

Style: (●) Attach label tag using 'for' attribute ( ) Wrap with label tag ( ) No label tag

Position: (●) Before form item ( ) After form item

Access key: Tab Index:

If you don't want to enter this information when inserting objects, change the Accessibility preferences.

Figure 13-5: *This window appears when you're inserting a form element. If you don't see it, you or someone else has turned off Dreamweaver's factory setting to automatically launch this window. To summon it, choose Edit→Preferences (Dreamweaver→Preferences), click the Accessibility category, and then turn on the Form Objects checkbox.*

The ID you type in this field also determines the *name* of the field. Remember, each field has a name so that the form-processing program can identify the information it receives (see page 516). Say you add a text field to a page that collects a visitor's town name. In the ID box, if you type *town*, then, when Dreamweaver inserts the text field onto the page, the underlying HTML Dreamweaver writes looks like this: <input type="text" name="town" id="town">.

Be sure to follow the same naming conventions you use for CSS ID names: Begin with a letter, use only numbers, letters, hyphens, or underscores, and skip spaces, punctuation, and other characters. (Keep in mind that adding an ID to an HTML tag doesn't create a CSS ID style—but if you want to create a special look just for that one field, you *can* create an ID style using the name you supply for the form element.)

Note: The ID value you type in step 3 has a slightly different effect when you add radio buttons. What you type becomes the radio button's ID value, but Dreamweaver sets its name to "radio." That's because you can only use an ID once per tag on a page, but radio buttons that form part of a group (like answers to a multiple-choice question) must share the same name. Using the same name for more than one button tells a web browser that a visitor can select one and only one radio button from the group (see page 531). Therefore, make sure you rename your radio button using the Property inspector as described on page 532.

4. **Type a label, and then select label options.**

   The label option lets you add text that identifies the form element's purpose. In fact, Dreamweaver wraps whatever text you type in an HTML tag, named, logically enough, the <label> tag. This tag identifies the form field's purpose, and your visitors see the text you enter here as they fill out the form. The label usually appears either to the left or right of the form field. For example, if you add a text field to collect someone's name, you might use the label *Name:*. Someone filling out the form then sees the word "Name:" followed by a box where they can type in their name. It's always a good idea to add a label. (You can read more about the <label> tag on page 539.)

Note: Sometimes you don't need or want a label. For example, HTML's buttons—like Submit or Reset—already have a label, so you don't need to add another. In cases like this, either click the Cancel button, which adds the form field without any of these *accessibility* properties, or leave the label box empty, and select the "No label tag" radio button.

You can attach a label to a form element two ways. The first method, "Attach label tag using 'for' attribute," wraps the text you type inside a <label> tag. The form field itself isn't inside the label tag, but the two elements (the label and the field) are connected by a *for* property that Dreamweaver adds to the <label> tag, which tells a browser which form element the label is "for." This is a good option when a label and its form field don't appear directly next to each other in the HTML code. For example, web designers often use a table to visually organize forms (see the box on page 535). By placing text labels in one column of the table and form fields in an adjacent column, you can neatly align the labels and their corresponding fields, but the label and their associated fields appear in far different places in the HTML

Here's an example that might make this all a bit clearer: Say you add a text field that lets someone enter her email address to subscribe to your site's email newsletter. If, when you insert the field, you use the ID *email*, the label "Your email address:", and the "Attach label tag using the 'for' attribute" option, you end up with this HTML code:

```
<label for="email">Your email address:</label>
<input type="text" name="email" id="email" />
```

At this point, you can move the <label> tag (or the text field) to any other location on your web page, and the label remains related to the field. Of course, if you place the label at the top of the page and the field at the bottom, your visitors don't know they're related, so it makes sense to keep them in close proximity to each other. People often put the <label> tag in one table cell of a row, and the field in a table cell to the right. (The label tag has one added benefit: click it and you select the associated form field, ready for the visitor to type in a response.)

The second way to attach a label, "Wrap with label tag," wraps the <label> tag around both the text you type and the form element itself. This method keeps the two together and easily identifies which label goes with each form field. However, although wrapping a field with the <label> tag produces valid HTML, it isn't considered as accessible (to screen readers, for example) so the "Attach label tag using 'for' attribute" is generally considered the best way to go.

Note: You can skip the <label> tag entirely simply by choosing "No label tag." Any text you type in the label field is just dumped onto the page as regular text positioned next to the form field.

5. **Optionally, type an "Access key" and a Tab Index number, and then press OK.**

These steps let visitors hop to form fields using the keyboard. The "Access key" option lets guests use a keyboard shortcut to jump immediately into or select a field. If you enter *M*, for example, for a form element's access key, visitors can jump to that element by typing the access key letter plus one or more other keys:

— Alt + the access key (for example Alt + M) on Internet Explorer and Safari for Windows.

— Alt + shift + the access key (for example Alt + shift + M) on Firefox for Windows and Google Chrome for Windows.

— Ctrl + the access key (for example Ctrl + M) for Firefox on Macs, and Safari 3 on Macs.

— Ctrl + Option + the access key (for example Ctrl + Option + M) for Google Chrome on Macs and Safari 4 and later on Macs

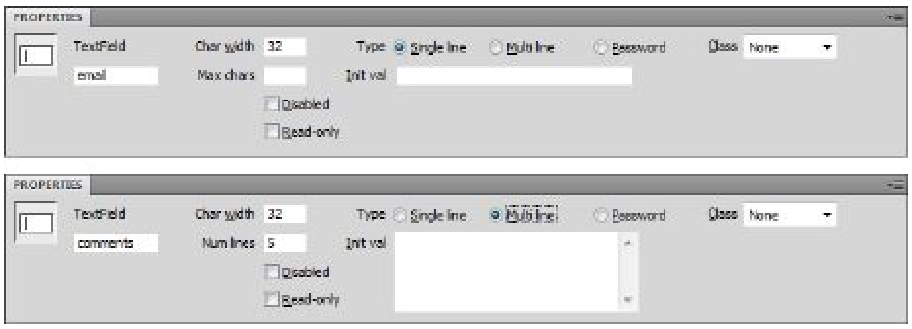
— Shift + ESC + the access key (for example Shift + ESC + M) for the Opera browser.

The biggest problem with this feature is that it's not at all obvious to your visitors what keyboard shortcuts work where. To make access keys useful, you need to list the shortcuts next to the form elements, or create a "user's manual" of sorts that explains the shortcuts. You're probably best off leaving the Access key blank.

The Tab Index lets you number each form field and, in the process, set the focus order for the fields as a visitor presses the Tab key. Number 1 indicates the first field selected, and each number after that—2, 3, 4, and so on—dictates the rest of the selection order. You don't usually need to go to this extreme, since most browsers automatically jump to the next form field when you press the Tab key, but it sometimes comes in handy when you have a particularly complex form and you use either tables or CSS to lay it out. In some cases, the default focus order doesn't match the visual presentation of the form. If that's the case, set the Tab Index so you can specify the correct tab order.

6. **In the Property inspector, set the form element's properties (Figure 13-6).**

   Some elements let you specify things like width, height, and other variables, for instance. The following descriptions indicate the options available for each form element.



PROPERTIES
TextField: email | Char width: 32 | Type: [✓ Single line] Multi line Password | Class: None
Max chars: | Init val:
Disabled
Read-only

PROPERTIES
TextField: comments | Char width: 32 | Type: Single line [✓ Multi line] Password | Class: None
Num lines: 5 | Init val:
Disabled
Read-only

Figure 13-6: *The Property inspector looks slightly different depending on the type of form element you choose. For example, the properties for a regular text field (top) differ from those for a multiline text field–actually called a "textarea" in HTML (bottom).*

Text Fields

When you need to collect a specific piece of information, like a person's name, phone number, or address, use a text field (shown in Figure 13-4). Text fields accept typed responses, and they're great for open-ended questions. They come in three different flavors: *single-line* fields for short responses, *password* fields to hide people's input from snooping eyes, and *multiline* fields for longer replies.

Once you insert a text field, you can adjust the following settings in the Property inspector:

- **Char Width**. The width of a text field is measured in characters; so if you type *20* for the Char Width (character width) setting, Dreamweaver creates a text field that holds 20 typed letters. Be aware, however, that the *physical* width of the field (how many inches or pixels wide it is) can vary from browser to browser. (You can use Cascading Style Sheets to set an exact width using the *width* property described on page 360.)
- **Type**. You can choose from three types of text field:
  - — A *single-line* text field, of course, holds just one line of text. This is the most common text field; use it for collecting small pieces of information, like a last name, Social Security number, or credit card number.

— *Multiline* fields let guests type in multiple lines of text. You need this kind of field when you let visitors type in long notes, such as those for a "Let us know what you think!" or "Nature of problem:" field.

Note: Dreamweaver includes a separate button for adding multiline text fields–called *textarea* in HTML (see Figure 13-2).

— *Password* fields hide a password a guest types from the prying eyes of passing spies. Whatever your web visitor types in appears as asterisks (*** in Windows) or bullets (••• on a Mac) on-screen. (However, the information in the password field isn't completely secret: it's still transmitted as plain text, just like any other form field. The masking action takes place only in your visitor's browser. See the Frequently Asked Question on page 528.)

- **Max Chars/Num Lines**. Max Chars (maximum characters) lets you limit the number of characters the field accepts. It's a good way to help ensure that guests type in the right information in the right place. For instance, if you use a field to collect a visitor's age, odds are you don't need more than three characters to do it; very few 1,000-year-olds surf the Web these days (and those who do don't like to reveal their ages).

  When you specify a multiline text field, the Max Chars box morphs into the Num Lines box. In this case, you can't limit the amount of text someone types into the field. Instead, you specify the height of the text field. (You can, however, use Spry Text Area to limit the number of characters a multiline text field accepts, as described on page 555.)

Note: The limit you specify here affects only how tall the field is *onscreen*. Your visitors can type as many lines of information as they want (a scroll bar appears if the number of lines exceeds the size of the box).

- **Init val**. Here, you can specify the Initial Value of the field—starter text that automatically appears in the field so that it isn't empty when a visitor begins completing the form. You can use this feature to include explanatory text inside the field itself, such as "Type your name in this box" or "Example: (212) 555-1212". Another common use for the "Init val" box: when you create an *update form*—a form for editing previously entered information. For example, if you want to update your Facebook profile, then you go to a page that presents all your current information. You can change that information and then submit the form to update your profile. An update form requires a database and some server-side programming—you'll learn how to build this type of form in Chapter 26.

Note: If your form page is one of the dynamic file types with which Dreamweaver works–ASP, PHP, or ColdFusion–you also see a small lightning bolt to the right of the "Init val" box. This button lets you add *dynamic data*–information drawn from a database–to the text field. (In-depth coverage of this feature starts on page 1034.)

- **Disabled and Read-only**. You probably won't ever have any reason to use these two options: both make the text field uneditable. The Disabled option grays out the text field and prevents visitors from clicking into it, or even selecting any text that's already there (from the Init Val property discussed above). In addition, when you disable a field, a browser doesn't submit that field's data when it submits the form itself.

  The Read-only option lets a visitor select and copy anything in the text field, but doesn't let him change it.

  Since forms are meant to collect information from visitors, don't taunt them with uneditable fields. Leave both options alone. So why do they exist? People usually use them in conjunction with JavaScript programming—for example, to disable a text field until a visitor selects another option.

FREQUENTY ASKED QUESTION

Using the Password Field for Credit Card Numbers

Can I use the Password field for credit card numbers and other sensitive information?

Yes, but it doesn't give the information any extra security.

The Password field does one thing: It hides what people type into it. Someone looking over your visitor's shoulder can't read what he's typing–it looks like a bunch of dots–but once a browser submits that information over the Internet, it's unprotected.

To provide real security for form information, you need an encrypted connection between your web server and the visitor's computer. Most website creators use SSL (Secure Socket Layer) technology for this purpose. You can identify a site using SSL by its URL; it begins with *https://*. The "s" stands for secure and browsers usually indicate a secure connection by displaying a lock at the top or bottom of the browser window.

Web browsers understand SSL, but your web server must be specially configured to work in this mode. Contact your web host to see if their servers support SSL (the answer is usually yes). If so, they can tell you where to put your files, and how to access them from a web browser. You don't have to make any special changes to your web pages, and once the server is set up, you put your web pages on it as you would for a non-secure website (Chapter 19 covers moving your files onto the Web).

Checkboxes and Checkbox Groups

Checkboxes (see Figure 13-4) are simple and to the point; a guest either checks them or not. They're great for questions that can have more than one answer. Suppose you offer visitors their choice of three email newsletters you send out each month. In your form, you might include some text—"Check the boxes for the newsletters you want to receive"—and three corresponding checkboxes with labels that indicate the name of each newsletter.

Once you add a checkbox to a form, you can set up these options in the Property inspector (see Figure 13-7):

- **Checked value**. Here's where you specify the information the browser sends to your form-processing program when a visitor selects the checkbox. Since visitors never actually see this information, it doesn't have to match the checkbox's label; it could transmit a coded response.

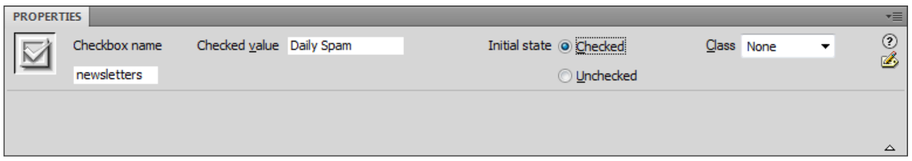


Figure 13-7: *The "Checked value" property defines the checkbox's actual value—that is, the value that's sent to the form-processing application when a guest submits the form.*

- **Initial state**. If you like, you can have a checkbox already filled in when your web page first loads. You've probably seen this setup if you've ever signed up for something on a commercial site. There's usually a checkbox—already checked—near the bottom of the form with fine print like this: "Check here if you want to get daily, unsolicited email from our marketing department."

Note: As with many form elements, the state of your checkbox (checked or unchecked) can reflect information it retrieves from a database. The Property inspector's Dynamic button–available only when you work on a dynamic page (ASP, PHP, or ColdFusion)–lets you set the checkbox state based on data in a database. See page 1036 for details.

After you set these options, if you don't use Dreamweaver's accessibility options (discussed on page 523), return to the document window to add a text label next to the field. You want to let people know what the checkbox is for: "Yes, sign me up!", for example.

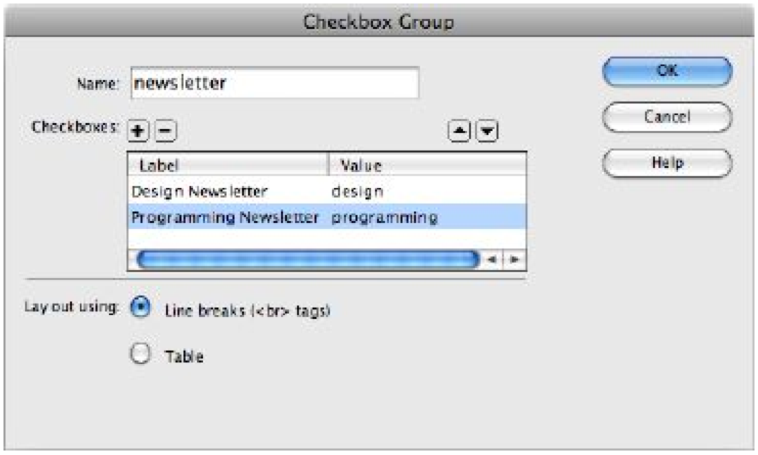
Checkboxes don't have to come in groups, but they often do, as in "yes" and "no" boxes. Dreamweaver includes a tool to make inserting multiple checkboxes easier, as discussed next.

Checkbox Groups

Checkboxes frequently travel in groups—"What activities do you like? Check all that apply." Here's how you set them up.

1. **On the Insert panel, click the Checkbox Group button.**

   The Checkbox Group window opens (see Figure 13-8).



Name: newsletter

Label	Value
Design Newsletter	design
Programming Newsletter	programming

Lay out using: [✓ Line breaks (<br> tags)] Table

Figure 13-8: *The Checkbox Group dialog box lets you quickly add multiple checkboxes to a page.*

2. **In the Name field, type a name.**

   This name applies to all the checkboxes in the group, saving you the trouble of typing in the name for each checkbox yourself. The name you type is the name the browser submits to your web server, so follow the formatting rules that apply to naming all form fields: letters and numbers, no spaces or other funny characters, except an underscore or hyphen. (To see how Dreamweaver differentiates checkboxes that all have the same name, see the box on page 531.) Although each checkbox shares the same name, if someone selects multiple checkboxes, the browser sends the data from *all* the checked boxes to the server.

3. **In the Label column, click "Checkbox" and type in a label for the first box.**

   For example, if you add a set of checkboxes so visitors can sign up for one or more newsletters, you might type the name of the newsletter here—"Our Design Newsletter," for example. This label will appear next to the checkbox.

Note: If you use the Checkbox Group tool, Dreamweaver skips the Accessibility Attributes window. You don't have any control over how Dreamweaver inserts the <label> tag; it just wraps the tag around the checkbox–the same as if you'd selected the "Wrap with label tag" option described in step 4 on page 524."

4. **Hit the Tab key to jump to the Value column for that checkbox, and then type in a value.**

   This is the value the browser passes to the web server when somebody selects the checkbox and submits the form—for example, "design" for the "Design Newsletter" option.

5. **Repeat steps 3 and 4 for the second checkbox in the group.**

   You can create additional checkboxes by clicking the + button. Follow steps 3 and 4 for each checkbox you add.

6. **Select a layout for the group.**

   Dreamweaver puts each checkbox on its own line. Choose whether you want Dreamweaver to do so using a line break (<br>) to separate each line or by creating a table with one checkbox per row.

   Don't care for either of these options? Pick the "Line breaks" option—it's easier to modify—and read the Note below.

Note: If you want a group of checkboxes to appear side by side instead of stacked one on top of the other, choose the "Line breaks" option in the Checkbox Group dialog box. Then, with Dreamweaver set to display the invisible line break character (see page 79), click the line break's gold shield in Design view, and hit Backspace or Delete to move the checkbox on the line below onto the same line as the current checkbox.

7. **Click OK to add the group of checkboxes to your page.**

   The checkboxes and their labels are essentially text (or buttons) on the screen. You can move the checkboxes around, change their labels, and, in the Property inspector, alter each checkbox's properties.

UP TO SPEED

How Dreamweaver Uniquely IDs Checkboxes

When you insert checkboxes using the Checkbox Group tool, Dreamweaver inserts all the checkboxes with the same name, but gives each a unique ID. For example, if you insert two checkboxes with this tool, you might end up with HTML that looks like this:

```
<label>
<input  type="checkbox"  name="newsletter"
value="design" id="newsletter_0" />
Design newsletter</label>
<br />
<label><input type="checkbox"
name="newsletter" value="programming"
id="newsletter_1" /> Programming newslet-
ter
</label>
```

Notice that the two boxes have the same name—newsletter—but, since you need unique ID names to differentiate the checkboxes, Dreamweaver creates them by tacking _0, _1, and so on onto the end of each ID.

It's perfectly valid to use the same name for multiple checkboxes, but keep in mind (and tell your programmer) that the data is submitted as an *array*—a data format common to programming languages that lets you store multiple items under a single name. So the values of every checked box are sent together in one group using the name you supplied in step 2, but using the unique ID that Dreamweaver added.

Radio Buttons and Radio Groups

Radio buttons, like checkboxes, are simple page elements (see Figure 13-4); they appear either selected (represented by a solid circle) or not (an empty circle).

But unlike checkboxes, radio buttons require your visitor to make a single choice from a group, just like the radio buttons on old cars (or, if you're too young to remember those car radios, like the buttons on a blender). Radio buttons are ideal for multiple-choice questions that require a single answer, like, "What is your income: A. $10,000–35,000, B. $35,001–70,000, C. $70,001–100,000, D. None of your business."

In the Property inspector, set the following options for a radio button (Figure 13-9):

- **Name**. Dreamweaver supplies the generic name *radio* (or radio2, radio3, and so on) when you insert a radio button. Make sure you change it to something more descriptive, and, when you insert a group of related radio buttons, give them all the *same name*. Your visitors should be able to select only one button in the group. To make sure that's the case, every button in the group needs to share the same name (although they should have different "Checked values;" see the next bullet point).

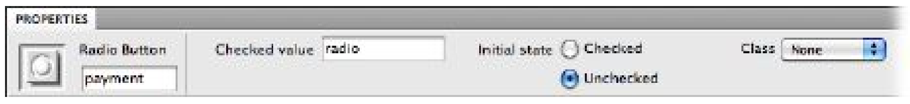

PROPERTIES
Radio Button: payment
Checked value: radio
Initial state: ○ Checked ● Unchecked
Class: None

Figure 13-9: *Radio buttons offer a choice of answers to a single question.*

  If, when you test your page, you notice that you can select more than one radio button at a time, you must have given them different names. (Consider using Dreamweaver's Radio Group object, described next. It acts like a wizard, simplifying the process of creating group radio buttons.)

- **Checked value**. This is the information your form submits to the server if a visitor selects the associated radio button. Once again, the info doesn't have to match the button's on-screen label. If you filled out the Accessibility window's ID box, Dreamweaver uses the ID you supplied as the checked value. If you don't like it, change it here.

- **Initial state**. When you create a radio-button form, you can set a button as prechecked when the page loads. To do your visitors this timesaving courtesy, turn on Checked for the button that holds the default value—the one they'll choose most often.

  Of course, if making a choice here is optional, leave all the buttons unselected by setting their initial state to Unchecked. However, once somebody *does* select a radio button, only the Reset button (if you add one) can unselect them *all* again (see page 537 for information on creating a Reset button).

  Finally, you should add a text description for the entire group. For example, if you use radio buttons to let visitors choose a method of payment, your introductory text might say, "How would you like to pay for your item(s)?" There isn't any special HTML for creating a label for an entire group of buttons, so you just type the descriptive text next to the radio buttons.

Radio Group

Although you can easily create a group of radio buttons using the Radio Button object, Dreamweaver makes it even simpler with the Radio *Group* object, a single dialog box that creates a group of radio buttons and their labels in one fell swoop. It works the same way as the Checkbox Group tool discussed on page 528, except that Dreamweaver inserts radio button form objects instead of checkboxes.

Pull-Down Menus and Lists

While checkboxes and radio buttons let you ask multiple-choice questions, use them when your questions offer relatively few answer choices. Otherwise, your form can quickly become overcrowded with buttons and boxes. And therein lies the beauty of lists and pull-down menus (usually called *pop-up menus* on the Mac)—they offer many answer choices without taking up a lot of screen space (Figure 13-10 shows an example).

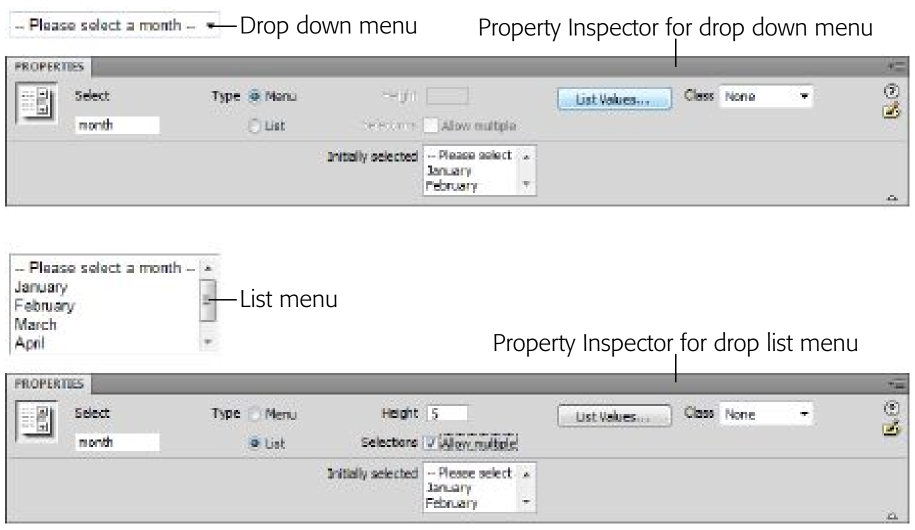


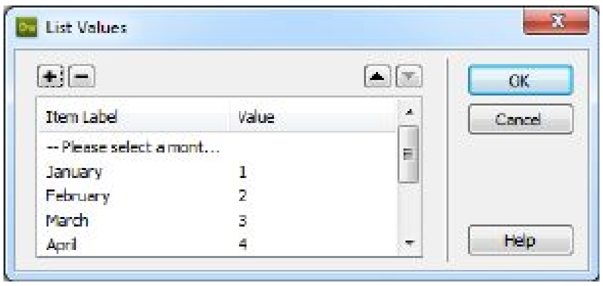
Figure 13-10: *A menu (top) is a single compact line; a list (bottom) can take up any number of lines on the page. Use the first menu or list item to tell visitors what to do. For example, "--Please select a month--" or "--Select One--".*

Once you insert a menu or list object into your document, adjust its settings in the Property inspector.

- **Type**. Menus and lists differ both in appearance and function, as described in a moment. Click the one you want (Menu or List).
- **Height**. In the Height box (available only for lists), type in the number of lines you want the list to take up on the page. That can vary from a single line (in which case you might as well use a menu) to many lines (displaying a number of choices at once). Dreamweaver adds a vertical scroll bar if the height you specify is smaller than the number of items in the list.

- **Allow multiple**. Here's a key difference between menus and lists: If you turn on this option, a visitor can select more than one item from a list, just by pressing the Ctrl (⌘) key while clicking items. (If you choose this option, be sure your instructions tell visitors they can select multiple items.)
- **List Values**. This button opens a dialog box where you type in the items that make up your menu or list. You specify two pieces of information for each item: a *label* (the text that appears in the menu or list on the web page) and the *value* (the information your form submits to the web server, which isn't necessarily the same thing as the label).

  To use this dialog box, type in an item label. Press Tab (or click in the Value column), and then type a value, if you like (see Figure 13-11 for details).



Item Label	Value
-- Please select a mont...	
January	1
February	2
March	3
April	4

Figure 13-11:
Using the + button, you can add an item to the end of a list; when you click inside the list's last item's Value column, pressing Tab creates a new list item. To delete an item, select it, and then click the minus sign (-) button. You can move an item higher or lower in the list by selecting the item and then clicking the up or down arrow buttons. Like radio buttons, pop-up menu and list items always flock together–nobody ever creates just one.

Note: A menu item's label isn't the same as the HTML <label> tag discussed on page 539. It's just the text that identifies an item in the menu.

  Values are optional; if you don't specify one, the form submits the item's label *as* the value. Still, you'll often find a separate value useful. Imagine you design a pull-down menu on an e-commerce site so your visitors can select their credit cards' expiration month. Figure 13-11 shows what the items for such a pull-down menu might look like. It displays the names of the months, but the form actually transmits the *number* of the month to your form-processing program. When a visitor selects "April," the form submits 4.

  Computer programs often work more easily with numbers than with names, while humans do the opposite. So when you offer visitors a pop-up menu of products, the label might use the human-friendly name of the product ("Blue Wool Cap"), while the value reflects a model number that your form-processing program readily understands (XSD1278, say).

- **Dynamic values**. Dreamweaver can also create *dynamic menus*, where the menu's labels and values come from a database. This option—available only when you insert a menu into one of the dynamic page types described in Part Six of this book—is great when the menu items change frequently, as they would in a list of employee or product names. Read more about this feature on page 1039.

Click OK when you finish building your menu or list. You can always return to this screen to edit the options; in the document window, click the menu or list and then, in the Property inspector, click the List Values button. You return to the List Values dialog box.

As with other form elements, you can, and probably should, add some explanatory text alongside the menu or list in the document window. One easy method: You can automatically add a label to a menu or list using Dreamweaver's accessibility features as described on page 524.

Note: Styling form fields with CSS can be frustrating. Not all browsers format form fields the same way, and some limit the kind of styling you can apply. For comprehensive coverage, check out the free appendix (164 pages on CSS and forms alone!) from Christopher Schmitt's *CSS Cookbook* at *http://tinyurl.com/3jpcr39*.

POWER USERS' CLINIC

Giving Order to Your Forms

If you're not careful, creating forms can quickly lead to visual chaos. The different shapes and sizes of text boxes, radio buttons, and other form objects don't naturally align well with text. One solution: Use tables to control your forms' appearance.

If you simply place form labels and fields line after line, you end up with an ungainly zigzag pattern created by the differing lengths of label text and the form fields. The result is not only ugly, but hard to read.

To better organize a form, you can insert the form tag, then insert an HTML table made of two columns and as many rows as you have form fields; one column holds the label, the other the text box (or other form field). Align the text in the first column to the right, and you'll create a clean edge that effectively mirrors the edge created by the form fields.

To make this table-based solution work most effectively, set each text field to the same width, using the *Char Width* property (page 526) or Cascading Style Sheets and the CSS *width* property (page 360).

You can also use CSS to lay out a form. This technique is a bit more complex, but if you're interested, you can find a good tutorial on CSS-based form layout at *http://tinyurl.com/3rhrec6*.

File Field

Receiving responses to checkboxes, radio buttons, and pull-down menus is all well and good, but what if you want your visitors to submit something a little meatier—like an entire file? Imagine a bulletin board system that lets guests post JPEG images of themselves, or upload word processing documents to share with others. They can do just that thanks to Dreamweaver's File Field feature (see Figure 13-4)—and a little magic from your web server.

Before you get carried away with the possibilities the File Field offers, you need to do a little research to see whether you can use it on your website. Although Dreamweaver lets you easily *add* a field so guests can upload image files, text files, and other documents, you need to check with your web host to see if they permit anonymous file uploads (some don't for fear of receiving viruses or performance–choking large files). Then, of course, you have to ensure that the program that processes the form actually *does* something with the incoming file—stores it on the server, for instance. Dreamweaver doesn't have any built-in functions that help with this back-end work, but you can enlist some third-party solutions as described in the box on page 537.

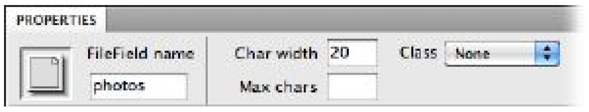
When you click the File Field button on the Insert panel's Forms category (or choose Insert→Form Objects→File Field), Dreamweaver inserts a text field *and* a Browse button; together, they constitute a single File Field. When you click either one, you highlight both.

The Browse button, once it appears in somebody's browser, opens the standard Windows or Macintosh Open File dialog box, letting your visitor navigate to and select a file to upload.

The Property inspector offers only two settings (other than specifying a more creative name):

- **Char width**. Dreamweaver measures the width of text fields in characters; type *20* in the character width box and Dreamweaver creates a field 20 characters wide.
- **Max chars**. Leave this blank, as explained in Figure 13-12.

You haven't finished the File Field until you add instructions or a label in the document window, something like "Click the Browse button to select a file to upload" (again, Dreamweaver simplifies this task with the Label option in the form's Accessibility window described on page 524).



FileField name: photos
Char width: 20
Max chars:
Class: None

Figure 13-12:
Avoid the "Max chars" field. It's intended to limit the number of characters that the field accepts, but doesn't have any effect on the File Field, which selects the full path to the file regardless of how long it is.

Hidden Field

Most form elements are designed to accept input from your guests: click a radio button, type into a text field, and make a choice from a menu, for example. But visitors don't even know about, and don't ever see, one kind of form field: the *hidden* field.

Note: Hidden fields aren't exactly hidden–it's true that visitors don't see them in a browser, but they (and their data) are visible if a visitor checks the page's HTML (using the browser's View→View Source or View Page Source command), In other words, despite their name, don't put anything into a hidden field that you wouldn't want someone to see.

Why, you're probably asking, would you need to submit a value you already know? Because hidden fields supply information to the programs that process forms—information that the program has no other way of knowing. Many web hosting services, for example, offer a generic form-processing program that collects information submitted with a form and emails it to the site's administrator. But how does the program know where to email the data? After all, it's a *generic* program that hundreds of other people use. The solution: A hidden field that stores the information required for the program to properly process the form—like *email=me@mydomain.com*.

To insert a hidden field, click the Insert panel's Hidden Field button (under the Forms category), or choose Insert→Form→Hidden Field. A gold shield icon appears on the page (this is Dreamweaver's symbol for HTML that you can't see in web browsers). Use the Property inspector to give the field a name and a *value*—that is, the value you want to submit to your form-processing program (in the example above, that value would be your email address).

Note: Gold shields indicating hidden fields appear only if, in the Preferences window's Invisible Elements category, you turn on the Hidden Form Fields checkbox (see page 79),and, in the View menu, you turn on Invisible Elements (View→Visual Aids→Invisible Elements).

POWER USERS' CLINIC

Adding File Upload Ability to Your Site

Imagine adding a "Job Application" page to your site, where applicants can upload their resumes for review. Or a web-based way for your clients to submit graphics files and word processing documents they want included in the pages you're building.

Dreamweaver lets you add a File Field to a form, but doesn't provide the tools you need to make this useful feature function on your site. To compensate for that glaring omission, you can turn to extensions that add this power to Dreamweaver when you build a dynamic site (described in Part Six of this book). But before you shell out any hard-earned cash for the extensions listed next, make sure your web hosting company allows anonymous file uploads from a web form–some don't.

DMXZone (*www.dmxzone.com/index?3/1019*) offers three fee-based extensions for ASP, ASP.NET, and PHP. The Pure Upload extension offers many different settings to manage the process of uploading files to a site, including the ability to rename duplicate files and to add file information to databases.

WebAssist (one of the big players in the Dreamweaver extensions market) offers a commercial product, Universal Email, for uploading *and* downloading files from a server (*http://tinyurl.com/68dj2lt).* As its name indicates, this extension also handles sending the contents of a form as an e-mail message. This extension works for PHP.

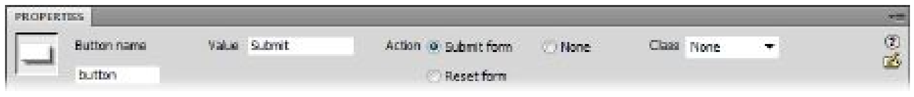
Buttons

No form is complete without a Submit button so your visitors can register their choices (see Figure 13-4). Only when guests click this button do their responses set out on their way to your form-processing application. People sometimes add a Reset button, which visitors can click if they make an error; it clears all their form entries, and resets all the form fields to their original values.

To add either type of button, use the Insert panel's Forms category or choose Insert→Form→Button. If the Accessibility window appears (see page 523), you don't need to add a label, since the button itself has "Submit," "Reset," or whatever text you wish emblazoned across its face, so just click the Cancel button.

The Property inspector controls (Figure 13-13) for a freshly inserted button are:

- **Button name**. The button's name provides the first half of the "name/value" pair that forms send to your server (see page 515).



PROPERTIES

Button name: button

Value: Submit

Action: [✓ Submit form] None Reset form

Class: None

Figure 13-13: *Buttons have just three properties: Name, Value, and Action. Like other form elements and HTML tags, Dreamweaver lets you apply a CSS class style to buttons to improve your form's design.*

- **Value**. The value is the label that appears on a button. Dreamweaver proposes *Submit*, but you're free to substitute *Buy Now*, *Make It So*, or *Send my data on its merry way*.

  What your visitors see printed on the button—"Click Me," for example—is the value transmitted along with the button's name when guests submit the form. This characteristic opens up some interesting possibilities. You could, for example, include *several* Submit buttons, each with a different label. If you create a form for a database application, for example, one button might say Delete, while another says Edit. Depending on which button your visitor clicks, the program processing the form either deletes the record from the database or modifies it.

- **Action**. The three Action options govern what happens when somebody clicks your button. The "Submit form" button transmits the form data over the Internet to your form-processing program. The "Reset form" button sets all the fields back to their original values. (The fields, checkboxes, or menu items aren't left blank or unselected, they return to their *initial* state, which you specified when you created the control. So, if you set the Initial State property of a checkbox to Checked, and your visitor unchecked the box and then clicked the Reset button, the box becomes Checked once again.)

  The Reset button used to appear on nearly every form on the Web; these days it's much less frequent, mainly because it's unlikely that anyone would want to *completely* erase *everything* she's typed into a form. In addition, its presence offers the unfortunate possibility that a visitor, after painstakingly filling out a form, will mistake the Reset button for the Submit button, and click it—erasing everything she's typed. So if you include a Reset button, it's probably best not to put it right next to the Submit button.

Note: A Reset button can come in handy on a page intended to *update* information. An update form contains previously recorded information (like the shipping address for your *Amazon.com* account). In this case, a Reset button lets you erase any mistakes you make when you update your account information. Click the Reset button and the form goes back to displaying the original information, like your original shipping address. You'll learn how to create an update form for a database-driven site in Chapter 26.

Setting the button's action to None means that clicking the button has no effect on the *form*. "Gee *that's* useful," you're probably thinking. But while the button doesn't trigger an action related to the form, you *can* use it to program one of Dreamweaver's built-in behaviors (see Chapter 15). The only way to do that is to make the button available for programming by choosing None. This way, you get a common user interface element—the cool 3-D look of a beveled form button—that can trigger any of many different actions, like opening a new browser window or popping up a message on the screen. If you're a JavaScript programmer, you can also use this button to activate your own programs.

Note: You can use a graphic as a Submit button, too, thus freeing you to be more creative with the look of the button. That's thanks to something called an Image Field. On the Insert panel, click the Image Field button or choose Insert→Form→Image Field to select the graphic you want to use. When a visitor clicks the image, it submits the form and all its data. (Image Fields do only one thing: Submit form data. You can't use them to customize a Reset button, for example.)

The <label> Tag

As discussed on page 524, the <label> tag lets you associate a label with a particular form element, like a checkbox or text field. Of course, you can always place plain text next to a form element on a page. But because a <label> tag is "attached" to a particular form element, it's more helpful in explaining the function and layout of your form to people who use assistive technologies like screen-reading software for the blind.

On the Insert panel's Forms category, the Label tag button doesn't do much more than switch you into Code view and drop the <label> tag into your HTML. You're much better off inserting labels with Dreamweaver's form accessibility option, as described on page 524. However, there are some cases where you don't want to put the label directly next to the form field; for example, when you use tables to lay out a form, you usually put the label in one table cell, and the form element in another. In such a case, you need to jump into Code view to add a label anyway, and this button can save you a little typing.

The <fieldset> Tag

The <fieldset> tag is a form-organization tool that lets you group related form fields. For example, if you create an online order form, you can organize all the "ship to" information—address, city, state, Zip code, and so on—into a single set. Again, this arrangement can help those using assistive technology to understand the organization and intent of a form.

The <fieldset> tag also has a visual benefit: Browsers display a border around fieldsets. In addition, the Legend tag (which Dreamweaver automatically adds whenever you insert a fieldset) lets you add a description of the fields grouped inside a fieldset. The legend appears at the top of the fieldset.

To use this tag, select the related form fields. You have to position the form fields next to each other onscreen, and you can organize them within other HTML elements like a table. Then, on the Insert panel's Forms category, click the Fieldset button. In the Label window that appears, type a label (called, somewhat dramatically, a "Legend") for the fieldset, and then click OK.

In addition to displaying the label you type, Dreamweaver creates a simple border around the group of fields you select. Because different browsers display this border differently, make sure you preview the page (F12, or Option-F12 on a Mac) in a recent version of Internet Explorer, Firefox, Opera, and Safari, or use Adobe's BrowserLab tool (page 748), to see how the label and the surrounding border look in different browsers.

Validating Forms

You might get frustrated when you review feedback submitted via a form on your web page, only to notice that your visitor failed to provide a name, email address, or some other critical piece of information. That's why, depending on the type of form you create, you might want to make certain information *mandatory*.

For instance, a form that signs up guests for an email newsletter isn't much use if the would-be reader doesn't type in an email address. Likewise, if you need a shipping address to deliver a brochure or product, you want to be sure that your visitor includes his address on the form.

Luckily, Dreamweaver includes a set of validation options that do exactly that. They're called Spry Validation "widgets." (The term *widget* refers to any of the Spry-based, interactive web page elements that Dreamweaver helps you create, such as the Spry Menu Bar, Spry Validation Text Field, and Spry Tabbed Panels.) With a Spry Validation widget, you can display a friendly "Hey, please fill out this box" message when someone tries to submit a form that's missing important information. You can specify that visitors can't leave a particular field blank, or that a field must contain information in a specific format, such as a phone number, email address, or credit card number. If someone tries to submit a form without the correct information, your message appears. And instead of an annoying and amateurish JavaScript error window popping up, Spry form validation widgets display error messages right on the web page, and right next to the faulty form field. You can even change the field's look to highlight a problem (add a red background to the field, for example).

FREQUENTLY ASKED QUESTION

Emailing Form Results

I don't want to store form submissions in a database or anything fancy like that. I just want to get an email message that includes the information a guest submits. How do I do that?

This common function—available on countless websites—may seem like an easy task, but Dreamweaver doesn't supply a tool to automate the process. Basically, you need a program to collect the data and send it off in an email message. Most web hosting companies provide just such a program. They generally work like this: You build a form, set the form's Action property (see page 520) to point to the URL of the server's form-emailing program, and then add one or more hidden fields. The hidden fields contain information that the program uses—your email address, for example, and the URL of the page the browser should load after it submits the form. Since this form-emailing program varies from server to server, you need to contact your hosting company for details.

Many commercial Dreamweaver extensions can help you, too. For basic form mailing, the Mail Form extension for ASP and PHP is available from Felix One for $38 (*http://tinyurl.com/6cty3lq*). Two other extensions offer much more advanced emailing features, including the ability to mass-mail newsletters to email addresses stored in a database: WA Universal Email ($99) from WebAssist (*http://tinyurl.com/5uhm4g4*) works for PHP pages and also supports file uploads, and DMXZone sells both an ASP (*www.dmxzone.com/go?5578*) and a PHP (*www.dmxzone.com/go?5628*) version of its Smart Mailer extension ($49).

For all these extensions, however, your server has to support the appropriate programming language (ASP or PHP)—Part Six of this book has more on server-side programming.

Spry Validation Basics

Spry Validation widgets let you verify input in a text field, a text area, a pull-down menu, a checkbox, or a group of radio buttons. You can make sure guests fill out a field, turn on a checkbox, select from a list, or click a radio button. You can limit input to a specific type of information, such as a date or phone number, and even limit the number of letters someone types into a text box.

The basic process for all form validation widgets is the same:

1. **Insert the Spry widget.**

   Buttons for inserting the seven types of Spry Validation form fields appear in four places: on the Insert panel's Forms menu, in the dedicated Spry menu, from the Insert→Form submenu, and from the Insert→Spry submenu. The initial steps are the same as those for inserting any form field. The Input Tag Accessibility window appears, and you follow steps 3, 4, and 5 on page 523.

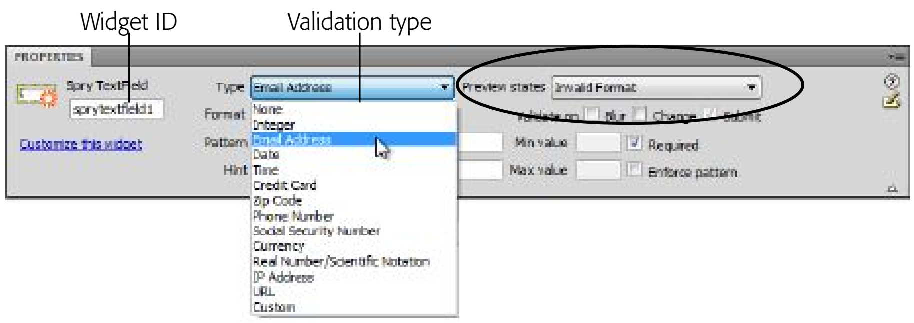
If you already inserted a text field, multiline text box, checkbox, or pull-down menu in a page, you can add Spry Validation to it by selecting the form element and then, on the Insert panel, clicking the appropriate Spry form button. If you want to validate a text field, for instance, select the text field, and then click the Spry Validation Text Field button. You can't add validation to a group of already-placed radio buttons, however. To do that, you have to create them as part of the Spry Validation Radio Group widget (page 577).

When you insert a widget, Dreamweaver adds more than just the HTML needed to validate the form field; it also inserts a <span> tag surrounding the field, a label, and the HTML necessary to display one or more error messages. In addition, the widget adds JavaScript programming to verify the validity of the information in the field, and CSS to style the appearance of the field and the error messages.

Note: When you save a web page after inserting a Spry widget, Dreamweaver pops up a window letting you know that it's added JavaScript and CSS files to the SpryAssets folder in the site's root folder (see the Tip on page 197).

2. **Rename the widget (optional).**

   Once you insert a widget, you can rename it using the Property inspector (see Figure 13-14). Dreamweaver assigns every Spry widget a generic ID like *sprytextfield1, sprytextfield2*, and so on. You can change this to something more descriptive, but for clarity's sake leave "spry" in the ID name. If you insert a Spry text field to collect a person's email address, for example, you might name the widget *spryEmail*.



Spry TextField
sprytextfield1
Type: Email Address
Preview states: Invalid Format
[✓ Required]

Figure 13-14: *Depending on the type of form field, Spry Validation can check for different types of input. In the case of a text field, pictured here, Dreamweaver lets you verify that the visitor-supplied text conforms to one of 14 formats.*

Dreamweaver applies the ID to a <span> tag that wraps around the actual form field, form label, and error messages that Spry creates. Don't get this ID confused with the ID you assigned to the form field—that's a different tag that requires its own ID. That's why it's a good idea to include "spry" in the ID you assign to the widget. If this all sounds confusing, do yourself a favor and don't bother renaming the widget. Dreamweaver can track the IDs just fine, and since the generic name Dreamweaver assigns is never visible on the form, no one visiting your site knows the difference.

3. **Assign a validation requirement.**

   Use the Property inspector to assign the type of validation you want to apply. The most basic type of validation simply ensures that your guests type *something* into a form field, make a selection from a pull-down menu, turn on a checkbox, or select a radio button. But each type of form field has additional validation options. For example, you can make sure a visitor fills out a text field with numbers in the correct format for a credit card. The options for each field are discussed below.

Note: Properties for a Spry widget appear in the Property inspector only when you select the widget (as opposed to selecting the form field itself). To do that, mouse anywhere over the form field until a blue Spry tab appears (see Figure 13-15), and then click the tab to select the widget.

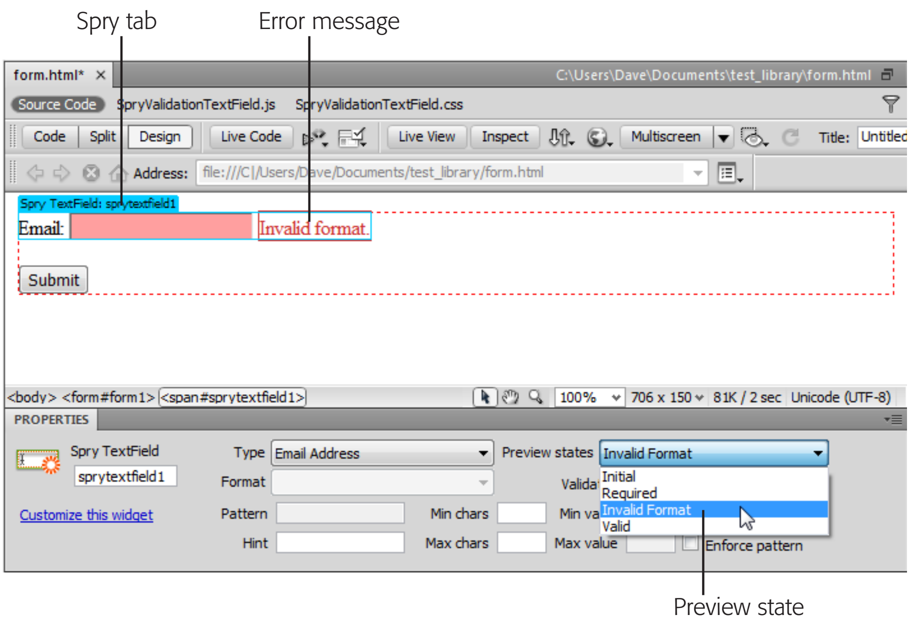


Figure 13-15: *Click the blue Spry tab to select a Spry Validation widget. This is the only way to display the widget's properties in the Property inspector. If you click the form field (text box, checkbox, or pull-down menu), you see just the regular form field properties, discussed on page 526.*

4. **Select when the validation occurs.**

   A browser validates form fields as soon as a visitor submits the form. So, when someone clicks the Submit button, the JavaScript in the web page checks to make sure your guest filled out everything correctly. If not, the form does not pass Go and one or more error messages appear, letting the visitor know what went wrong. In fact, Spry tools always validate information when a browser submits a form, and you can't turn this behavior off (that's why, in the Property inspector, the Submit checkbox is checked and grayed out [see Figure 13-14]).

   However, to provide more responsive feedback, you can also check to make sure the form has the right kind of info immediately after your visitor interacts with it. Say you add a text field to collect a visitor's email address, and some wisenheimer types in, "I'm not telling" instead of his email address. You could present an error message—like "This is not a valid email address"—when he tries to submit the form. Or you could display an error message the moment he moves on to the next field. This kind of instant feedback can make it easier for your site visitors: They immediately see and fix their mistakes instead of waiting until they submit the form.

   You dictate *when* a field is valid by turning on one or both of the Property inspector's "Validate on" checkboxes (circled in Figure 13-14). Dreamweaver lets you validate a form field when the field is "blurred," "changed," or both. "Blurred" doesn't mean the field suddenly gets fuzzy; it refers to the moment when a visitor clicks on another field or another part of the page. Also, if you type something into a text box and then press the Tab key to jump to another field (or click anywhere outside that filed), the browser considers the text box "blurred." This blurred state is a great time to validate a text field, because you know the visitor is done with that field.

   You can also validate a field when the field "changes." In this case, "change" means anything entered in the field. When a guest types into a text field, for example, each letter the person types represents a "change" to the field, so the browser validates the field following each keystroke. This can be a bit annoying, since an error message might appear the moment your visitor starts typing. For example, if you validate email addresses, the JavaScript looks for text in the form of *bob@somewhere.com*. So say Bob clicks in the field and starts to type. The moment he hits the 'b' key, he changes the field and JavaScript validates its contents. But since 'b' isn't a proper email address format, Bob gets an "Invalid Format" message. That's a bit rude. In a case like this, the "blur" option is better, since it waits until your visitor finishes filling out the field.

   On the other hand, validating a pull-down menu field when it changes can be quite useful. Say you add a pull-down field to a form, and the first option is "Please make a selection." Obviously, you want people filling out the form to select something other than that the initial "Please make a selection" option. Imagine that someone starts to fill out the form, and she scrolls down to an item on the pull-down menu; then, for whatever reason, she changes her selection to "Please make a selection." This option isn't valid, and the browser should notify her immediately. If you validate the form when it changes, that's what will happen. But if you validate the page when the menu is "blurred," the browser notifies your guest only after she clicks somewhere else on the page—a few moments later, rather than immediately.

In general, "blur" works best for text fields and text areas, while "change" is better for checkboxes, radio buttons, and pull-down menus.

5. **Set other options for the widget.**

   Some widgets have other settings that can come in handy. For instance, with a text validation widget, you can limit the number of letters someone can type into a text area, and you can add "hints," like "Type your name in this box," in a text field widget. These options are discussed below.

6. **Modify error messages.**

   Preventing incomplete fields solves only part of the valid-information problem. When a visitor leaves a required field blank, or types incorrect information into a field, you need to let him know what went wrong so he can fix it. Every form validation widget includes one or more error messages. An error message appears next to an invalid form field entry, for example, and different error messages appear under different circumstances. A Spry Validation Text Field left blank displays the message "A value is required," for example. A field filled with the wrong type of response—a word instead of a year in answer to "What year were you born?", for instance—triggers the error message "Invalid format."

   You can customize each of these messages from the Property inspector by first selecting the proper "preview state," which shows you where Dreamweaver will display the message and what the message will say. In Figure 13-15, for example, selecting Invalid Format displays the error message "Invalid format" if a guest leaves the field blank.

   To change the error message, select the text in Design view and type in a new message. It's generally a good idea to come up with a friendly and descriptive notice. If you programmed a text field to accept a date in a particular format, for instance, you might change the "Invalid format" error message to something like "Please enter a date in this format: 02/27/2011."

Note: Be careful when you select a Spry error message; you can inadvertently delete both it and the <span> tag that the Spry widget relies on. Without that <span> tag, the validation won't work. A good precaution is to select everything up to (but not including) the final period in the error message and then type in the new message.

Most validation widgets have more than one error message, so make sure you preview each of the "states." Some states have no error message, so you're just previewing the page in the selected state. For example, no error message appears when a form first loads, so the "Initial" option in the preview state menu just shows you what the form field will look like when a browser loads the page.

Every other widget (except the checkbox widget) also includes a "Valid" preview state. This is how the form field looks when it receives input. There's no error message in this instance, but the form field's background changes to green. You create this green formatting with CSS, which you'll learn to modify next.

Tip: You can change the placement of a Spry form field error message by going into Code view, and then moving the <span> containing the error message—it'll look something like *<span class="textfieldInvalid FormatMsg">Invalid format</span>*. Because each form field can have multiple error messages (for Required and Invalid formats, and so on), there may be more than one <span> element. However, keep in mind that each Spry form widget has *another* <span> that surrounds the label, the field, and the error messages—it looks something like *<span id="sprytextfield1">*. You can only move the error messages' <span> tags to another location *within* the surrounding widget's <span> tag. If you move them outside that, the error messages no longer work.

Formatting Spry Error Messages and Fields

Spry error messages appear in red with a red outline. Fortunately, you're not stuck with this factory setting. CSS controls the display of the Spry widgets, and a single style controls the "invalid" error message format. When you insert a Spry Validation widget, Dreamweaver adds its style sheet to the SpryAssets folder in your site's root folder (see page 197). Each Spry Validation widget (text boxes, text areas, menus, and checkboxes) has its own external style sheet. Dreamweaver names the style sheet after the type of widget. For example, the style sheet for a Spry Validation text field is named *SpryValidationTextField.css*. If you add several types of Spry Validation fields, you have to edit several different style sheets to change the look of the error messages.

Fortunately, you don't have to hunt and peck through the .css file to modify an error message style. By using Dreamweaver's CSS Styles panel's Current view, you can easily identify the proper style, and then edit it. Here's how:

1. **Open the CSS Styles panel (see Figure 13-16) if it's not already open.**

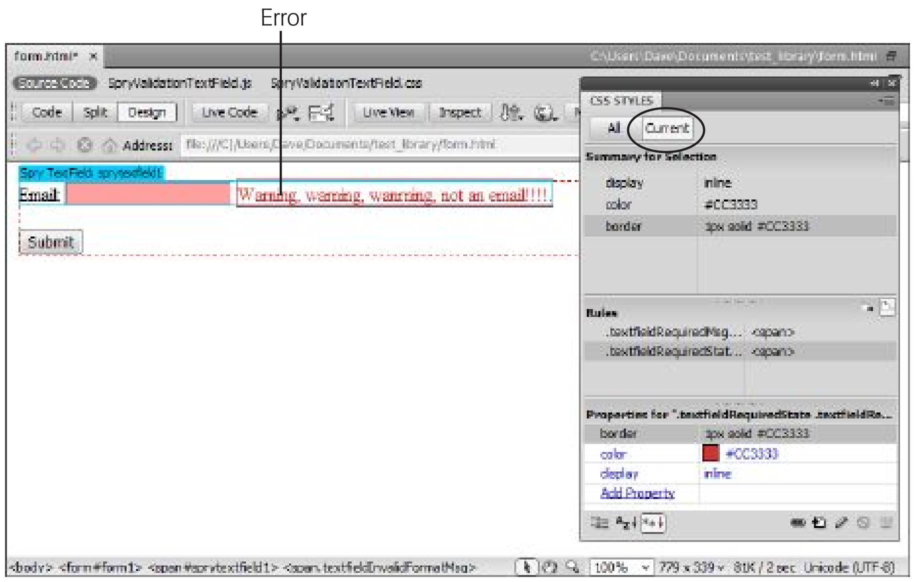
   Choose Window→CSS Styles in either Windows or on a Mac, or Shift-F11 on Windows.

2. **At the top of the panel, click the Current button.**

   The Current view shows the styles and properties that Dreamweaver applies to the selection in the document window.

3. **Make sure you select the Cascade button (circled in Figure 13-16).**

   The Cascade button activates the Rules pane in the middle of the CSS Styles panel. It displays all the CSS styles that Dreamweaver applies to the selection in the order of their specificity—least specific on top, most specific at the bottom (see page 329 for a refresher on specificity). The style that most directly applies to the given selection is listed last.



Warning, warning, warning, not an email!!!!

display	inline
color	#CC3333
border	1px solid #CC3333

.textfieldRequiredMsg... <span>
.textfieldRequiredStat... <span>

Properties for ".textfieldRequiredState .textfieldRe...

border	1px solid #CC3333
color	🟥 #CC3333
display	inline

<body> <form#form1> <span#sprytextfield1> <span.textfieldInvalidFormatMsg> 100% 779 x 339 81K / 2 sec Unicode (UTF-8)

Figure 13-16: *You can easily adjust the formatting of a Spry Validation error message by clicking inside the message, identifying the style in the Rules pane, and then, in the Properties pane, editing the style properties. Or, just double-click the style name to open the friendly CSS Rule Definition window.*

4. **Select the Spry Validation widget.**

   To select the widget, mouse anywhere over the form field in Design view until a blue Spry tab appears; click the tab to select the Spry widget.

5. **From the Property inspector's "Preview states" menu, select the preview state you want to format.**

   Dreamweaver displays the error message that appears for the selected state. In addition, it displays any formatting it applies to the form field in that state; for example, a text field in its "Valid" state has a green background. You can adjust the formatting of the form field as well.

6. **In the document window, click anywhere in the text of the error message or the form field you wish to format.**

   The Styles panel displays the style in the Rules pane (see Figure 13-16). Its name is a rather long-winded group selector (beginning with something like *.textfieldRequiredState, .textfieldRequiredMsg*) made up of classes and descendent selectors (see page 318 for details on group styles). You don't really need to pay attention to the name, however, since you already selected the style you want to edit.

Note: When you format an error message, just click inside the text. Don't try to select the entire message by double-clicking it—if you do, you'll select more than the error message, and the CSS Styles panel won't display the message's style.

7. **Edit the style's properties.**

   You can do this most easily by double-clicking the style's name in the Rules pane; that opens the CSS Rule Definition window, where you can edit CSS properties just as you would any other style, as described on page 129. You can also use the Properties pane for a more rapid edit: To quickly change the text color, for instance (see page 319 for more on how to use the Properties pane to edit and set CSS properties).

Note: Dreamweaver puts Spry error messages inside <span> tags, and displays them inline (meaning on the same line as the form field). If you want to put the error message on its own line, change the *display* property from *inline* to *block* (the tutorial at the end of this chapter includes an example of this trick; see step 15 on page 570).

In addition, you can move the <span> tags containing the error messages, and even change them from a <span> tag to a <div> or <p> tag. The exact tag type doesn't matter—but if you change the tag type (from <span> to <p>, for instance), make sure the class name remains the same. Spry depends on the proper class name to identify the error message. If you move the error message, it must remain inside the outer <span> tag that forms the Spry Validation widget.

A few other styles affect the appearance of Spry form fields, but you can't preview or adjust them using the method just described. For example, when you click in a Spry-enabled text field, its background color changes to yellow; when you click a Spry menu, its background also changes to yellow. Dreamweaver applies these different colors to what's called the field's "focus state"—that's the moment when a visitor interacts with the field. The styles to control these focus states are:

- **Text field focus style**: *.textfieldFocusState input, input.textfieldFocusState*. You'll find it in the *SpryValidationTextField.css* file.
- **Text area focus style**: *.textareaFocusState textarea, textarea.textareaFocusState*. Located in the *SpryValidationTextarea.css* file.
- **Menu focus style**: *.selectFocusState select, select.selectFocusState*. Found in the *SpryValidationSelect.css* file.

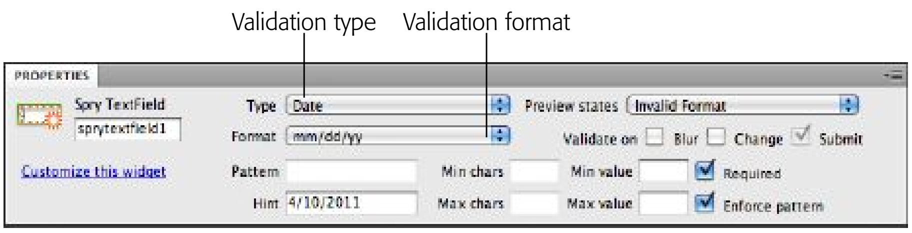
In addition, another special style formats text fields and text area boxes when a visitor presses an invalid key on the keyboard (see the Tip on page 554).

Tip: Dreamweaver's Live View can also help you style CSS elements controlled by JavaScript (like the Spry Validation widgets). Page 642 has more information.

Spry Text Field

Spry text fields have the most options of any Spry Validation widget. Dreamweaver lets you choose from 14 validation types, and lets you control several other settings, such as limiting the minimum and maximum number of characters allowed.

First, decide whether you want to *require* your visitor to enter information in a field; if so, turn on the Required box in the Property inspector (circled in Figure 13-17). A form almost always requires *some* information (an email address to sign up for a newsletter, for instance). But sometimes you want to make a response optional, such as when you ask for a phone number. In a case like that, you don't want to turn on the Required box, but you do want your guest to format the information accurately. That's where validation types come in.



Spry TextField: sprytextfield1
Type: Date
Format: mm/dd/yy
Preview states: Invalid Format
Validate on: ☐ Blur ☐ Change ✓ Submit
Pattern:
Hint: 4/10/2011
Min chars:
Max chars:
Min value:
Max value:
✓ Required
✓ Enforce pattern

Figure 13-17: *Spry Validation text fields provide multiple options for verifying visitor input.*

Validation types

To make sure your visitors supply answers in the appropriate format on your forms, you validate the contents of a field. Use the Type menu (see Figure 13-17) in the Property inspector to assign one of 14 validation options:

- **None**. This option is the default setting. The JavaScript code Dreamweaver creates doesn't inspect the contents of the field to make sure it matches a particular format. Use this setting in combination with the "Required" checkbox when you don't care what someone types in, so long as they enter something. You might use this setting to capture the reason a customer wants to return a product, for example.

Note: When you assign any validation option other than "None," Dreamweaver adds an "Invalid format" error message to the page. You can change this message as described in step 6 on page 545.

- **Integer**. Use this option to verify that your guest entered a whole number, like one asking for their age or year of birth. If someone types in 1.25, JavaScript won't submit the form; it displays an "Invalid format" error message. (If you *do* want to allow decimal values, use the Real Number option discussed below.)

  If you specify integer validation, you can also assign minimum and maximum allowed values, as discussed on page 554.

- **Email**. This option looks for a validly formatted email address (like *missing@sawmac.com*). It can't verify that the email address is real, so someone could enter a fake address (like *nobody@nowhere.com*), but this option at least makes sure an honest visitor doesn't enter a typo.

- **Date**. When you require visitors to enter a specific date, use this option. If you create a form that schedules the use of a meeting room, for example, you could add a "Date needed" field. Dates comprise a month, day of the month, and year, which visitors can enter in many ways: 12-02-2011, 12/02/11, 02.12.11, and so on. To specify the format you want, use the Property inspector's Format menu (see Figure 13-17).

Note: With date validation, the option *yyyy* means visitors have to enter the full year (2011) to pass validation. However, *mm* and *dd* both allow single-digit values, like 1 for January, or 2 for the second day of the month—guests don't need to enter an initial zero (01 or 02, for example).

You specify the format for a month by *mm*, a day by *dd*, and the year by either *yy* (for just the last two digits of the year: 11) or *yyyy* (for a complete year: 2011). You should also indicate the kind of separator you want, like a backslash (/) or a hyphen (–), between the month, day, and year values. So, for example, the option *mm/dd/yyyy* means that 1/2/2011 and 12/15/2011 are both valid entries, but 1-2-2011 or 12/15/11 are not.

Tip: If you'll eventually store the responses from a date form field in a MySQL DATE field, choose yyyy-mm-dd as the format since it matches the format MySQL uses.

- **Time**. This option validates time entries in one of several formats, such as 12:15 PM or 23:15. You can use it along with the date field to capture the exact time of an event: You could include time fields so guests can specify a beginning and an ending time on a meeting room scheduling form, for example. As with the Date format, Time validation requires that you specify a format using the Property inspector's Format menu. *HH* indicates the hour using 24-hour time—13 for 1 p.m., in other words; *hh* is the hour using nonmilitary time; *mm* represents minutes; *ss* indicates seconds; *tt* means before noon and after noon in AM or PM format, and *t* indicates the same thing using just a single letter, A or P.

  So, the HH:MM option validates 13:35, but not 1:35 PM (guests must enter a zero for single-digit hours, minutes, and seconds). The hh:mm:ss tt option requires visitors to format time like this: 01:35:48 PM.

Note: Whenever you require a visitor to type information in a specific format—12:45 PM, for instance—be sure to include clear instructions. Something like, "Please enter the time you'd like to reserve using this format: 12:45 PM." You can also take advantage of a Spry text field's Hint setting, as described on page 554. You can prevent visitors from entering invalid letter, numbers, or symbols using the "Enforce Pattern" option described on page 553.

- **Credit Card**. An e-commerce site isn't much good if you don't give people a way to pay for their purchases. To make sure visitors enter a validly formatted credit card number, choose this option. If you accept only one type of credit card—like Visa or MasterCard—you can specify it using the Property inspector's Format menu. As with email addresses, the validation checks only that someone has correctly formatted the number—it doesn't actually check to see if this is a real (and not stolen!) credit card.

Note: Be careful if you accept credit card numbers online. An awful lot of responsibility goes along with taking someone's credit card number, including potential liability if the card is stolen, or someone manages to steal the credit card numbers you collect. For an introduction to online payment processing, check out *http://tinyurl.com/5vbvkvy.*

- **Zip Code**. To mail a brochure, t-shirt, book, or any other product, you need a Zip code. Use the Zip Code validation format to make sure guests format it correctly. The Format menu lets you specify a Zip code type and country. For example, US-5 means you want to see a five-digit US Zip code, like 97213, whereas US-9 is the nine-digit US Zip code format, composed of five digits, a hyphen, and four more numbers: 97213-1234. Dreamweaver also offers Canadian and UK Zip code formats, and you can create your own by specifying a custom pattern (see the next section).
- **Phone Number**. The US/Canada phone number format looks like this: (555) 555-1234, with the parentheses, space, and hyphen required. If you'd like a different format (555-555-1234, for example), you can define a custom pattern in the Pattern field, as described in the next section. For an alternative style, see page 553.
- **Social Security Number**. This option requires three numbers, a hyphen, two numbers, a hyphen, and three more numbers, like this: 555-12-4888. You should avoid requesting Social Security numbers. Many people are reluctant to disclose them for reasons of privacy and fear of identity theft, and by law, they don't have to.
- **Currency**. If you require someone to specify a monetary amount in a field—"How much money would you like to contribute to the home for wayward web designers?"— select the currency option to validate their responses. You can choose US or European formatting. US format appears like 1,000.00, while the same value in European format is expressed as 1.000,00. The comma (period for the European value) that indicates "thousands" (1,000) is optional; JavaScript considers both 1000.00 and 1,000.00 valid. However, it doesn't accept an opening dollar sign; if a visitor enters $1,000.00 into a currency field, she gets an "Invalid format" error message.
- **Real Number/Scientific Notation**. To allow decimal points in a field intended to capture numeric values, use this option. For a serious, scientific audience, this format even allows scientific notation: 1.231e10.

- **IP Address**. Since we all like having people type the unique set of numbers that identify a computer on the Internet, you can make sure a form accepts only properly formatted IP addresses (like 192.168.1.1). The Format menu lets you choose between the current IPv4 and (the newer, not yet fully implemented) IPv6, or both—oh, please, do people really go around asking for people's IP addresses?
- **URL**. Make sure visitors enter properly formatted web addresses using this option. The address has to include the protocol (http://). So *http://www.sawmac.com* is valid, but *www.sawmac.com* isn't.
- **Custom**. If you're unhappy with the validation options Dreamweaver offers, you can create your own. That's described next.

Custom validation

If you need information entered into a field in a very precise way and none of Dreamweaver's validation types fit the bill, you can create your own validation format. Say your company has an internal ID system for employees. Each employee is assigned an ID composed of three numbers, a hyphen, and the first three letters (in uppercase) of the person's last name, like 348-MCF. To enforce this format, you can create your own validation "pattern." If a visitor's input matches the pattern, JavaScript considers the information valid and submits the form. If the input doesn't match the pattern, it displays an error message.

A pattern is just a series of symbols that indicate acceptable input; each letter in the pattern has a special meaning that defines the valid character type. AAA means "Accept three uppercase letters in a row as valid."

To create a custom validation, select a Spry text field widget and, from the Property inspector's Type menu, choose Custom. Then, in the Pattern field, type the pattern you want (see Figure 13-18).

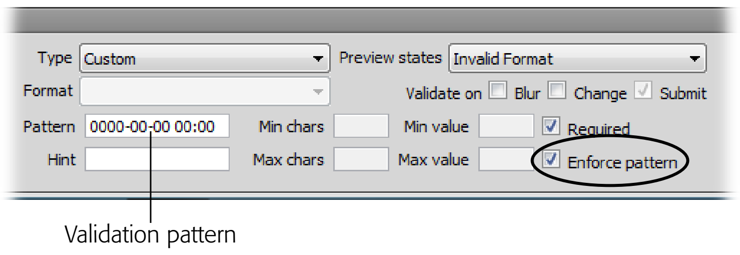

Type: Custom
Preview states: Invalid Format
Format:
Validate on: ☐ Blur ☐ Change ✓ Submit
Pattern: 0000-00-00 00:00
Min chars:
Min value:
✓ Required
Hint:
Max chars:
Max value:
✓ Enforce pattern

Figure 13-18: *Are Spry's validation types not enough for you? Create your own by devising a pattern that a form field's input must match in order to validate.*

Here's a key to the symbols you use to create a pattern:

- **0 means a whole number between 0 and 9.** If you want to make sure that someone enters five digits, type *00000* in the Property inspector's Pattern field. This pattern is the same as the one for a five-digit Zip code.

- **Type *A* to indicate a single uppercase alphabetic character.** The pattern, A0A, for instance, is good for an uppercase letter, followed by a number, followed by another uppercase letter, like U5U.
- **A lowercase *a* identifies a lowercase alphabetic character.** The pattern *aaa*, then, matches *abc*, but not *ABC*.
- **To accept either an uppercase *or* a lowercase letter, use *B*.** The pattern *BBB* matches both *abc* and *ABC*.
- **To include numbers along with uppercase letters, use *X*;** the letter *x* matches both numbers and lowercase alphabetical characters. **Use *Y* for a case-insensitive match for numbers and letters.** *XXX* matches *B2B, BBB* and *123*, but not *b2b* or *bbb*. To match b2B or bb1, use YYY as the pattern.
- **Finally, use *?* as a kind of wild card.** It stands in for any character whatsoever, and you should use it when a character other than a letter or number (like a period, !, or $ symbol) is also valid.

You can include any required symbol, like a period, comma, or hyphen, as part of the pattern. In the employee ID example discussed at the beginning of this section, the pattern to match that format is 000-AAA. In other words, three numbers, a hyphen, and then three uppercase letters. To match a phone number like 503-555-1234, use the pattern 000-000-0000. To match the MySQL DATETIME format, use 0000-00-00 00:00:00.

Enforcing a pattern

You can make sure visitors can't even type in incorrect characters by turning on the "Enforce pattern" checkbox in the Property inspector (circled in Figure 13-18). When you select this option, JavaScript prevents guests from entering invalid characters in the form field. It's a very useful way to prevent visitors from entering incorrect information in the first place.

For example, suppose you add a Spry text field and set its validation type to Zip code, using the US-Zip5 format. That box can accept only digits, and only five digits at that. If you turn on the "Enforce pattern" option for this field, a visitor could type only five numbers into the field. If a visitor types the letter A, the field remains blank. If the visitor types five numbers and then any other character (like another number or even a letter), that sixth character never appears.

For some validation formats, the "Enforce pattern" feature is even more useful. For example, with a US/Canada phone number, the Spry validation expects a number in this format: (503) 555-1212. Because the only part of the number that will vary from visitor to visitor is the actual numbers—not (,), -, or spaces—the Spry programming automatically adds those so visitors only needs to type in the numbers, not any of the required punctuation.

You can choose the "Enforce pattern" option for any validation type except *None*. It even works with custom patterns.

Tip: When someone types invalid characters into a form field that has the "Enforce pattern" option set, any text inside the box flashes bright red to indicate a problem. If you want to change that color, you can edit the styles responsible. For text fields, in the *SpryValidateTextField.css* file, the style is a group selector named *.textfieldFlashText input, input.textfieldFlashText*. In the *SpryValidationTextarea.css* file, a similar style named *.textareaFlashState textarea, textarea.textareaFlashState* applies to text fields.

Supplying a hint

When you require a very specific format for a form field, you should provide clear instructions on how visitors should fill it out. You can have these instructions appear next to the label or below the form field.

Dreamweaver also lets you add a short "hint" inside a Spry text field. This hint appears when the form first loads, but the moment a visitor clicks into the field, it disappears; visitors are then free to type in a response.

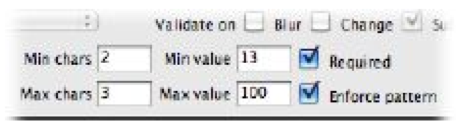
To add a hint, select the Spry widget, and then, in the Property inspector's Hint field, enter what you want to appear (see Figure 13-17).

Since text fields are relatively short, you don't have much room for instructions. A better use of the hint is an example of the format the field requires. If you want to collect an email address, for example, make the form field hint something like *your_email@your_site.com*. If you're looking for a phone number and use the phone number validation type, add a sample phone number, like (555) 555-1234. This lets visitors know that they should include the parentheses and hyphen.

Limiting characters and enforcing a range of values

At times, you may want to control the amount of text someone types into a field. If you create a member profile form as part of your "members-only" website, you might want to collect a person's age—so you want an integer that's at least two numbers long (no babies allowed!), but no more than three numbers long (no immortals either!). As you've read, you can control the maximum number of characters in a text field with HTML's "Max chars" property (see page 527). However, HTML gives you no way to require *a minimum* number of characters. In addition, setting the "Max chars" property doesn't alert a visitor when she's typed the maximum number of allowable characters.

With Spry text fields, you can set both, using the Property inspector's "Min chars" and "Max chars" fields (see Figure 13-19). Select the Spry widget by clicking the blue Spry tab that appears when you mouse over the Spry text field, and then set the minimum and maximum number of characters in the "Min chars" and "Max chars" fields. You can fill in either field, both fields, or neither. In the age example above, in the "Min chars" box, you'd type *2* and in the "Max chars" box, you'd enter *3*.



Validate on ☐ Blur ☐ Change ✓ Su

Min chars 2 Min value 13 ✓ Required

Max chars 3 Max value 100 ✓ Enforce pattern

Figure 13-19:
Go ahead, be a dictator. The Spry text field validation widget even lets you control how many characters someone can type into a field.

Each setting has its own error message. You can view and edit them by choosing the appropriate state from the "Preview states" menu (see step 6 on page 545). For example, while the error message for the minimum number of characters reads "Minimum number of characters not met," you can change that to something more descriptive, like "You're too young to join our club."

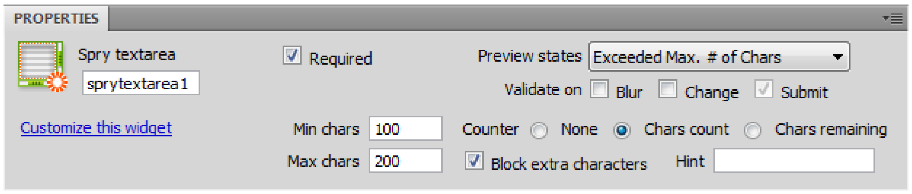
Some validation fields also let you enforce a *range* of values. If you select the Integer validation type (see page 549), the "Min value" and "Max value" boxes become active in the Property inspector. Say you include a question on a form that reads, "Please rate the quality of our service from 1 to 10," and supply a text box for a response. In this case, set the "Min value" to *1* and the "Max value" to *10*; that way, JavaScript won't allow answers like 100, or –10.

You can set Min and Max values for other numeric validation types, too, like currency (page 551) and real numbers (page 551). These two settings even work with the date and time validation types (page 550). Say you offer rebates to anyone who buys your product before a certain date—08/05/2012, for instance; the online rebate form includes a "Date purchased" field. In this instance, you can choose the Date validation type from the Format menu, select mm/dd/yyyy, and then, in the "Max value" field, type *08/05/2012*. If someone who buys the product on September 15, 2012, tries to claim the rebate, he gets an error message when he fills out the form.

Note: Setting a minimum and maximum value for a text field that uses Time validation works reliably only if you use 24-hour time (like 18:00 for 6:00 PM). If you use one of the formats that requires the AM or PM notation, you can end up with inaccurate results. Spry treats 12:00 PM (noon) as later than 5:00 PM, and 8:00 AM as earlier than 12:00 AM (midnight).

Spry Text Area

A Spry text area has far fewer validation options than a normal text field. You can't select a type of validation or enforce a pattern on the text box's contents. However, the Property inspector does let you specify whether content is required, dictate the minimum and maximum number of characters allowed, and supply a hint that appears inside the text box when the form page loads (see Figure 13-20). (These options works just like those for a text field, described on page 554.)



PROPERTIES

Spry textarea: sprytextarea1 — Customize this widget

Required: ✓ — Preview states: Exceeded Max. # of Chars

Validate on: Blur ☐, Change ☐, Submit ✓

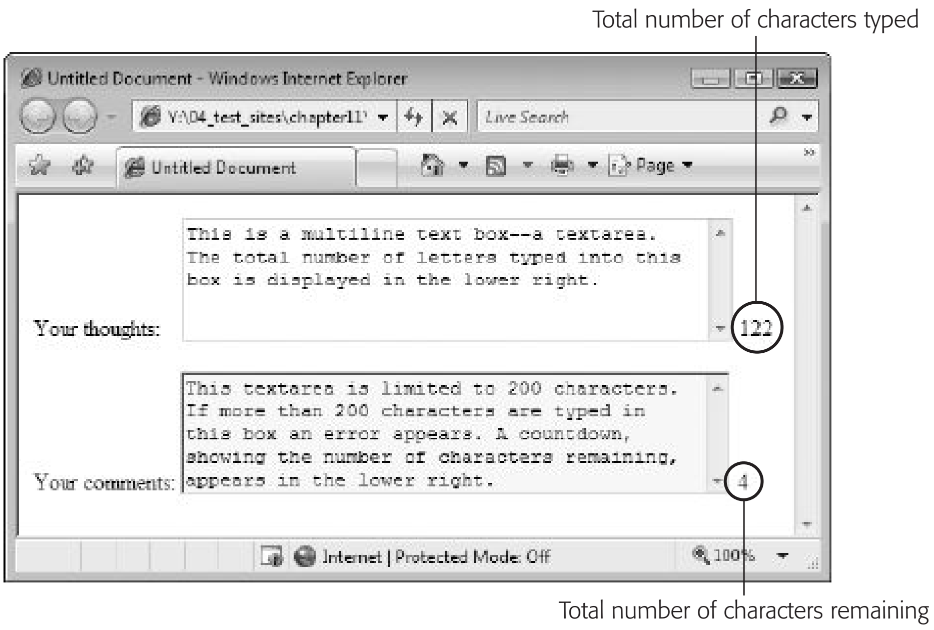
Min chars: 100 — Max chars: 200

Counter: None ○, Chars count ●, Chars remaining ○

Block extra characters: ✓ — Hint:

Figure 13-20: *If you turn on the "Block extra characters" checkbox after you set a value in the "Max chars" field, anything someone types beyond that maximum character limit doesn't appear. HTML gives you no way to limit the amount someone can type into a multiline text box, so this Spry feature offers a nice workaround.*

In addition, you can include a counter alongside the text area that tells your guest either how many characters they've entered (turn on the "Chars count" radio button you see in Figure 13-20), or how many more they *can* enter before they hit the limit ("Chars remaining"). That's helpful if you limit the amount of feedback a visitor can type in; you can include a message like "Please limit your feedback to under 300 letters" and either tally up the number of characters your guest enters or count down the number to zero (Figure 13-21).



Total number of characters typed
Untitled Document - Windows Internet Explorer
Y:\04_test_sites\chapter11
Live Search
Untitled Document
Page
Internet | Protected Mode: Off
100%
Total number of characters remaining

Your thoughts: This is a multiline text box--a textarea. The total number of letters typed into this box is displayed in the lower right. 122

Your comments: This textarea is limited to 200 characters. If more than 200 characters are typed in this box an error appears. A countdown, showing the number of characters remaining, appears in the lower right. 4

Figure 13-21: *A Spry text area can include feedback regarding how many letters a guest types into a multiline text box (top). It's also possible to display a countdown that shows how many more letters your visitor can type before they reach the limit (bottom).*

Both of these counters are helpful, but neither gives your guest a context for the number: 300? 300 what? Letters remaining, or letters already typed into the box? To add a clarifying message next to the number, you have to go into Code view. The best method is to select the text area field in Design view, click the Code or Split view button, then find a <span> tag that looks something like *<span id="countsprytextarea2">&nbsp;</span>*. In this example, "sprytextarea2" is the Spry widget's name. You must add your message either before or after the <span> tag, but not *inside* the tag. The code above could be changed to:

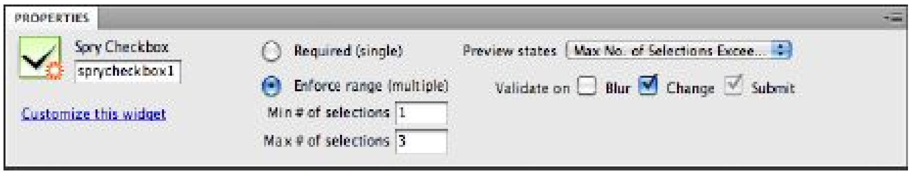
```
<span id="countsprytextarea2-id001">&nbsp;</span> characters remaining
```

for a text area with the "Chars remaining" option turned on. This way, a visitor sees something like "300 characters remaining."

Spry Checkbox

The Spry Validation checkbox lets you make sure that a visitor turns on a checkbox, an especially handy tool for those ubiquitous "I agree to your rules and conditions" disclaimers. In addition, you can add several checkboxes as a group, and require that your guest select a minimum number of options ("Please make at least two choices") or a maximum number ("Please choose no more than two").

To add a single Spry checkbox, choose Insert→Form→Spry Validation Checkbox, or, in the Insert panel's Forms category, click the Spry Validation Checkbox button (Figure 13-22). The Spry checkbox that appears on the page already has the "Required" option selected in the Property inspector. If you want just a single checkbox, you're done. But beyond the kind of "You must turn on this checkbox to free us from all legal responsibility" scenario, a single, required checkbox isn't so useful. After all, checkboxes more commonly come in groups as part of a multiple-choice question.



sprycheckbox1
Required (single) [✓ Enforce range (multiple)]
Min # of selections 1
Max # of selections 3
Preview states: Max No. of Selections Excee...
Validate on: Blur [✓ Change] [✓ Submit]

Figure 13-22: *Use the Spry Validation Checkbox widget to make sure your site visitors turn on a checkbox (in cases where you want that to happen, of course).*

Unfortunately, Dreamweaver doesn't include a simple "Add a group of checkboxes" tool. If you insert several Spry checkboxes in a row, Dreamweaver creates a Spry widget for each one and JavaScript validates each box separately, rather than as a group. Nor can you insert a bunch of regular checkboxes, select them all, and then apply the Spry Validation Checkbox to them.

To create a group of related Spry checkboxes, you either need to go into Code view or execute a delicate keyboard dance to get all the code just right. If you want to stay in Design view, here's a way to insert a group of checkboxes that JavaScript validates together:

1. **Insert a Spry checkbox.**

   Use either the Insert→Form menu or the Insert toolbar. Dreamweaver inserts a checkbox with the familiar blue Spry tab. Add a label in the text field to the right of the checkbox (the "Input Tag Accessibility Attributes" window described on page 523 can do this for you). Now, say you want to add another checkbox to the right of the one you just inserted.

2. **Click anywhere in the label text and then click <label> in the tag selector at the bottom of the Document window (see top image in Figure 13-23).**

   In Code view or Split view, Dreamweaver highlights the current label. Your goal is to make sure your cursor is *outside* the <label> tag for the current checkbox. If you simply click to the right of the label and insert the next checkbox, one of two (bad) things can happen: You insert another checkbox inside the first checkbox's <label> tag—when this happens, Dreamweaver gets really confused and omits a <label> tag for the new checkbox. Or, you insert the checkbox outside the Spry widget, meaning the new checkbox won't be validated along with the first checkbox.

3. **Press the right arrow key until the <label> tag disappears from the Tag selector at the bottom of the document window, but you still see something like <*span#sprycheckbox1*> (Figure 13-23, bottom).**

   When you no longer see <label> in the Tag selector, your cursor is outside the label and you can insert another checkbox. The <*span#sprycheckbox1*> identifies the tag responsible for the Spry checkbox widget. As long as you see that in the Tag selector, the next checkbox you insert receives Spry validation.

4. **Insert a *regular* (non-Spry) checkbox as described on page 528.**

   The cursor is already inside a Spry checkbox widget, so don't insert another Spry checkbox.

5. **Repeat steps 2–4 to insert as many checkboxes as you need.**

   As long as you insert the checkboxes inside the Spry widget, they'll be part of the validation process.

Tip: If you want to put each checkbox in its own paragraph, change the <span> tag that the Spry checkbox validation widget uses to a <div> tag. According to the rules of HTML, you can't wrap a <span> tag around block-level elements like a paragraph. Go into Code view, locate the opening span tag (it should look something like <*span id="sprycheckbox1"*>), and change *span* to *div*. Then locate the closing tag, </span>, and change it to </div>.

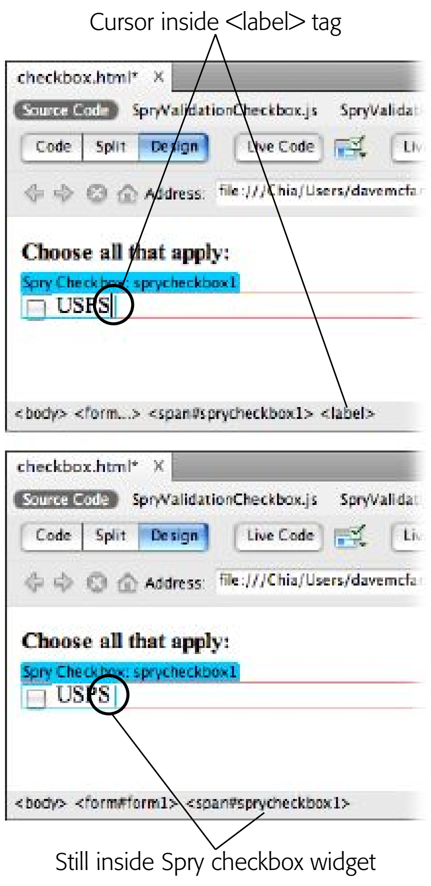

Cursor inside <label> tag
checkbox.html* X
Source Code
SpryValidationCheckbox.js
Code
Split
Design
Live Code
Address: file:///Chia/Users/davemcfa
Choose all that apply:
Spry Checkbox: sprycheckbox1
USPS
<body> <form...> <span#sprycheckbox1> <label>
<body> <form#form1> <span#sprycheckbox1>
Still inside Spry checkbox widget

Figure 13-23:
If you want to include multiple checkboxes in a single Spry Checkbox Validation widget, you need to first insert a Spry checkbox, and then insert additional regular (non-Spry) checkboxes. The trick is making sure you don't insert the new checkbox inside the <label> tag of another checkbox (top), but do insert it inside the Spry Validation widget's code (bottom).

6. **Click the blue Spry tab to select the widget; in the Property inspector, select the "Enforce range" button, and then, in the "Min # of selections" and the "Max # of selections" fields, type *numbers* (see Figure 13-22).**

   You don't have to fill out both the Min and Max fields. If you have a question like "What type of food do you like (select as many as apply)," you might choose 1 for "Min # of selections" but leave the Max field blank. That way, you require at least one choice, but your visitor can choose as many other options as she wants.

   Or, you might have a question like "Select your four favorite foods." In this case, you'd type *4* in the Max field if you don't want more than four answers. (You could also type *4* in the Min field if you want to make sure you get exactly four choices.)

Spry Select

The Spry Validation Select widget validates the choices in pull-down menus, and has two options to determine whether or not a menu selection is valid (see Figure 13-24). Remember that a pull-down menu (which Dreamweaver creates using the <select> tag) consists of a label and a value (see page 533). The label is what someone sees when he makes a selection from the menu, and the value is what the browsers sends over the Internet when it submits the form.

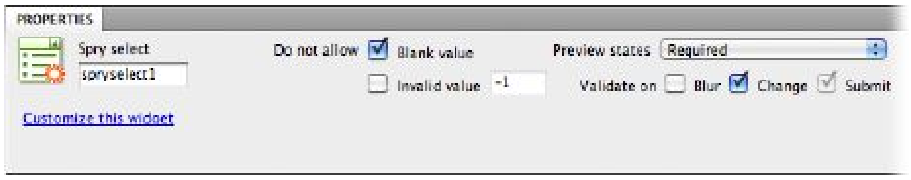

PROPERTIES
Spry select: spryselect1
Do not allow: [✓] Blank value, [ ] Invalid value -1
Preview states: Required
Validate on: [ ] Blur [✓] Change [✓] Submit
Customize this widget

Figure 13-24: Clicking the "Customize this widget" link that appears when you select a Spry widget launches the Adobe Help program, and displays documentation that tells you which CSS styles affect the appearance of the widget. You can use this information to edit the supplied widget styles to make them match the look of your site.

With a Spry menu, if a guest makes no selection, or if she's makes an invalid selection, you can prevent JavaScript from submitting the form, and have it display an error message instead. Say you have a menu listing all the months of the year. The label is the month's name and its value is a number (see Figure 13-11). Suppose you added "Please select a month" as the first item in the menu. This common technique lets visitors know that the menu's a list of months they should select from. Of course, when the browser submits the form, you want it to send the value for a month and *not* "Please select a month."

Note: Although Dreamweaver inserts a pull-down menu when you add a Spry Validation Select widget, you can convert the menu to a static list, as described on page 533. The same validation options apply.

To make sure this is the case, leave the value for "Please select a month" blank and, in the "Do not allow" section of the Property inspector, turn on "Blank value" (see Figure 13-24).

Sometimes a label and a value are the same. For example, on a menu with a list of years ("In what year were you born?"), the label ("1967") is the same as the value ("1967"). In cases like this, it would be frustrating to have to set both the label and value for each menu item. Fortunately, you don't have to. As you read on page 534, a label doesn't require a value, and if you don't specify a value, JavaScript submits the label *as* the value.

If you have a list without values, the "Blank value" validation option won't work. After all, even valid selections (the selected label) won't have a value explicitly set. In cases like this, use the "Invalid value" option in the Property inspector to ensure that the form gets submitted. Here's how you set this option up: First, identify the invalid selection(s). There's just one in the example above, the label at the top of the menu ("In what year were you born?"). Assign an arbitrary value to the illegitimate selection(s). In this case, assign a value of -1 to "In what year were you born?" Then select the Spry widget and, in the Property inspector, turn on the checkbox for "Invalid value" and enter -1 in the field next to it (see Figure 13-24). Should a guest select "In what year were you born?" from the list, JavaScript recognizes its value as invalid and prevents the form from winging its way to your server. (If this sounds confusing, you'll find a hands-on example in the tutorial on page 575.)

Note: If your form menu has a long list of options, you might add a separator (like a row of hyphens, ----------) as a label, to demarcate groups of options. You could either forego assigning a value to that separator and use the "Blank value" validation option, or assign it an invalid value (like –1) and use the "Invalid value" setting. Now, if someone accidentally selects the separator, she can't submit the form.

Spry Password

A password like "sesame," "password," or "bob" isn't very secure. Any hacker with a dictionary (or access to an infinite number of monkeys) can easily infiltrate a password-protected web page, or gain access to someone's personal information. If you ever add a sign-up form that requires someone to come up with a password, use Dreamweaver's Spry Password widget. This helpful tool lets you enforce a set of rules for password names so that visitors don't create easily hacked credentials. For example, you can say that a password must be at least eight characters long, have at least three numbers, and contain a minimum of two uppercase letters. This kind of password-naming strategy means visitors have to come up with hard-to-crack passwords like AB3859kirI.

Note: Use the Spry Password widget only for forms where a visitor creates a password. Don't use it for a form where a visitor logs in with an already created password. After all, there's no point in telling a visitor that she needs a certain number of letters or numbers in her password if she already has a valid password.

To add a Spry password field, click the area in the form where you want to add the field, and then, on the Insert Panel's Forms category, select the Spry Validation Password button or choose Insert→Form→Spry Validation Password. Then, just as with any form field you insert, the Input Tag Accessibility window opens; follow steps 3–5 on page 523 to insert the field.

Note: If you want to turn a text field in a form into a Spry password field, you first need to make sure the password option is turned on for that field (see page 527). Then select the field and click the Spry Validation Password button in the Insert Panel or choose Insert→Form→Spry Validation Password.

Once inserted, the password field has a blue Spry Password tab, and the Property inspector shows the options for validating the field (see Figure 13-25). Since one of the goals of a good password is to make it hard to figure out, the validation options for the password widget try to enforce a pattern that's essentially a random collection of numbers, letter, and characters. The Min and Max characters options let you specify the length of a password. You can set either or both of these options, but at the very least, you should specify a minimum number of characters—8 is a solid amount—so that no one creates an easily hacked password like 1, A, or A1.

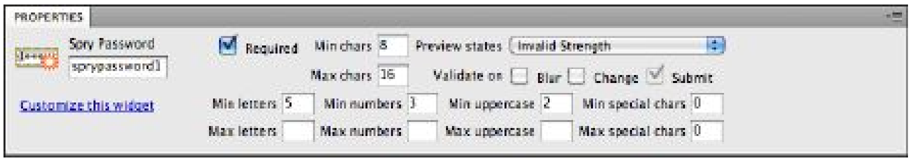

PROPERTIES
Spry Password: sprypassword1
[✓ Required] Min chars 8 Preview states Invalid Strength
Max chars 16 Validate on Blur Change [✓ Submit]
Min letters 5 Min numbers 3 Min uppercase 2 Min special chars 0
Max letters Max numbers Max uppercase Max special chars 0
Customize this widget

Figure 13-25: *Use Dreamweaver's Spry Password Validation widget to make sure that new visitors creating a password for your site make it suitably random and difficult to crack.*

In addition, you can specify the *types* of characters visitors must include in their password. For example, you might decide that passwords should have at least four letters, two numbers, and one special character (like an exclamation point). That rule would make a password like ABCDE38! valid, but wouldn't let someone create a password like ABCDEFGH, or 12345678. You can even dictate that passwords have a certain number of uppercase letters as well, just to mix things up. For example, say you type 6 in the "Min letters" box; this rule means a valid password needs at least six letters in it. But to make sure there's a good mix of upper- and lowercase letters, you could set the "Min uppercase" value to 2, and the "Max uppercase" value to 4. This would mean that a visitor has to include at least a few uppercase and lowercase letters in the password.

Depending on which validation options you select for the password widget, you can customize up to four different error messages: the "Required" message, which your page displays when a visitor leaves the password field blank and tries to submit the form; the "Min # of characters" message that appears when you specify a value in the "Min chars" box and your guest uses fewer characters; the "Max # of characters" message that appears when your visitor exceeds the specified number ; and, finally, the "Invalid Strength" message that appears if a guest types in a password that doesn't match the options you set ("Min letters" or "Min numbers" for example).

The "Invalid Strength" error message that Dreamweaver supplies—"The password doesn't meet the specified strength"—doesn't really tell your visitor what he did wrong. So either change this message to something like "Please type a password that's at least 8 characters long and which contains letters, numbers, and at least one special character, like a period, question mark, or exclamation point" or, even better, provide those instructions on the form to begin with. That way, a visitor won't waste his time trying to decode the runes of your password requirements. (See step 6 on page 545 for instructions on editing Spry form validation error messages.)

Spry Confirm

The Spry Confirm validation widget comes in handy when you want to make sure someone correctly enters important information. For example, if you create a form for people to sign up for your email newsletter, you want to make sure they give you the correct email address. One way to do that is to have a visitor enter the same information twice, by adding a second field that asks her to confirm her address by typing it in again.

Note: You can also use this double-checking maneuver with a "Create a password" field. A password field displays what the visitor types as dots or asterisks, so it's easy to make a mistake without ever realizing it. Adding a second, confirmation field can help make sure the visitor gets it right.

The Spry Confirm widget works only with text fields, and displays an error message if the value in the text field doesn't match the value in another text field on the page. To use this widget, first add a text field—either a Spry text field, a Spry password field, or just a regular text field. That field is the original "Type your email" or "Create a password" box. Next, from the Insert Panel's Forms category, add the Spry Confirm widget, or choose Insert→Form→Spry Validation Confirm. (It's best to put this field directly after the original field, and use a label like "Please confirm your password.")

The options for a Spry Confirm widget are simple (see Figure 13-26). From the "Validate against" menu, simply select the name of the field you're comparing. For example, say you add a Spry Password validation widget, and you named that field *password1*. When you insert the Spry Confirm field, select *password1* in the Property inspector's "Validate against" menu. Then, when someone fills out the form, she must type a password in the *password1* field. That password becomes *password1*'s value. Then, in the confirmation field, she types the same password. The Spry Confirm widget compares the two values. If they're different, the widget displays an error message, letting the visitor know she made a typo.

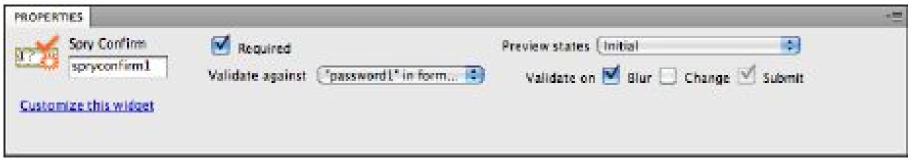


Figure 13-26: *If the field you're comparing against is required, then make sure the "Required" box is turned on for the confirmation field as well. Since the information is so important that you need a second field to confirm it, odds are that it's required anyway, so you almost always leave the Required box checked.*

Spry Radio Group

Sometimes you want to make sure a visitor selects a radio button before she submits a form. For example, say you have an e-commerce site and you collect shipping information from a customer. For the shipping method, you want the customer to select either USPS, FedEx, or UPS. Since visitors have to choose a delivery method in order for you to ship a package, it's a good idea to make sure they click one of the radio buttons before submitting the form. This is where the Spry Radio Group comes in handy. Essentially, it's just like the radio group described on page 533, except that it displays an error message if a guest tries to submit a form without a button selected.

To add a Spry Radio Group, use the Insert panel's Forms category or choose Insert→Form→Spry Validation Radio Group. The process is the same as inserting a regular radio group or a checkbox group. Once you add the group of buttons to a page, use the Property inspector to set the validation options (see Figure 13-27). Dreamweaver gives you several ways to validate a radio group, but only one is really useful: The "Required" checkbox simply means that a radio button must be selected (and it doesn't matter which one).

The other two options—Empty Value and Invalid Value—produce error messages should a visitor select a radio button that you specify. In either of the boxes beside these options, enter the same value you used when you created the buttons. If a guest selects a button with the specified value, he sees one of two error messages when he tries to submit the form. In the case of Empty Value, the error message tells the visitor that he *hasn't* made a selection (huh?); and for the Invalid Value, the error message announces that the choice he made was invalid. Neither of these options seem like they would ever be useful. After all, do you really want to display an error message when someone clicks a radio button that says, in effect, "Ha, ha, you fool, you just released the hounds!"

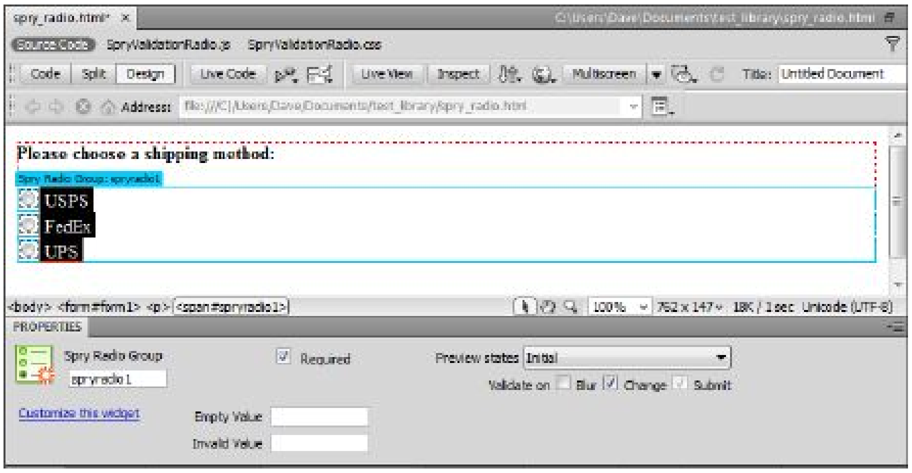


Figure 13-27: *The Spry Radio Group's Property inspector's only useful validation option is Required, which makes sure that site visitors select a button within a group of radio buttons. The Empty Value and Invalid Value boxes are just plain silly—even Adobe's online documentation doesn't provide a good use for them. If you come up with one, send an email to* missing@sawmac.com*.*

Forms Tutorial

In this tutorial, you'll build a simple appointment sign-up form for the Chia Vet website (skip ahead to see Figure 13-38 if you want to see the final result). To make sure the folks at Chia Vet get the right information, you'll use the Spry form validation tools.

Note: To complete this tutorial, you need to download the tutorial files from *www.sawmac.com*. See the Note on page 47 for more details.

Once you download the tutorial files, open Dreamweaver and define a new site as described on page 39. Name the site *Forms*, and then select the Chapter13 folder (inside the MM_DWCS5.5 folder). (In a nutshell: Choose Site→New Site. In the Site Definition window, type *Forms* into the Site Name field, click the folder icon next to the Local Site Folder field, navigate to and select the Chapter13 folder, and then click Choose or Select. Finally, click OK.)

Insert a Form and Add a Form Field

The first step in building a form is inserting a <form> tag. The tag encloses all the fields within a form, and indicates where the form begins and ends. As noted earlier in this chapter, you can insert other HTML elements, too, like text elements and <div> tags.

1. **Choose File→Open. Double-click the file *appointment.html* in the chapter13 folder to open it.**

   If you have the Files panel open (Window→Files), just double-click *appointment.html*. The page is partly designed, with a banner, sidebar, and footer.

2. **Click the empty white space directly below the paragraph that begins "Make an appointment with *Chia-Vet.com* 24 hours a day." On the Insert panel, select the Forms category (see Figure 13-28).**

   The Insert panel shows you the forms icons you need.

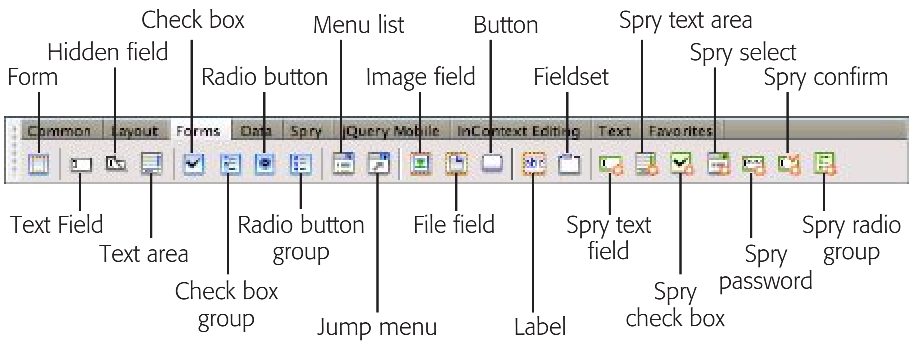


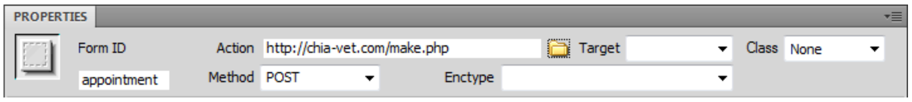
Figure 13-28: *Normally, the Insert panel appears at the right of the screen, with the other panels (Files, CSS Styles, and so on). But if you choose the "Classic" workspace layout as described on page 37, the Insert panel becomes the Insert bar and moves to the top of the screen. Click the "Forms" tab to see all the Form objects. If your monitor is wide enough (most are), the Classic view is usually the better way to go – it frees up space in the right side of your monitor for the CSS and Files panel.*

3. **Click the Insert panel's Form button (see Figure 13-28), or choose Insert→Form→Form.**

   A red, dashed rectangle appears in the document window, indicating the boundaries of the form.

4. **In the Property inspector's "Form ID" field, replace *form1* with *appointment* (see Figure 13-29).**

   You just added an ID to your form.



Form ID: appointment
Action: http://chia-vet.com/make.php
Target:
Class: None
Method: POST
Enctype:

Figure 13-29: *The Action property of a form is simply a URL pointing to the program that processes the form.*

5. **In the Action field, type *http://chia-vet.com/make.php*.**

   As shown in Figure 13-29, leave off the period after the URL in the sentence above (we added it to make our copy editors happy).

   A form's Action property identifies the Internet address of the program that processes the form's data. In this case, you've been spared the effort of writing (or hiring a programmer to write) the required form-processing software. Such a program already exists on the website whose address you just specified, and it's waiting to process the form you're about to design.

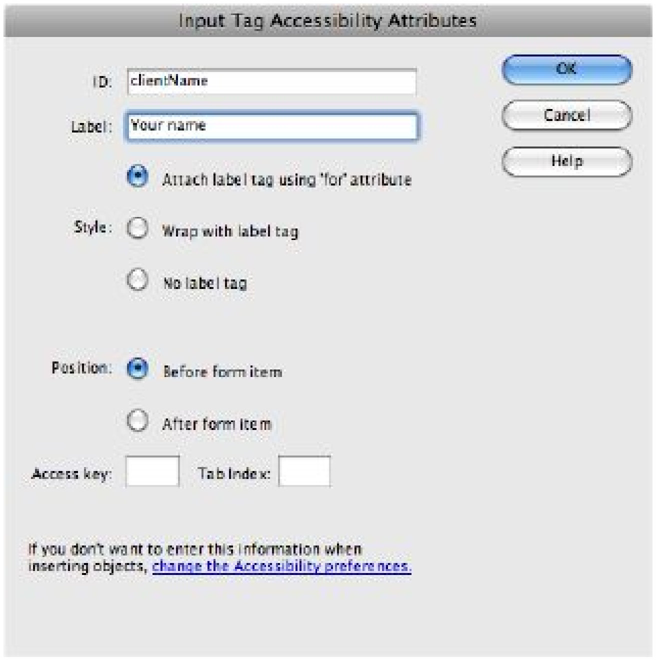
6. **In the Method menu, make sure you see POST selected. Leave the Target and Enctype fields blank.**

   The Method specifies how a form sends information to a form-processing program, and the POST option is the most common (see page 520).

   Now you're ready to insert a text field.

7. **In the document window, click inside the form—anywhere within the red dashed lines. On the Insert panel, click the Text Field button or choose Insert→Form→Text Field.**

   Dreamweaver displays the Input Tag Accessibility window (see Figure 13-30). If you don't see it, choose Edit→Undo to remove the text field you just inserted, use the Preferences window to turn on the accessibility options for form objects as described in Figure 13-5, and then repeat this step.



ID: clientName
Label: Your name
Style: [✓ Attach label tag using 'for' attribute] Wrap with label tag No label tag
Position: [✓ Before form item] After form item
Access key: Tab Index:

If you don't want to enter this information when inserting objects, change the Accessibility preferences.

Figure 13-30:
The Input Tag Accessibility Attributes window gives you a great way to quickly insert a bunch of form-related HTML.

8. **In the ID box, type *clientName*.**

   The widget adds the name you type to both the *name* and *ID* properties of the field's HTML. The form-processing program uses the *name* property to connect an ID with the value a visitor types in. In this case, when a client types his name into the text field, the form-processing program receives information (*clientName=Bob*, for instance) in what's called a name/value pair (see 515).

   The ID uniquely identifies the form element. If you want, you can create an ID style to format this particular form field—for example, to assign a width or background color to this one field. Next, you'll add the label that appears along with the text field on the web page.

9. **In the Label box, type *Your name* and select "Attach label tag using 'for' attribute." Then select the "Before form item" button.**

   The window should now look like the one in Figure 13-30.

10. **Click OK to insert the text field.**

    The label and text field appear side by side on the page. The label's text resides inside the HTML <label> tag (page 539). Your first order of business is to dictate how wide you want the text field to be.

11. **Click inside the newly inserted field (the box on the page); in the Property inspector's "Char width" field, type *35*.**

    This action defines the box's onscreen width in characters, so this box displays up to 35 letters (though a visitor can actually type in more than 35 letters, as described on page 526).

    Now it's time to add some style.

Note to Mac owners: Dreamweaver occasionally exhibits a frustrating bug when working with form fields and the Property inspector. Sometimes, when you select a form field and then click inside a box in the Property inspector (like the "Char width" field), you see some gobbledygook (pardon the hi-tech jargon) appear in the Property inspector–frequently, some information from a previously selected form field. When this happens, just click back into the document window, select the form field a second time, and then click into the box in the Property inspector.

12. **Make sure you have the CSS Styles panel open (choose Window→CSS Styles); on the Styles panel, click the + button to create a new style.**

    Alternatively, you can choose Format→CSS Styles→New. Either way, the New CSS Style window opens. (If you need a refresher on creating styles, see "Creating Styles" on page 117.)

13. **If it's not already selected, choose Compound from the top menu. Delete whatever is currently in the Selector Name box, and type in *#appointment .question*.**

    The descendent selector (page 315) *#appointment .question* contains a new class style named *.question*. The point of this style is to let you create a unique look for each label in this particular form. The formatting you're about to assign applies only to an element with the *.question* class applied to it that also happens to be inside *another* tag with an ID of *appointment*. Since, in step 4 above, you gave the form itself the ID *appointment*, this style applies only to tags with the *.question* class that are inside this form. If you want to use this style on other forms, you can simply create a class style named *.question* and use it throughout the site. But in this instance, you want to create a distinct look for just this form (and you also want to get some practice creating descendent selectors).

14. **From the bottom menu, select *global.css* and then click OK.**

    The CSS Rule Definition window appears.

    This page already has an external style sheet named *global.css*. In fact, it has several style sheets—one for a Spry Navigation Bar (page 192), one for a Spry collapsible menu (page 596), and one for the page's layout (using one of Dreamweaver's CSS layouts, described on page 365). The *global.css* file defines the basic format for the pages on the site, so you should add this style to it.

15. **In the Rule Definition window's Type category, from the Font-weight menu, choose "bold." Then select the Block category, and, from the Display menu near the bottom of the window, choose "block."**

    The block option formats a tag as a block-level element—it adds a line break above and below the element. You use this option to change the display of a tag that would normally appear "inline" (side by side) with other elements. For example, the <label> tag is an inline element and, in the form you're working on, the label "Your name" appears directly to the left of the text field. By applying this style, with its *block* format, to that label, you force the label to appear above the field. Positioning the label this way isn't any kind of requirement for forms, it's just a design choice to make the form more readable.

    Next, you'll add a border above the label to visually separate it from the element above it.

16. **Select the Border category and turn off the three "Same for all" checkboxes. From the Top Style menu, choose "dashed;" in the "top Width" box, type *1*; and then, in the "top Color" box, type *#9BBF13*.**

    Finally, you'll add a bit of padding to separate the border and the label. You'll also add a top margin to create a bit more space between form elements.

17. **Select the Box category. Turn off the two "Same for all" checkboxes and, in the Padding Top box, type *5*; in the Top Margin box, type *15*. Click OK to complete the style.**

    Because this is a class style, you must apply it manually.

18. **In the document window, click anywhere inside the label "Your name." In the Tag selector, click <label> (circled in Figure 13-31) to select the label tag. In the Property inspector, make sure the HTML button is selected, and then, from the Class menu, choose "question."**

    The form field drops below the label. Next you'll add a field to collect the patient's name.

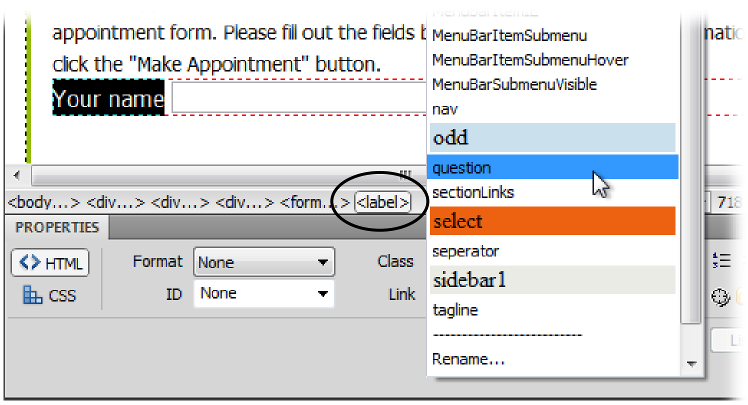


Figure 13-31: *The Tag selector and the style menu are a great one-two punch for precisely applying a CSS class style to any tag.*

Note: After step 17, you might wonder why you didn't just create a tag style for the <label> tag instead of a class style you had to manually apply. Good question. In some cases you can, but if you add checkboxes or radio buttons to a form, you can't always go that route. That's because these form fields usually have their own label (the text that appears next to it)–for example, "Select a shipping method" or "Pick your four favorite fruits." That text isn't in a <label> tag since it's not associated with any single form field. So creating a class style–*.question*–lets you apply the same CSS style–and therefore the same format–to different types of tags.

19. **In the document window, click to the right of the text field (which now sits underneath the text "Your name"). Hit Enter (Return) to create a new paragraph.**

    Now, you'll add a form field to collect the name of the Chia pet.

20. **On the Insert panel, click the Text Field button, or choose Insert→Form→Text Field. Repeat steps 8-11 to insert this new field. Type patient for the field ID, and Your Chia Pet's Name for the label.**

    You added a second text field—now it's time to add a little style.

21. **Repeat step 18 to apply the *.question* class to the new field's label.**

    So much for regular text fields; now it's time to add a little Spry Validation.

Adding a Spry Validation Text Field

Next up for this form: a phone number field. The vets at Chia Vet need a way to contact clients in case they have to reschedule an appointment, so it's important that clients fill in this field. In addition, you want to make sure the customer doesn't enter too few or too many numbers. A Spry Validation text field is the perfect solution.

1. **In the document window, click to the right of the text field you added in the last part of this tutorial. Hit Enter (Return) to create a new paragraph.**

   On the Insert panel's Forms tab, click the Spry Validation Text Field button (see Figure 13-28).

   Alternatively, you could choose Insert→Form→Spry Validation Text Field. Either way, the Input Tag Accessibility window appears just as it does when you insert a regular text field. You fill it out the same way, too.

2. **In the ID field, type *phone*; in the Label field, type *Your phone number*. Make sure you have the "Attach label tag using 'for' attribute" and the "Before form item" radio buttons selected (they should be). Then click OK to insert the field.**

   A new text field and label appear. A blue tab also appears, identifying the field as a Spry widget. If you look at the Property inspector, you see all the properties available for Spry Text Fields. You'll choose a few options in a minute, but first you'll format this field like the one you inserted earlier.

3. **Repeat step 11 on page 569 to set the character width of the new field to 35.**

   The label also needs some formatting.

4. **Repeat step 18 on page 570 to format the "Your phone number" label.**

   When you use the Property inspector's style menu to apply the *.question* class, you see a whole bunch of new classes listed. Those classes come from another external style sheet that Dreamweaver quietly attached to this page after you inserted the Spry form field.

5. **Choose File→Save.**

   Dreamweaver opens the Copy Dependent Files dialog box, letting you know the page now requires both a style sheet file and a JavaScript file for the new Spry Validation field.

6. **Click OK to close the Copy Dependent Files window.**

   If you look at the Files panel (Window→Files), you see a folder named SpryAssets where Dreamweaver just saved the two new files (see the Note below on for an explanation). You see other files in there as well, for two other Spry widgets on this page—a Spry Menu Bar and a Spry Collapsible panel.

7. **Move your mouse over the phone form field in Design view; click the blue Spry tab when it appears.**

   The Property inspector displays the Spry field's properties (see Figure 13-32). Now you'll assign a validation type to this widget.

Note: Dreamweaver automatically creates an ID for the Spry widget. The widget you just inserted has the ID *sprytextfield1*. You can change it to something more descriptive, like *spryPhone*, but you don't have to because the name doesn't appear on your web page. The Spry programming uses it to identify this particular validation widget.

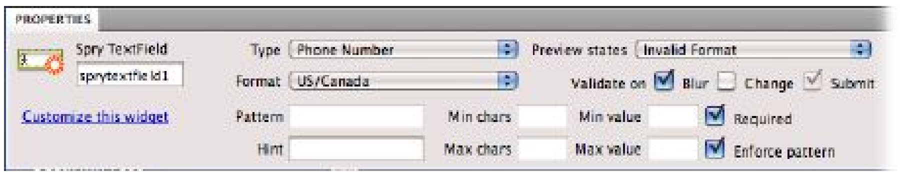

PROPERTIES
Spry TextField: sprytextfield1
Type: Phone Number
Format: US/Canada
Preview states: Invalid Format
Validate on: [✓ Blur] Change [✓ Submit]
Pattern:
Hint:
Min chars:
Max chars:
Min value:
Max value:
[✓ Required]
[✓ Enforce pattern]

Figure 13-32: *Select a Spry Validation widget in the document window, and then use the Property inspector to assign validation options.*

8. **From the Property inspector's Type menu, select Phone Number, and then turn on the Blur and Enforce pattern checkboxes.**

   The Type menu defines the type of information the widget allows in the field. In this case, only a validly formatted US or Canadian phone number works. The Blur box determines when the widget validates the field's contents. For example, if a visitor types "not telling" into the field and presses the Tab key to jump to the next field, he receives an error message. The Enforce pattern box will only let visitors type their phone number; it also conveniently fills out the required bits of punctuation—the (,), -, and spaces required to display a phone number in the format of (503) 555-2122.

If you look at the document window now, you see that the phone field has a red background and a red "Invalid format" message to the right of it. Every Spry Validation field has several "preview states." You can preview the field at the various points a visitor interacts with it—when she's typed invalid information, for example, or simply left a required field blank. You'll tweak this error message now.

9. **In the document window, replace the text "Invalid format" with *Format the # like this: (555) 555-1212*.**

   The Spry Validation widget requires that guests enter a phone number in a specific format—(555) 555-1212—so it's helpful to tell visitors what that format is in the error message. The Enforce pattern option (page 553), however, does most of the work by preventing anyone from typing in anything but numbers. Guests will only see the format error message if they leave out one or more digits in the phone number. An error message also appears if a guest leaves the field blank. You can adjust this message as well, but first you have to switch to a different "preview state."

Note: When you replace the error message in step 9, don't delete the "Invalid format" message before you type in the new one. If you do, you might accidentally delete the <span> responsible for making the error message work. Just select the text "Invalid format" and then type in the new message. As an added precaution, select all the text except for the period after the words "Invalid format." It's not common, but occasionally if you select all of the text, Dreamweaver actually deletes the <span> tag as well.

10. **Click the blue Spry tab again; from the Property inspector's "Preview states" menu, select Required.**

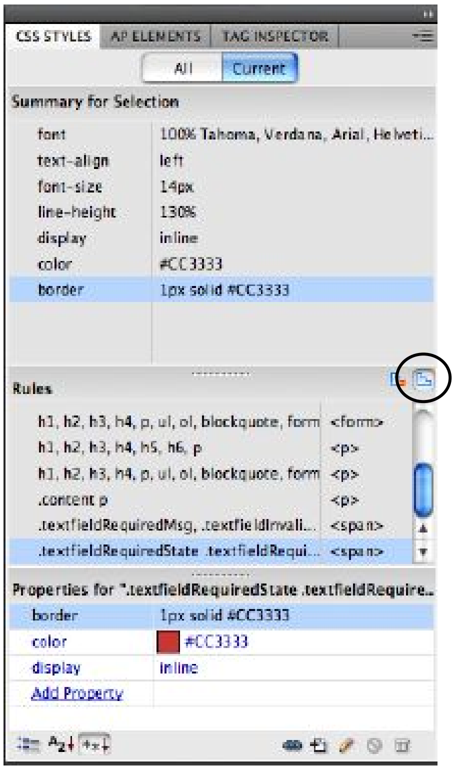
    A new error message appears: "A value is required." This message appears if a visitor leaves the phone-number field blank and tries to submit the form.

11. **Repeat step 9, replacing the "A value is required" with "We need your number to contact you."**

    The red outline surrounding the error message doesn't fit the look of Chia-Vet.com. Fortunately, you can easily update it.

12. **Click anywhere inside the error message you added in the last step; make sure you have the CSS Styles panel open (Window→CSS Styles). At the top of the panel, select the Current button, and then make sure you have the Cascade button (circled in Figure 13-33) selected.**

    A long-winded group selector (page 318), made up of a bunch of descendent selectors (page 315), controls the error message's style. The one you want is the last one listed in the Styles panel's Rules pane; it begins with "*.textfieldRequiredState .textfieldRequiredMsg*" (the middle pane in Figure 13-33).



Summary for Selection

font	100% Tahoma, Verdana, Arial, Helveti...
text-align	left
font-size	14px
line-height	130%
display	inline
color	#CC3333
border	1px solid #CC3333

Rules

h1, h2, h3, h4, p, ul, ol, blockquote, form	<form>
h1, h2, h3, h4, h5, h6, p	<p>
h1, h2, h3, h4, p, ul, ol, blockquote, form	<p>
.content p	<p>
.textfieldRequiredMsg, .textfieldInvali...	<span>
.textfieldRequiredState .textfieldRequi...	<span>

Properties for ".textfieldRequiredState .textfieldRequire...

border	1px solid #CC3333
color	🟥 #CC3333
display	inline
Add Property	

Figure 13-33:
To format Spry widgets most easily, use the CSS Styles panel's Current view. If you have the Cascade button (circled) selected, then you merely need to click the element whose style you want to change. In this case, selecting the error message for a Spry Validation text field highlights the appropriate style's name. Double-click the name to open the CSS Rule Definition window, and style away!

13. **In the Rules pane, double-click the style beginning with *.textfieldRequired-State .textfield-RequiredMsg* to open the CSS Rule Definition window.**

    You'll make the error message bold and remove the border.

14. **In the Rule Definition window, from the "font-weight" menu, choose Bold. Click the Border category, leave the "Same for all" boxes checked, and then delete the contents of the top row of boxes.**

    Deleting the contents of the type, width, and color boxes removes the border entirely.

15. **Click OK to complete the style.**

    Notice that the error message next to the Spry form field is bold, but no longer has a border. Time to insert another form field.

16. **Repeat steps 1–4 to add another Spry text validation field. Use the ID *date*, and, for the label, type *Please specify the date you'd like*.**

    Make sure you click to the right of the phone number widget's error message before hitting Return to insert the new field.

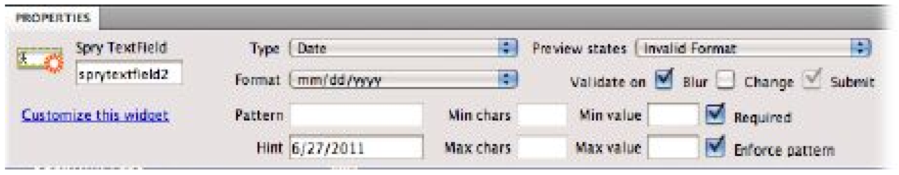
    This field collects the day, month, and year of the appointment. Since you want to make sure you receive properly formatted dates, add that validation option next.

17. **Select the Spry widget by moving your mouse over the new field, and then click the blue "Spry TextField" tab. In the Property inspector, choose Date from the Type menu, and then choose "mm/dd/yyyy" from the Format menu.**

    You just specified that visitors need to enter a date in a format like "9/22/2009." If they don't, they get an error message. Of course, people filling out this form might not know that, so you should give them a hint. And while you're at it, make sure they can't type in any incorrect characters, and that the validation check occurs the moment a visitor exits the form field.

18. **In the Hint box of the Property inspector, type 6/27/2011 and then select the Enforce pattern and Blur boxes.**

    The Property inspector should now look like Figure 13-34. Time to change the error message.



PROPERTIES

Spry TextField: sprytextfield2

Customize this widget

Type: Date | Preview states: Invalid Format

Format: mm/dd/yyyy | Validate on: [✓ Blur] Change [✓ Submit]

Pattern: | Min chars: | Min value: | [✓ Required]

Hint: 6/27/2011 | Max chars: | Max value: | [✓ Enforce pattern]

Figure 13-34: *There are many ways to type in a date—12/31/2009 or 12.31.2009, for example—so if you use the Spry text validation widget to collect a date, always provide a hint or other instruction that shows visitors the correct format.*

19. **From the "Preview states" menu in the Property inspector, select "Invalid Format." On the web page, change the text "Invalid format" to "Please enter a date."**

    Notice that the error message shares the same style as the one you added for the phone number; that's because you modified the look of error messages for *all* Spry text fields in step 14. Now, you just need to make the label and field match the others in this form.

    Dreamweaver supplies Spry widgets for other types of form fields as well. You'll add a Spry Form Menu next.

Adding a Spry Form Menu

Text boxes aren't the only form fields you can validate. You can make sure someone's made a selection from a pull-down menu by adding a Spry menu to the form.

1. **Click to the right of the error message you just added (after "Please enter a date"), and then press Enter or Return to insert a new, empty paragraph.**

    Make sure you click to the right of the Spry widget's blue outline before you press Return. If you hit Return immediately after you type in the error message (step 19 in the previous section of this tutorial), you're still inside the Spry Text Field widget, and the next steps won't work.

2. **Choose Insert→Form→Spry Validation Select, or, on the Insert panel, click the Spry Validation Select button (see Figure 13-28).**

   As with any form field, the Input Tag Accessibility window appears.

3. **In the ID field, type *time*; in the label box, type *Time of Appointment*, and then click OK to insert the menu.**

   A form menu appears on the page.

4. **Choose File→Save.**

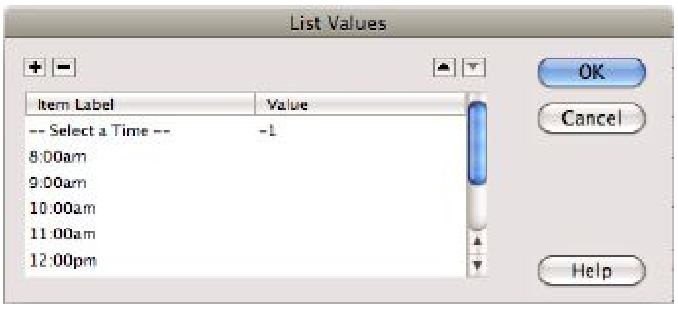
   Another window appears, letting you know that you need an additional JavaScript file and CSS file to make this Spry Validation widget work. Click OK to dismiss that window. Time to style the label.

5. **Click inside the "Time of appointment" text; click the <label> tag in the Tag selector and choose *.question* from the Property inspector's class menu.**

   In other words, repeat step 18 on page 570 to format the label. At this point, the pull-down form menu is empty, so your next step is to add a few options.

6. **In the document window, click the newly inserted menu to select it. In the Property inspector, click the List Values button.**

   The List Values window opens (see Figure 13-35), where you can add options for the menu.



Item Label	Value
-- Select a Time --	-1
8:00am	
9:00am	
10:00am	
11:00am	
12:00pm	

Figure 13-35: *The Item Label column displays what appears in the menu on the web page, while the Value column contains the value the form actually submits (-1 in the first row, for example). If you don't specify a value (as with all of the time ranges here), then the form submits the label's text as the value.*

7. **Type -- *Select a Time* --, press the Tab key, and then type *-1*.**

   The text ("--Select a Time--") appears at the top of the menu. It's an instruction telling your visitor what to do. Of course, you want your guest to choose an option *other* than this text. If he does select it, the -1 serves as a kind of secret message that flags the Spry Validation program that this option isn't valid. Before you get to that, though, add the valid selections for this form.

8. **Press the Tab key, and then type *8:00am*; press the Tab key twice to create another list option, and then type *9:00am*. Continue adding options until you add *5:00pm* (or until you've got the hang of adding menu items).**

   The List Values window should look like Figure 13-35.

9. **Click OK to insert the menu. Move your mouse over the menu, and then click the blue Spry tab to select it.**

   You need to specify invalid menu selections.

10. **In the Property inspector, turn off the "Blank value" box, turn on the "Invalid value" box, and then make sure *-1* appears in the field. Then turn on the Change checkbox.**

    The Property inspector should look like Figure 13-36. In the "Preview states" menu, make sure you have "Invalid" selected. In the document window, to the right of the form menu, you should see this red error message: "Please select a valid item." You'll change the message and its style next.

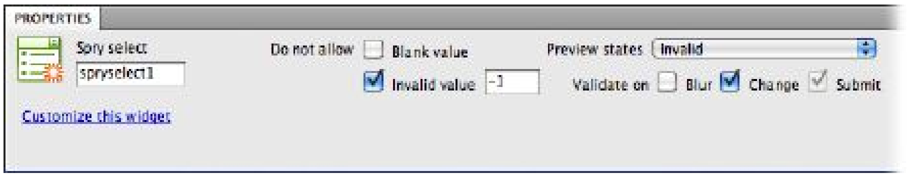

PROPERTIES
Spry select: spryselect1
Customize this widget
Do not allow: ☐ Blank value ✓ Invalid value -1
Preview states: Invalid
Validate on: ☐ Blur ✓ Change ✓ Submit

Figure 13-36: You can assign an arbitrary value (like -1) to any menu item that you want to prevent your visitors from submitting.

11. **Replace the text *Please select a valid item* with *Please select a time*. Repeat steps 12–15 on page 573 to make the error message bold and remove the borderline.**

    Each type of Spry Validation widget (text field, menu, checkbox, text area) has its own CSS styles to format error messages. So, unfortunately, if you use more than one type of validation in a form, you have to repeat the same formatting steps to get error messages that match. In the case of a pull-down menu error message, the style name begins with *.selectRequiredState .selectRequiredMsg*, and it's in an external style sheet named *SpryValidationSelect.css*.

Adding a Spry Radio Group

Just a couple more form fields and you'll be done. Now it's time to ask a multiple-choice question that requires a single response. This task is perfect for a group of radio buttons, but since you require an answer, you'll use Spry Validation as well.

1. **Click to the right of the error message you just formatted (after "Select a time"), and then press Enter (Return) to insert a new empty paragraph. Type *Please select a reason for this appointment*.**

   If you no longer see the error message you created in step 11 in the previous section, you probably just clicked somewhere else on the page and deselected the Spry widget. Time to format the question.

2. **From the Property inspector's Class menu, select *.question*.**

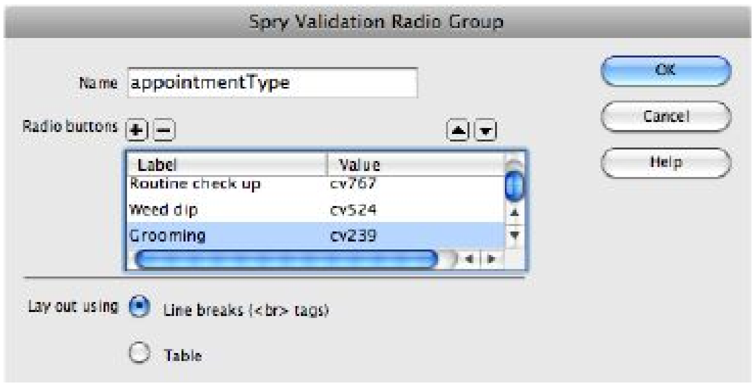
   If you don't see the Class menu in the Property inspector, make sure you have the HTML button selected. The Class menu formats the paragraph so that it looks like the labels for other questions on the form. Next you'll add a few checkboxes.

3. **Press the Enter (Return) key to create a blank paragraph.**

   The new paragraph also has the *.question* class applied to it. You need to remove that style.

4. **From the Property inspector's Class menu, choose None (it's at the top of the list). Choose Insert→Form→Spry Validation Radio Group.**

   Under the Insert panel's Forms category, you can also click the Spry Radio Group button (see Figure 13-28). Either way, the Spry Validation Radio Group window appears (Figure 13-37).



Name: appointmentType

Radio buttons

Label	Value
Routine check up	cv767
Weed dip	cv524
Grooming	cv239

Lay out using: [✓ Line breaks (
 tags)] Table

Figure 13-37: *When you insert a Spry Validation Radio Group, you can put the group of buttons and labels in a table or on single lines separated by line breaks (<br> tags). Choose the line breaks option–it gives you the most flexibility for controlling the display of your radio buttons.*

5. **In the Name field, type appointmentType.**

   This names the group of radio buttons—each button shares the same name, but obviously the label and value of each button should be unique. You'll add those next.

6. **In the Label column, click the first instance of the word "Radio" and type *Routine check-up*. Press the Tab key to jump to the Value column, and then type *cv767*.**

   The label is the text that appears next to the button, while the value is the data the form transmits to the form-processing program—in this case, "cv767" is Chia Vet's internal code for "Routine check up" (yeah, sure it is).

7. **Press the Tab key again to jump to the label column for the second radio button, type *Weed dip*, press Tab, and then type *cv524*.**

   You've now changed the labels and values for the two generic buttons supplied in this window. To add another, you need to use the + button.

8. **Press the + button to add another pair of label and value options. Type *Grooming* for the label, and *cv239* for the value.**

   The window should now look like Figure 13-37.

9. **Make sure you select the "Line breaks" button, and then click OK to insert the group of three radio buttons.**

   Three rows of radio buttons appear on the page, along with the now-familiar blue Spry tab. In the Property inspector, you don't have to make any validation choices because you always want the default option, Required—so, someone requesting an appointment can't submit this form until she's clicked one of the three radio buttons.

   You should, however, make sure the style of the error message matches that of the other messages on the page.

10. **From the "Preview states" menu in the Property inspector, choose Required. Repeat steps 12-15 on page 573 to format the "Please make a selection" error message.**

11. **Choose File→Save.**

    Another window appears, letting you know that you need an additional JavaScript file and a CSS file to make this new Spry Validation widget work (click OK to close that window).

Completing and Testing the Form

At this point, nobody can submit the form after they fill it out—you need to add a Submit button.

1. **Click to the right of the "Please make a selection" message for the radio group. Hit Return to create a new paragraph. From the Property inspector's Class menu, choose *.question*.**

   A green line and a little extra space appear above the paragraph. This look matches that of the other parts of the form.

2. **Choose Insert→Form→Button or, on the Insert panel, click the Button icon (see Figure 13-28).**

   Your old friend the Input Tag Accessibility window appears. This time, however, you don't need an ID or label. Buttons (like Submit) already have a message printed on them, so you don't need to add a label.

3. **Click the Cancel button.**

   In the Input Tag Accessibility window, clicking Cancel doesn't actually cancel the process of inserting a form field—it just skips the steps for providing an ID and label for the field. A Submit button appears on the page. You can change the generic "Submit" message to something more reflective of the form's purpose.

4. **Click the button on the page; in the Property inspector's Value field, type *Make appointment*.**

   The form is done. Now take it for a test drive.

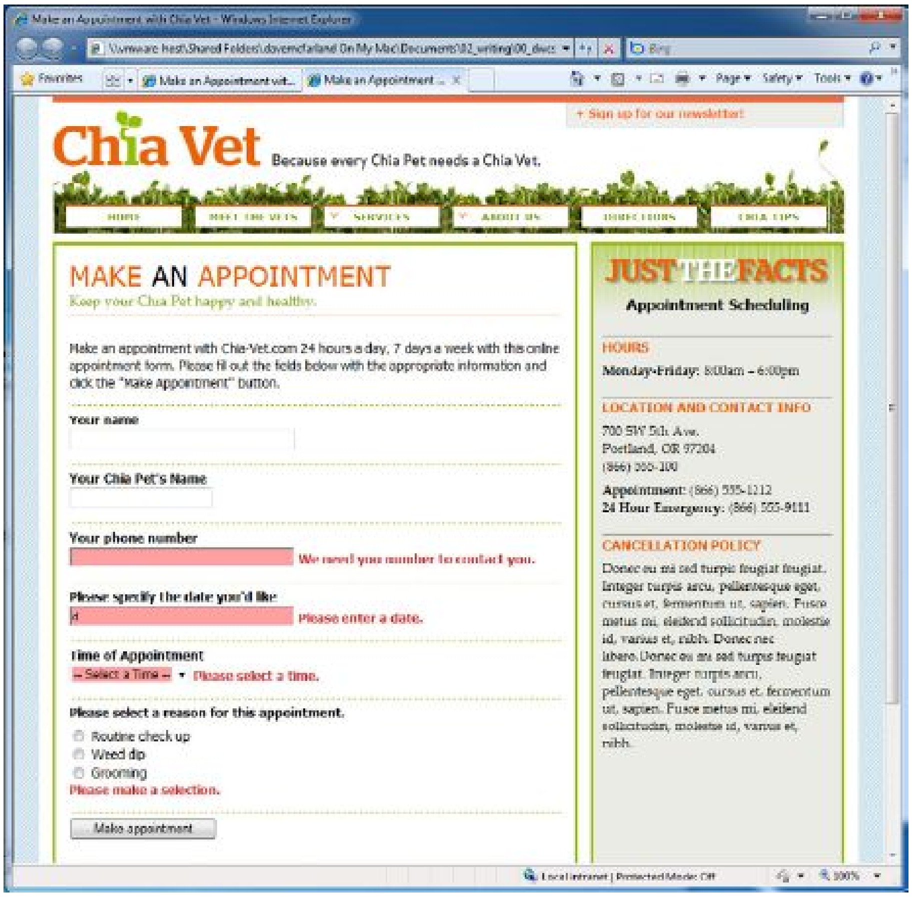
5. **Choose File→Save All, and then press the F12 key (Option-F12).**

   A web browser opens with the new form.

6. **Click the "Make appointment" button.**

   The form doesn't submit (see Figure 13-38). Instead, several error messages appear. Fill out the form correctly, and try to submit it again.

Note: If, after you submit the form, you notice that some of the information you entered doesn't show up on the form-processing page ("Appointment Scheduled"), you may not have typed the name of the field exactly as specified in the tutorial. Form-processing programs are very particular–if you don't provide the exact name it's expecting, it won't correctly capture the form data. You can change the names of form elements by selecting them and using the Property inspector.


Make an Appointment with Chia Vet - Windows Internet Explorer
Favorites
Make an Appointment wit...
Make an Appointment ...
Page
Safety
Tools
+ Sign up for our newsletter!
Chia Vet
Because every Chia Pet needs a Chia Vet.
HOME
MEET THE VETS
SERVICES
ABOUT US
DIRECTIONS
CHIA TIPS
Make appointment
Local intranet | Protected Mode: Off
100%

MAKE AN APPOINTMENT

Keep your Chia Pet happy and healthy.

Make an appointment with Chia-Vet.com 24 hours a day, 7 days a week with this online appointment form. Please fill out the fields below with the appropriate information and click the "Make Appointment" button.

Your name

Your Chia Pet's Name

Your phone number — We need you number to contact you.

Please specify the date you'd like — Please enter a date.

Time of Appointment: -- Select a Time -- — Please select a time.

Please select a reason for this appointment.

- Routine check up
- Weed dip
- Grooming

Please make a selection.

JUST THE FACTS

Appointment Scheduling

HOURS

Monday-Friday: 8:00am – 6:00pm

LOCATION AND CONTACT INFO

700 SW 5th Ave.
Portland, OR 97204
(866) 555-100

Appointment: (866) 555-1212
24 Hour Emergency: (866) 555-9111

CANCELLATION POLICY

Donec eu mi sed turpis feugiat feugiat. Integer turpis arcu, pellentesque eget, cursus et, fermentum ut, sapien. Fusce metus mi, eleifend sollicitudin, molestie id, varius et, nibh. Donec nec libero.Donec eu mi sed turpis feugiat feugiat. Integer turpis arcu, pellentesque eget, cursus et, fermentum ut, sapien. Fusce metus mi, eleifend sollicitudin, molestie id, varius et, nibh.

Figure 13-38: *Dreamweaver's Spry Validation Form widgets can help ensure that your forms collect the information you want. Professional-looking error messages, placed next to the offending responses, give visitors clear feedback.*