macromedia®
DREAMWEAVER®
UltraDev™

Using Dreamweaver UltraDev

macromedia™

Trademarks

Afterburner, AppletAce, Attain, Attain Enterprise Learning System, Attain Essentials, Attain Objects for Dreamweaver, Authorware, Authorware Attain, Authorware Interactive Studio, Authorware Star, Authorware Synergy, Backstage, Backstage Designer, Backstage Desktop Studio, Backstage Enterprise Studio, Backstage Internet Studio, Design in Motion, Director, Director Multimedia Studio, Doc Around the Clock, Dreamweaver, Dreamweaver Attain, Drumbeat, Drumbeat 2000, Extreme 3D, Fireworks, Flash, Fontographer, FreeHand, FreeHand Graphics Studio, Generator, Generator Developer's Studio, Generator Dynamic Graphics Server, Knowledge Objects, Knowledge Stream, Knowledge Track, Lingo, Live Effects, Macromedia, Macromedia M Logo & Design, Macromedia Flash, Macromedia Xres, Macromind, Macromind Action, MAGIC, Mediamaker, Object Authoring, Power Applets, Priority Access, Roundtrip HTML, Scriptlets, SoundEdit, ShockRave, Shockmachine, Shockwave, Shockwave Remote, Shockwave Internet Studio, Showcase, Tools to Power Your Ideas, Universal Media, Virtuoso, Web Design 101, Whirlwind and Xtra are trademarks of Macromedia, Inc. and may be registered in the United States or in other jurisdictions including internationally. Other product names, logos, designs, titles, words or phrases mentioned within this publication may be trademarks, servicemarks, or tradenames of Macromedia, Inc. or other entities and may be registered in certain jurisdictions including internationally.

This guide contains links to third-party Web sites that are not under the control of Macromedia, and Macromedia is not responsible for the content on any linked site. If you access a third-party Web site mentioned in this guide, then you do so at your own risk. Macromedia provides these links only as a convenience, and the inclusion of the link does not imply that Macromedia endorses or accepts any responsibility for the content on those third-party sites.

Apple Disclaimer

APPLE COMPUTER, INC. MAKES NO WARRANTIES, EITHER EXPRESS OR IMPLIED, REGARDING THE ENCLOSED COMPUTER SOFTWARE PACKAGE, ITS MERCHANTABILITY OR ITS FITNESS FOR ANY PARTICULAR PURPOSE. THE EXCLUSION OF IMPLIED WARRANTIES IS NOT PERMITTED BY SOME STATES. THE ABOVE EXCLUSION MAY NOT APPLY TO YOU. THIS WARRANTY PROVIDES YOU WITH SPECIFIC LEGAL RIGHTS. THERE MAY BE OTHER RIGHTS THAT YOU MAY HAVE WHICH VARY FROM STATE TO STATE.

Copyright © 2000 Macromedia, Inc. All rights reserved. This manual may not be copied, photocopied, reproduced, translated, or converted to any electronic or machine-readable form in whole or in part without prior written approval of Macromedia, Inc.
Part Number ZUD10M100

Acknowledgments

Project Management: Sheila McGinn

Writing: Charles Nadeau and Gayle Kidder

Editing: Anne Szabla

Multimedia Design and Production: John "Zippy" Lehnus and Noah Zilberberg

Print and Help Design and Production: Chris Basmajian

Help Engineering: Jed Hartman

Web Editing: Jane Dekoven and Jeff Harmon

Special thanks to Rob Christensen, Rebecca Godbois, Nick Halbakken, and Ken Sundermeyer

First Edition: May 2000

Macromedia, Inc.
600 Townsend St.
San Francisco, CA 94103

CONTENTS

INTRODUCTION
Getting Started

Macromedia Dreamweaver UltraDev is a professional visual editor for building Web applications. A Web application is a collection of pages that interact with each other and with various resources on a Web server, including databases.

Dreamweaver UltraDev is also a professional editor for creating and managing Web sites and pages. Because it incorporates all of Dreamweaver's advanced page design and site management tools, UltraDev makes it easy to create, manage, and edit cross-platform, cross-browser Web pages.

Dreamweaver UltraDev is fully customizable. You can create your own objects and commands, modify menus and keyboard shortcuts, and even write scripts to extend UltraDev with new actions, behaviors, and property inspectors.

Requirements

This section outlines three different sets of hardware and software requirements: one for running Dreamweaver UltraDev, one for running dynamic pages, and one for storing data. To create a functioning Web application, you must at minimum satisfy the first two sets of requirements. To use a database with your application, you must also satisfy the requirements for storing data.

What you need to run Dreamweaver UltraDev

The following hardware and software is required to run Dreamweaver UltraDev.

For Microsoft Windows:

- An Intel Pentium processor or equivalent, 166 MHz or faster, running Windows 95, Windows 98, Windows NT 4.0 with Service Pack 5, or Windows 2000.
- 48 MB of random-access memory (RAM) plus 30 MB of available disk space.
- If Microsoft Office 2000 is not installed, Microsoft Data Access Components (MDAC) 2.1, which you can download from Microsoft's Web site at http://www.microsoft.com/data/download.htm. (MDAC 2.1 is installed on your system when you install Office 2000.)
- If using JDBC to connect to a database, Sun's Java Development Kit version 1.1.8, which you can download from Sun's Web site at http://java.sun.com/products/jdk/1.1/jre/download-jre-windows.html.
- Macromedia Shockwave 8 to view the Guide Tours and Show Me movies. The Shockwave installer file is located on the CD or you can download it from Macromedia's Web site at http://www.macromedia.com/shockwave/download/.

For the Macintosh:

- A Power Macintosh running Mac OS 8.5 or later.
- 48 MB of random-access memory (RAM) plus 30 MB of available disk space.
- Mac OS Runtime for Java (MRJ) version 2.2. The MRJ installer file is located on the CD or you can download it from Apple's Web site at http://www.apple.com/java.
- Macromedia Shockwave 8 to view the Guide Tours and Show Me movies. The Shockwave installer file is located on the CD or you can download it from Macromedia's Web site at http://www.macromedia.com/shockwave/download/.

What you need to run dynamic pages

Creating a Web application in Dreamweaver UltraDev involves building dynamic pages. A dynamic page is processed at run time by an application server before it is sent to a browser by a Web server. To run dynamic pages, you need an application server that supports Microsoft's Active Server Pages (ASP), Sun's JavaServer Pages (JSP) version 1.0, or Allaire's ColdFusion—technologies that give a server the ability to generate HTML source code in a page before the page is served up to a browser.

This document assumes you have a Web server and an application server, or that your Web server has an integrated application server, as is the case with Microsoft's Personal Web Server (PWS) and Internet Information Server (IIS). If you don't have these, or are new to working with application servers, please read more on this topic at http://www.macromedia.com/software/ultradev/support/resources.

Here's a list of servers that run dynamic pages created in Dreamweaver UltraDev. Choose the one that best fits your needs.

ASP servers:

- Microsoft's Internet Information Server (IIS), which comes with Windows NT Server and Windows 2000 Server, and the hardware to run it.
- Microsoft's Personal Web Server (PWS), a scaled-down version of IIS that runs in Windows 95, 98, or Windows NT Workstation.

  If you have Windows 95 or NT Workstation, you can download PWS for free from the Microsoft Web site at http://www.microsoft.com/windows/ie/pws. If you have Windows 98, you'll find a copy in the Add-Ons/PWS folder on the Windows 98 CD.
- Any functioning Web server with the latest version of Chili!Soft ASP installed.

JSP servers:

- IBM's WebSphere with the operating system and hardware to run it.
- Allaire's JRun 2.3.3 with the operating system and hardware to run it.
- Any Web or application server that supports the JSP 1.0 specification.

ColdFusion server:

Any functioning Web server with Allaire's ColdFusion 4.0 or later installed.

What you need to store data

Most Web applications need a way to store and retrieve data on a server. If you plan to build applications that have this ability, you need some kind of database system to store the data. You may already have a database system that you plan to use. If not, your choice of database system will depend on your needs and budget.

If you plan to build small low-cost applications, you can use a desktop database program like Microsoft Access or Lotus Approach to store your data. You can even use a spreadsheet program like Lotus 1-2-3 or Microsoft Excel to create the data files.

If you plan to build robust, business-critical applications, use a larger, industrial-strength database system like Microsoft SQL Server or Oracle8i.

Installing Dreamweaver UltraDev (Windows version)

Follow these steps to install Dreamweaver UltraDev on a Windows computer.

To install Dreamweaver UltraDev on a Windows computer:

1. Insert the Dreamweaver UltraDev CD into the computer’s CD-ROM drive.
2. In Windows Explorer, locate the Dreamweaver UltraDev installer file on the CD and double-click it to begin the installation.
3. Follow the onscreen instructions.
4. If prompted, restart your computer.

Installing Dreamweaver UltraDev (Macintosh version)

To install Dreamweaver UltraDev on a Macintosh, first install Mac OS Runtime for Java, then install UltraDev. Finally, you need a JDBC driver to work with databases. If you don't have a JDBC driver, install the RmiJdbc driver included on the UltraDev CD. See Installing RmiJdbc.

Installing Mac OS Runtime for Java

UltraDev needs Apple's Mac OS Runtime for Java (MRJ) to work properly.

To install Mac OS Runtime for Java (MRJ):

1. Copy the MRJ installer file from the CD, or download it from Apple's Web site at http://www.apple.com/java.
2. Double-click the MRJ_Update file.
3. Double-click the MRJ Install icon, double-click the Installer icon, and follow the onscreen instructions.

Installing UltraDev

After installing MRJ, install Dreamweaver UltraDev.

To install Dreamweaver UltraDev:

1. Insert the Dreamweaver UltraDev CD into the computer's CD-ROM drive.
2. On the Macintosh, double-click the Dreamweaver UltraDev Installer icon.
3. Follow the onscreen instructions.
4. If prompted, restart your computer.

Installing RmiJdbc

If you're a Macintosh user, you must use a JDBC driver to connect to your database. If you don't currently have a JDBC driver to work with your database, you can install the client and server components of the RmiJdbc driver included on the UltraDev CD. The RmiJdbc driver allows you to connect to your database at run time and "view" the database at design time. (You can use other JDBC drivers to connect to your database.)

The RmiJdbc driver consists of two components: a client and a server component. This section describes how to install the client component on the Macintosh and the server component in Windows NT Server.

Installing the RmiJdbc driver on the Macintosh

You can install the RmiJdbc driver on a Macintosh to create a connection to a database hosted on a server running the RmiJdbc server component. Before installing the RmiJdbc driver on the Macintosh, make sure the Mac OS Runtime for Java is installed. See Installing Mac OS Runtime for Java.

If you use Windows NT Server to host your database, see Installing the RmiJdbc server component in Windows NT Server.

To install the RmiJdbc driver on the Macintosh:

1 Download the uncompressed version of the RmiJdbc.jar file from the UltraDev Support Center at http://www.macromedia.com/support/ultradev/.

  If you bought the boxed version of UltraDev, the file is located in the RmiJdbc folder on the CD.

2 Place the file in the following folder on your Macintosh:

```
System Folder:Extensions:MRJ Libraries:MRJClasses
```

Installing the RmiJdbc server component in Windows NT Server

You must install the RmiJdbc server component on a server to allow a client running the RmiJdbc driver to create a connection to a database hosted on the server. This section describes how to install the RmiJdbc server component in Windows NT Server.

Before installing the RmiJdbc server component in Windows NT Server, make sure Sun's Java Development Kit version 1.1.8 is installed on the server. You can download the Java Development Kit from Sun's Web site at http://java.sun.com/products/jdk/1.1/jre/download-jre-windows.html.

Installing the RmiJdbc server component consists in downloading and setting up the server component on NT Server, then starting RmiJdbc.

To download and set up the RmiJdbc server component in Windows NT Server:

1 Create a new folder to put the downloaded server component file. Example:

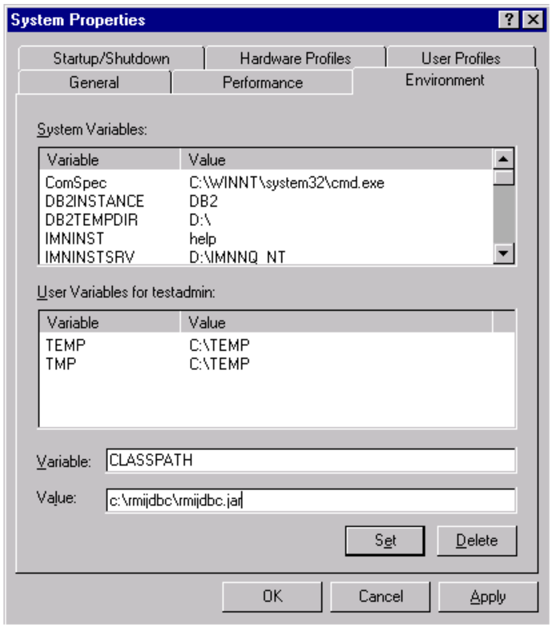
   c:\RmiJdbc

2 Download the uncompressed version of the RmiJdbc.jar file into the new folder from the UltraDev Support Center at http://www.macromedia.com/support/ultradev/.

   If you bought the boxed version of UltraDev, the file is located in the RmiJdbc folder on the CD.

3 On the NT desktop, right-click the My Computer icon, select Properties from the pop-up menu, and click the Environment tab.

   The Environment tab appears:



System Variables:

Variable	Value
ComSpec	C:\WINNT\system32\cmd.exe
DB2INSTANCE	DB2
DB2TEMPDIR	D:\
IMNINST	help
IMNINSTSRV	D:\IMNNQ_NT

User Variables for testadmin:

Variable	Value
TEMP	C:\TEMP
TMP	C:\TEMP

Variable: CLASSPATH

Value: c:\rmijdbc\rmijdbc.jar

4 In the System Variables list, select the CLASSPATH variable if one exists, or select any other variable, if the CLASSPATH variable doesn't exist.

   The selected variable's name and value appear in the Variable and Value text boxes at the bottom of the dialog box.

5 In the Variable text box, make sure CLASSPATH appears. If you selected another variable, delete its name and enter CLASSPATH.

   Note: The initial variable is unaffected and continues to be listed in System Variables.

6 In the Value text box, enter the path to the rmijdbc.jar file you downloaded in step 2. Example:

c:\RmiJdbc\rmijdbc.jar.

7 Click the Set button to add this variable to the System Variables list.

8 Next, tell the server where to find the Java Development Kit. In the System Variables list, select the Path variable if one exists, or select any other variable, if the Path variable doesn't exist.

The selected variable's name and value appear in the Variable and Value text boxes at the bottom of the dialog box.

9 In the Variable text box, make sure Path appears. If you selected another variable, delete its name and enter **Path**.

Note: The initial variable is unaffected and continues to be listed in System Variables.

10 In the Value text box, enter the path to the the Java Development Kit folder on your system (jdk1.1.8\bin). If your server already has a server variable, append jdk1.1.8\bin to it. If you installed the Java Development Kit in the root folder, append c:\jdk1.1.8\bin. Example:

D:\Oracle\Ora81\bin;d:\Oracle\Ora81\jdbc\lib;c:\jdk1.1.8\bin

11 Click the Set button to add this variable to the System Variables list.

12 Click OK to close the System Properties dialog box.

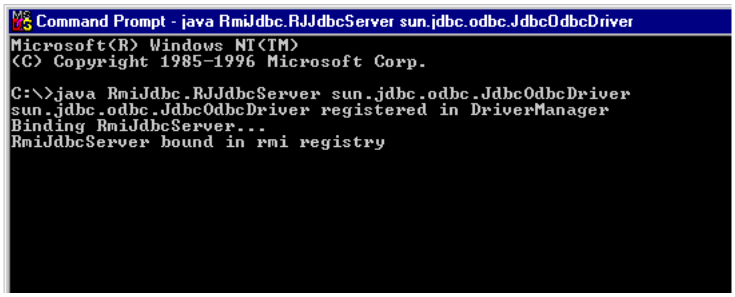
13 Restart the computer so the new settings can take effect.

To start RmiJdbc:

1. Open an MS-DOS command prompt by selecting Start Menu > Programs > Command Prompt.
2. At the C:\ prompt, type the following line:

   java RmiJdbc.RJJdbcServer sun.jdbc.odbc.JdbcOdbcDriver
3. Press Enter.

   Your window should look like this:


Command Prompt - java RmiJdbc.RJJdbcServer sun.jdbc.odbc.JdbcOdbcDriver

```
Microsoft(R) Windows NT(TM)
(C) Copyright 1985-1996 Microsoft Corp.

C:\>java RmiJdbc.RJJdbcServer sun.jdbc.odbc.JdbcOdbcDriver
sun.jdbc.odbc.JdbcOdbcDriver registered in DriverManager
Binding RmiJdbcServer...
RmiJdbcServer bound in rmi registry
```

4. Keep the Command Prompt window open (closing it shuts down the RmiJdbc server).

   You might want to create a batch file to open the window every time Windows NT is rebooted.

A connection can now be created between a database on the server and any Macintosh or Windows system running the RmiJdbc driver.

Learning Dreamweaver UltraDev

The Dreamweaver UltraDev package contains a variety of resources to help you learn the program quickly and become proficient in creating Web sites, Web pages, and Web applications. These resources include online help pages, Guided Tour movies, tutorials, and two printed user guides. In addition, the Dreamweaver UltraDev Support Center (Help > UltraDev Support Center) is updated regularly.

Dreamweaver UltraDev has all the functionality of Dreamweaver 3 for building Web pages and managing Web sites. To learn more, see *Using Dreamweaver* or Dreamweaver Help (Help > Using Dreamweaver). To learn how to build Web applications, consult this guide or use UltraDev Help (Help > Using UltraDev).

The following sections describe the instructional resources at your disposal.

Guided Tour and Show Me movies

The Show Me movies in the UltraDev help system provide guidance to creating basic Web application pages such as a search page, a results page, and a detail page. The Guided Tour includes all the Show Me movies in sequence. To view the Guided Tour, choose Guided Tour of UltraDev from the Help menu.

You can also view individual Show Me movies in their respective sections. In the help system's table of contents, the following icon indicates that a topic contains a Show Me movie:

Show Me movies require the Shockwave plugin, which is included on the Dreamweaver UltraDev CD. If you purchased your copy of Dreamweaver UltraDev electronically, you can download the latest Shockwave plugin from the Macromedia Web site at http://www.macromedia.com/shockwave/download/.

Help systems

The Dreamweaver UltraDev package comes with two help systems: Dreamweaver Help and UltraDev Help. These HTML-based help systems provide comprehensive information about all Dreamweaver and UltraDev features.

To view the help systems, use Netscape Navigator 4.0 and later or Microsoft Internet Explorer 4.0 and later. Because the help systems make extensive use of JavaScript, make sure JavaScript is enabled in your browser. If you plan to use the search feature, make sure Java is enabled as well.

Each help system includes the following components:

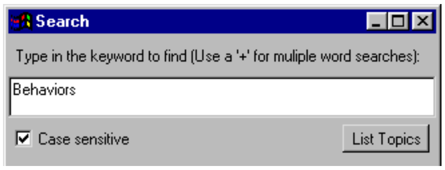
The table of contents enables you to see all information organized by subject. Click top-level entries to view subtopics.

The index, like a traditional printed index, helps you find important terms and go to related topics.

Search allows you to find any character string in all topic text. The search feature requires a 4.0 browser with Java enabled.

Note: After clicking Search, a Java security window may appear, asking for permission to read files on your hard disk. You must grant this permission for the search to work. The applet does not write anything to your hard disk, nor does it read any files outside the HTML help pages.

- To search for a phrase, simply type the phrase in the text field.

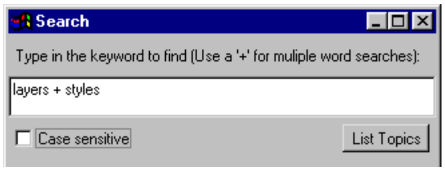


Type in the keyword to find (Use a '+' for muliple word searches):

Behaviors

☑ Case sensitive

- To search for files that contain two keywords (for example, *layers* and *styles*), separate the search terms with a plus (+) sign.

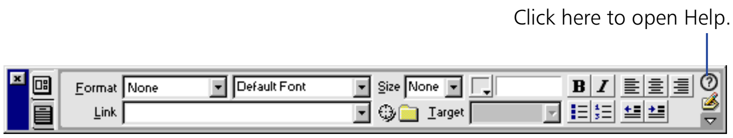


Type in the keyword to find (Use a '+' for muliple word searches):

layers + styles

☐ Case sensitive

Context-sensitive help provides a Help button in each dialog box, or a question mark icon in inspectors, windows, and palettes, that opens a relevant help topic.



The navigation bar provides buttons you can click to move through topics. The Previous and Next buttons move to the previous or next topic in a section (following the topic order listed in the table of contents).

Tutorials

The Dreamweaver UltraDev package comes with one tutorial to get you started with Dreamweaver, and a different tutorial to introduce you to Web application building.

The Dreamweaver tutorial is the best place to start if you don't have much experience with Dreamweaver. By working through the tutorial, you'll learn how to edit a sample Web site with some of Dreamweaver's most useful and powerful features. The tutorial is in Dreamweaver Help and in the *Using Dreamweaver* book.

If you're already familiar with Dreamweaver, start with the UltraDev tutorial. The UltraDev tutorial teaches you how to build a small Web application that draws content from a database. The UltraDev tutorial is in UltraDev Help and in the *Using Dreamweaver UltraDev* book.

User guides (printed books)

The *Using Dreamweaver* and *Using Dreamweaver UltraDev* books included in the boxed version of UltraDev provide a printed alternative to the two help systems. Certain reference topics about program options are not included in the printed books; see the help pages for information on those topics.

Extensibility documentation

The *Extending Dreamweaver* book and help pages provide information on the Dreamweaver Document Object Model and the APIs (application programming interfaces) that allow JavaScript and C developers to create objects, commands, property inspectors, behaviors, and translators.

Support centers

To help you get the most out of Dreamweaver UltraDev, you can consult two Web-based support centers:

- For information on the site-creation and page-design side of Dreamweaver UltraDev, visit the Dreamweaver Support Center at http://www.macromedia.com/support/dreamweaver/.
- For information on the application-building side of the product, visit the UltraDev Support Center at http://www.macromedia.com/support/ultradev/.

Both support centers are updated regularly with the latest information, plus advice from expert users, information on advanced topics, examples, and tips.

Note to Drumbeat users

Like Drumbeat, Dreamweaver UltraDev is both a Web page authoring tool and an application development tool. You can make a smooth transition to its page authoring dimension by using the many Dreamweaver instructional resources, including the Dreamweaver Guided Tour, tutorial, help system, and printed user guide. For more information, see Where to start in *Using Dreamweaver*.

You can get up to speed with developing Web applications in UltraDev by using the UltraDev instructional resources, including the UltraDev Guided Tour, tutorials, help system, and user guide. For more information, see Learning Dreamweaver UltraDev.

To find out what's different between Drumbeat and UltraDev, see Appendix A, "UltraDev for Drumbeat Users." The appendix also discusses converting existing Drumbeat applications to UltraDev applications.

Where to start

The Dreamweaver UltraDev package includes information for readers at a variety of levels. To get the most out of the documentation, start by reading the parts that are most relevant to your level of experience.

For Web designers new to Dreamweaver:

See Where to start in *Using Dreamweaver*.

For experienced Web designers new to building Web applications:

1. Take the UltraDev Guided Tour (Help > Guided Tour of UltraDev).
2. Work through the Dreamweaver UltraDev Tutorial to learn the basics of using UltraDev.
3. Read Dreamweaver UltraDev Basics to make sure you understand basic concepts and terms.
4. Read the following topics as needed: Laying the Groundwork for an Application, Page Blueprints, Creating a Recordset, Adding Dynamic Content, Activating Dynamic Pages, and Editing and Debugging Dynamic Pages.

For experienced Web application builders:

1 Take the UltraDev Guided Tour (Help > Guided Tour of UltraDev).

2 Work through the Dreamweaver UltraDev Tutorial to learn the basics of using UltraDev.

3 Read the following topics as needed: Laying the Groundwork for an Application, Page Blueprints, Creating a Recordset, Adding Dynamic Content, Activating Dynamic Pages, and Editing and Debugging Dynamic Pages.

Typographical conventions

The following typographical conventions are used in this guide:

- `Code font` indicates scripts, SQL statements, HTML tag and attribute names, and literal text used in examples.
- *`Italic code font`* indicates replaceable items (sometimes called *metasymbols*) in code.
- **Bold roman text** indicates text for you to enter verbatim.

Web application resources

The following are some useful resources available on the Web:

The HTML 4.01 specification (http://www.w3.org/TR/REC-html40/) is the official specification for HTML from the World Wide Web Consortium.

Microsoft's ASP Overview pages (http://msdn.microsoft.com/workshop/server/asp/ASPover.asp) provide information about Active Server Pages (ASP).

Sun's JSP page (http://java.sun.com/products/jsp/) provides information about JavaServer Pages (JSP).

Allaire's ColdFusion product page (http://www.coldfusion.com/Products/ColdFusion/) provides information about ColdFusion.

The XML.com site (http://www.xml.com) provides information about XML.

Microsoft's Personal Web Server page (http://www.microsoft.com/windows/ie/pws/) provides information about Microsoft's Personal Web Server.

Chili!Soft's product page (http://www.Chilisoft.com/products/) provides information about Chili!Soft ASP.

1

CHAPTER 1 Dreamweaver UltraDev Tutorial

This tutorial shows you how build a simple Web application using Dreamweaver UltraDev. You'll create a small extranet Web application for the fictitious Scaal Coffee company. The application will give Scaal employees the ability to view summaries of the company's press releases and read the full text of any press release they choose.

Along the way, you'll learn how to perform the following one-time-only tasks:

- Define a local site and specify a server technology for that site
- Define a remote site and configure the Live Data window
- Create a database connection

You'll also learn how to build the following pages:

- A master page to let employees view summaries of the company's press releases
- A detail page to display the full text of a press release of the user's choosing

To complete this tutorial, you need either a Web server that supports your chosen server technology (ASP, for example), or a Web server working in tandem with an application server that supports your chosen server technology (the ColdFusion application server, for example). For more information on setting up your system and servers, see the Macromedia Web site at http://www.macromedia.com/software/ultradev/support/resources. You may also need to specify information found in your Web server's documentation. Your system administrator may be able to provide this information.

Take a guided tour of Dreamweaver UltraDev

If you haven't already done so in Getting Started, watch the Guided Tour movie in UltraDev Help to familiarize yourself with UltraDev's features and workflow.

In Dreamweaver UltraDev, choose Help > Guided Tour of UltraDev, then select UltraDev Workflow from the list of Show Me movies.

Preview the completed Web application

You can view a completed version of the tutorial application on Macromedia's Web site at http://www.macromedia.com/software/ultradev/tutorial/.

Define a local site

A local site tells UltraDev where all the documents and files of a particular Web site are stored on your local disk.

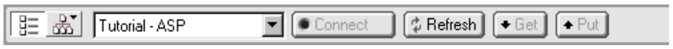
Defining a local site in UltraDev consists of two steps:

- Defining a local site folder
- Specifying a server technology

Define a local site folder

When it installed, UltraDev automatically created three local sites for you: one for the ASP tutorial, one for the JSP tutorial, and one for the ColdFusion tutorial. Choose the local site that's appropriate for your server technology (ASP, JSP, or ColdFusion). For more information on these technologies, see "What you need to run dynamic pages" on page 9.

To choose a local site, launch UltraDev, open the Site window (Site > Site Files), and select the site from the pop-up menu on the toolbar. For example, if you have an ASP server, select the Tutorial - ASP site.



For information on creating your own local sites, see Laying the Groundwork for an Application.

If for any reason you need to restart the tutorial from scratch, clean copies of the tutorial files are available in the Tutorial folder in the UltraDev application folder. Open the subfolder appropriate to your server technology (ASP, JSP, or ColdFusion) and copy the clean copies from the scaal_extranet_backup folder.

Note: The complete path to the Tutorial folder varies, depending on where you installed Dreamweaver UltraDev.

Specify a server technology

Server technologies such as ASP, JSP, or ColdFusion give a Web server the ability to modify Web pages at run time. This is where the term *dynamic pages* comes from; a dynamic page is essentially one that changes at run time.

Specifying a server technology tells UltraDev what server-side scripts to insert in your dynamic pages. A server-side script is a set of instructions the server executes at run time. In UltraDev, these scripts are called server behaviors.

Choose the server technology of the Web server you intend to use for the tutorial. For example, if you intend to use Personal Web Server or Windows NT Server, choose ASP. For more information, see “What you need to run dynamic pages” on page 9.

You specify a server technology for a site as a whole, not for individual pages. This ensures that all the pages in your application are compatible.

For this tutorial, you don't need to specify a server technology. During the installation, UltraDev automatically specified it for you.

For information on specifying the server technology for your own sites, see “Laying the Groundwork for an Application” on page 55.

Define a remote site

To tell UltraDev where all the documents and files of your Web site are stored on your Web server, you define a remote site. Defining a remote site also allows you to view your page's dynamic content in UltraDev's Live Data window, a fully functional, visual design and editing environment.

To define a remote site, you must complete the following tasks:

- Set up a published directory for the tutorial on your Web server
- In UltraDev, define the remote site folder for the Tutorial site by specifying the published directory you set up on the Web server, then uploading all the tutorial files to it
- Configure UltraDev so that it uses the Web server hosting the remote site to run pages opened in the Live Data window

Set up a published directory on your Web server

Make sure your Web server supports either ASP, JSP, or ColdFusion (see "Specify a server technology" on page 23), then set up a published directory for the tutorial on the server. For setup instructions, see the server documentation, or consult your system administrator.

After setting up the published directory, write down the URL of the folder at the root of the site. Example:

http://my_server_name/my_root_folder/

Define a remote site folder

In UltraDev, you define the tutorial's remote site folder by specifying the published directory you just set up on the Web server, then upload all the tutorial files to it.

1 Choose Site > Define Sites.

   A dialog box appears listing currently defined sites.

2 Select your Tutorial site from the list and click Edit.

3 In the Category list on the left, click Web Server Info.

4 Choose one of the following Server Access options: Local/Network or FTP.

5 If you chose Local/Network, click the folder icon and select the folder you set up as a published directory in your Web server.

6 If you chose FTP, enter the host name of the FTP host, and enter the name of the host directory at the remote site. The host directory is where documents visible to the public are stored. Next, enter the login name and password used to connect to the FTP server, then select the appropriate firewall options.

7 Click OK and click Done.

8 Open the Site window (Site > Site Files) to verify that you've specified the correct directory on the remote site. If you're using FTP, click Connect to view the remote site.

9 Select all the files in the Local Folder pane, including the Images folder, and click Put to upload them to the remote site.

Configure the Live Data window

UltraDev's powerful Live Data window allows you to view the dynamic content on your page while you work on the page. The Live Data window works with the Web server that hosts your remote site. UltraDev borrows the server to run the page and display the results in the Live Data window.

You must tell UltraDev which Web server to use to run dynamic pages in the Live Data window.

1 Choose Site > Define Sites.

  A dialog box appears listing currently defined sites.

2 Select your site and click Edit.

3 In the Category list on the left, click App Server Info.

4 In the Live Data Server text box, choose Remote Web Server to use the server that runs your remote site. The remote site is defined in the Web Server Info category.

  Note: Choose Remote Web Server even if your "remote site" is located on the system running UltraDev—for example, if you're using Personal Web Server to host your Web site.

5 In the Live Data URL Prefix text box, enter the URL prefix of your site.

  The URL prefix is the URL of the folder at the root of your site, not the full URL of any subfolder containing your documents. You wrote down the URL prefix earlier in the tutorial (see "Set up a published directory on your Web server" on page 24).

  If the site is published on your local system, you can simply enter the site's alias at the end of the http://localhost/ string, as in this example:

  http://localhost/my_tutorial_alias/

6 Click OK, then click Done.

Create a database connection

A database connection is a set of parameters you define to establish a link to a database. Without it, your application won't know where to find the database or how to connect to it. You'll create a run-time connection to the tutorial's database file, scaalcoffee.mdb, which should now reside on a remote site. (You uploaded the file to your remote site folder earlier; see "Define a remote site folder" on page 25.)

The type of run-time connection you create will depend on the server technology supported by your Web server. If the server supports ASP, you'll create an ADO run-time connection. If the server supports JSP, you'll create a JDBC connection. If you have a ColdFusion server, you'll create a ColdFusion connection.

If you're a Macintosh user designing an ASP or ColdFusion site, you also need to create a design-time JDBC connection.

Requirements

The tutorial uses a database file created in Microsoft Access. You don't need Microsoft Access to run the tutorial, but you do need a driver capable of reading Access 97 files. The driver allows your Web application to read the contents of the database file.

For Windows users:

If your Web site is hosted on a Windows system, make sure the system has the Microsoft Access Driver, version 3.5 or later. If you've installed Microsoft Office 2000, or if your Web site is hosted on Windows NT Server or Windows 2000 Server, then you should already have the driver.

To find out if you have the Microsoft Access Driver, in Windows choose Start > Settings > Control Panel, then look for the ODBC Data Sources icon. (Depending on your system, the icon could also be called ODBC or 32bit ODBC.) If the icon is not there, then download and install the Microsoft Data Access Components (MDAC) 2.1, which will install the missing Access driver.

If the ODBC Data Sources icon is present, double-click it. The ODBC Data Sources Administrator dialog box appears. Click the Drivers tab for a list of drivers installed on your system. If the Microsoft Access Driver (*.mdb) doesn't appear in the list, then download and install the Microsoft Data Access Components (MDAC) 2.1, which will install the missing Access driver.

You can download the Microsoft Data Access Components (MDAC) 2.1 from Microsoft's Web site at http://www.microsoft.com/data/download.htm.

For Macintosh users:

Macintosh users must use a JDBC driver capable of reading Access 97 files. The JDBC driver called RmiJdbc that ships with UltraDev has this capability. For more information, see "Installing RmiJdbc" on page 12.

Create a run-time connection: ASP users

When creating a connection, you must choose from several types of connections: ADO, JDBC, and ColdFusion. If your tutorial site is hosted on an ASP-capable Web server, you must choose an ADO run-time connection. ADO stands for Microsoft ActiveX Data Objects. This requirement applies to both Windows and Macintosh users.

If the Web server and UltraDev both run on the same Windows system, you can use a data source name (DSN) to create the connection. A DSN is a kind of shortcut to a database.

A DSN to the tutorial database already exists if you installed UltraDev on the Windows system that runs your Web server—that is, if you're using Personal Web Server in Windows, or if you installed UltraDev on Windows NT Server or Windows 2000 Server. (The DSN was created during the UltraDev installation.)

If UltraDev is not running on the system that runs your Web server, you must use a connection string to create the connection.

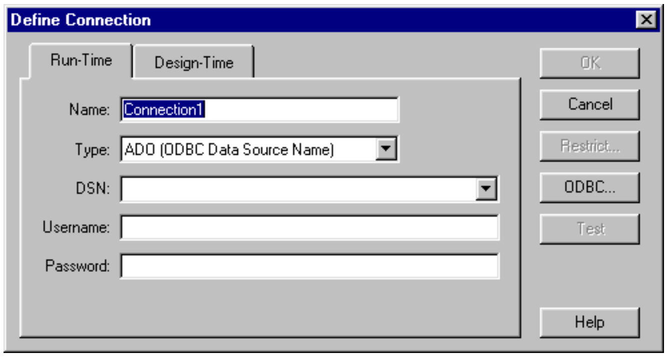
To create a run-time ADO connection if UltraDev runs on the Windows system running your Web server:

1 In UltraDev's Document window, choose Modify > Connections.

  The Connections dialog box appears.

2 Click New.

  The Define Connection dialog box appears.



Name: Connection1
Type: ADO (ODBC Data Source Name)
DSN:
Username:
Password:

3 On the Run-Time tab, click the Name text box and enter **connScaal** as your connection name.

  A common practice is to add the prefix *conn* to connection names to distinguish them from other object names in your code.

4 In the Type pop-up menu, make sure ADO (ODBC Data Source Name) is selected.

5 Select scaalcoffee from the DSN pop-up menu.

6 Click OK to close the Define Connection dialog box.

  Your new connection, connScaal, should appear in the Connections dialog box.

7 Click Done to close the Connections dialog box.

To create a run-time ADO connection if UltraDev does not run on the system running your Web server:

1 In UltraDev's Document window, choose Modify > Connections.

   The Connections dialog box appears.

2 Click New.

   The Define Connection dialog box appears.

3 On the Run-Time tab, click the Name text box and enter **connScaal** as your connection name.

   A common practice is to add the prefix *conn* to connection names to distinguish them from other object names in your code.

4 In the Type pop-up menu, make sure ADO (Connection String) is selected.

5 In the String text box, enter a connection string to the Scaal Coffee company database located on the Web server.

   If you don't know the connection string, see your server documentation or consult your system administrator. Here's an example of a connection string if the database file resides in the c:\UltraDevTutorial\ directory on a Windows NT Server system called ANAKIN-3:

```
Provider=MSDASQL;Driver={Microsoft Access Driver (*.mdb)};
DBQ=\\anakin-3\c-drive\UltraDevTutorial\scaalcoffee.mdb;
```

6 Click OK to close the Define Connection dialog box.

   Your new connection, connScaal, should appear in the Connections dialog box.

7 Click Done to close the Connections dialog box.

Create a run-time connection: JSP users

If your tutorial site is hosted on a JSP-capable server, you must create a JDBC run-time connection. This requirement applies to both Windows and Macintosh users.

1 In UltraDev's Document window, choose Modify > Connections.

The Connections dialog box appears.

2 Click New.

The Define Connection dialog box appears.

3 On the Run-Time tab, click the Name text box and enter **connScaal** as your connection name.

A common practice is to add the prefix *conn* to connection names to distinguish them from other object names in your code.

4 In the Type pop-up menu, select JDBC.

5 Enter the JDBC connection information.

Here is the information required if the tutorial database resides on a Windows NT Server:

Driver:	sun.jdbc.odbc.JdbcOdbcDriver
Username:	*my_username*
Password:	*my_password*
URL:	jdbc:odbc:*my_DSN*

To create a DSN (*my_DSN*), see "Setting up a DSN (ODBC)" on page 70.

For example, if the scaalcoffee.mdb database file resides on a Windows NT Server and you set up a DSN called *scaalcoffee* on the server, enter the following information in Windows or on the Macintosh:

Driver:	sun.jdbc.odbc.JdbcOdbcDriver
Username:	*my_username*
Password:	*my_password*
URL:	jdbc:odbc:scaalcoffee

6 Click OK to close the Define Connection dialog box.

Your new connection, connScaal, should appear in the Connections dialog box.

7 Click Done to close the Connections dialog box.

Create a run-time connection: ColdFusion users

If your tutorial site is hosted on a ColdFusion server, you must create a ColdFusion run-time connection. This requirement applies to both Windows and Macintosh users.

1 If ColdFusion is installed on a remote server, define a data source name (DSN) for the database in ColdFusion.

  For instructions, see the ColdFusion documentation or consult your system administrator.

2 In UltraDev's Document window, choose Modify > Connections.

  The Connections dialog box appears.

3 Click New.

  The Define Connection dialog box appears.

4 On the Run-Time tab, click the Name text box and enter **connScaal** as your connection name.

  A common practice is to add the prefix *conn* to connection names to distinguish them from other object names in your code.

5 In the Type pop-up menu, select ColdFusion Data Source Name.

6 If ColdFusion is installed on the same system that runs UltraDev, enter **scaalcoffee** in the DSN text box.

7 If ColdFusion is installed on a remote server, enter the data source name you defined in the ColdFusion Administrator in step 1.

8 If required, complete the Username and Password text boxes.

9 Click the Design-Time tab, then enter the following design-time connection information:

Type:	ADO (ODBC Data Source Name)
DSN:	scaalcoffee

10 Click OK to close the Define Connection dialog box.

  Your new connection, connScaal, should appear in the Connections dialog box.

11 Click Done to close the Connections dialog box.

Create a design-time connection: Macintosh users

If you're a Macintosh user designing an ASP or ColdFusion site, you must create a design-time JDBC connection to view information about the database while you design your pages.

Note: If using the RmiJdbc driver as the JDBC driver, make sure you properly install both the client and the server components of the driver. For instructions, see "Installing RmiJdbc" on page 12.

1 In UltraDev's Document window, choose Modify > Connections.

   The Connections dialog box appears.

2 Click New.

   The Define Connection dialog box appears.

3 Make sure you've defined a run-time ADO or ColdFusion connection. The connection should be called connScaal on the Run-Time tab.

   If you haven't yet defined a run-time ADO or ColdFusion connection, see "Create a database connection" on page 27.

4 Click the Design-Time tab, then click Same as Run-Time to deselect it.

5 In the Type pop-up menu, select JDBC.

6 Enter the JDBC connection information.

Here is the information required if the tutorial database resides on a Windows NT Server with the RmiJdbc server component installed:

Driver:	RmiJdbc.RJDriver
Username:	*my_username*
Password:	*my_password*
URL:	jdbc:rmi://*my_server_name*/jdbc:odbc:*my_DSN*

If you're unsure of your server's name (*my_server_name*), consult your system administrator. To create a DSN (*my_DSN*), see "Setting up a DSN (ODBC)" on page 70.

For example, if the scaalcoffee.mdb database file resides on a Windows NT Server system called ANAKIN-3 and you set up a DSN called *scaalcoffee* on the server, enter the following information on the Macintosh:

Driver:	RmiJdbc.RJDriver
Username:	*my_username*
Password:	*my_password*
URL:	jdbc:rmi://ANAKIN-3/jdbc:odbc:scaalcoffee

7 Click the Test button to test the connection.

8 Click OK if you get a message informing you the connection test was successful, then click OK to close the Define Connection dialog box.

9 Click Done to close the Connections dialog box.

Create a master page

You're now ready to create dynamic pages for your Web application. In this part of the tutorial, you'll create a master page that displays press release summaries. Each summary will have a link taking readers to a detail page displaying the full text of the press release. (You'll work on the detail page later in the tutorial.) The master page has already been designed for you; all you need to do is make it dynamic.

The press releases are contained in the database file, scaalcoffee.mdb. By now you should have a connection to this database. (If not, see "Create a database connection" on page 27. The tutorial will not work without a connection to this database.)

Creating a dynamic page consists of three steps:

- Defining a recordset.
- Adding dynamic content to the page.
- Adding server behaviors to make the page work properly. For this page, you'll add two server behaviors: one allowing the page to display more than one record, and one allowing a link to pass certain information to the detail page.

Define a recordset for the master page

A recordset is a subset of data extracted from one or more tables in a database. It acts as a source of data for your dynamic pages.

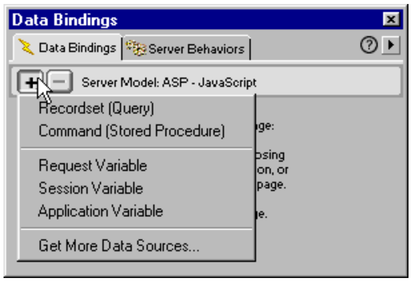
You can define a recordset to include all the fields and records of a database table, or only certain fields and records. When defining recordsets, strive to include only the fields and records your application needs to work. This will improve the performance of your application.

You'll begin by opening the predesigned master page.

1. Choose Window > Site Files to switch to the Site window.
2. Make sure your Tutorial site is selected, and double-click the extranet_news file in the Local Folder pane.

   The Corporate News page opens.

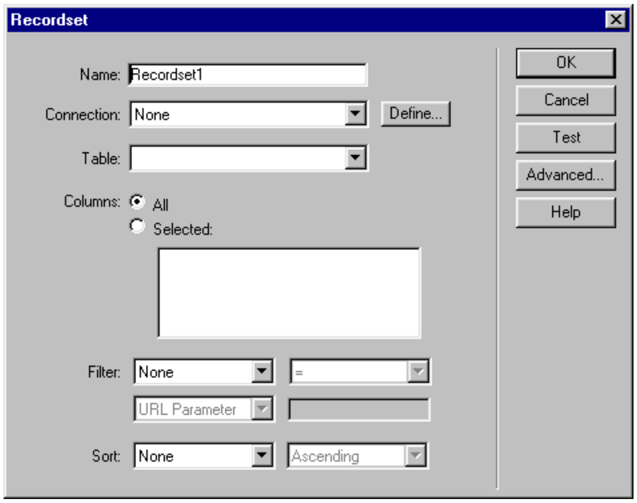
3 In the Data Binding inspector (Window > Data Bindings), click the plus (+) button.



4 Select Recordset (Query) from the pop-up menu.

5 If the advanced Recordset dialog box appears, click Simple.

The simple Recordset dialog box appears.


Recordset

Name: Recordset1

Connection: None

Table:

Columns: (●) All ( ) Selected:

Filter: None =

URL Parameter

Sort: None Ascending

6 In the Name text box, enter **rsReleaseSummaries**.

A common practice is to add the prefix *rs* to recordset names to distinguish them from other object names in your code.

Note: Do not use spaces or special characters in recordset names.

7 Select the connScaal connection from the Connection pop-up menu.

The connScaal connection should be the default selection. If connScaal doesn't appear in the list, click the Define button to create it. For instructions, see "Create a database connection" on page 27.

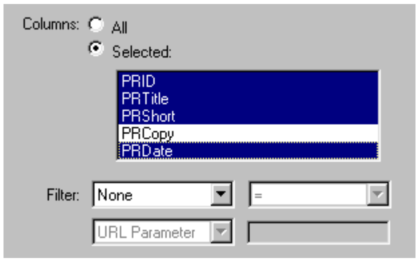
8 In the Table pop-up menu, select PressReleases.

The pop-up menu displays all the tables in the Scaal Coffee company's database.

9 In the Columns area, click the Selected option to choose only certain columns (or fields) in the PressReleases table.

You need four fields for your page: the PRID field, which contains numbers to help you identify records; the PRTitle field, which contains the titles of the press releases; the PRShort field, which contains summaries of the press releases; and the PRDate field, which contains the dates the press releases were issued.

10 Control-click (Windows) or Command-click (Macintosh) the following fields in the list to include them in the recordset: PRID, PRTitle, PRShort, and PRDate.



Columns: All [✓ Selected:]

PRID
PRTitle
PRShort
PRCopy
PRDate

Filter: None =

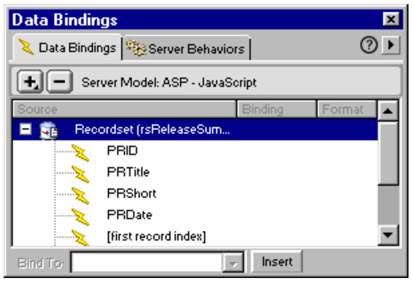
URL Parameter

11 Click Test to test the recordset.

A recordset appears containing data extracted from the database. Click OK to close it.

12 Click OK and save your work (File > Save).

UltraDev adds the recordset to your list of available data sources in the Data Binding inspector. To view the fields you defined for the recordset, expand the recordset branch.



Data Bindings

Data Bindings | Server Behaviors

Server Model: ASP - JavaScript

Source	Binding	Format
Recordset (rsReleaseSum...		
PRID		
PRTitle		
PRShort		
PRDate		
[first record index]		

Bind To: Insert

Add dynamic content to the master page

After defining a recordset, you can use its fields as sources of dynamic content for your page. For the master page, you'll use fields containing title, summary, and date information.

1 Make sure the Data Binding inspector is open (Window > Data Bindings) and lists the rsReleaseSummaries recordset you just defined. Expand the recordset's branch to see the data sources you need—namely, PRTitle, PRShort, and PRDate.

   If these fields don't appear in the list, click the plus (+) button to define a new recordset. For instructions, see "Define a recordset for the master page" on page 35.

   Next, you'll add the press release titles to the page.

2 On the page, select the words "Release title" in the table under Corporate News.

3 In the Data Binding inspector, select PRTitle and click Insert, or drag and drop PRTitle onto the text you selected on the page.

   A placeholder replaces the text selection on the page and the necessary server-side scripts are added to the page's HTML source code. When the server runs the page, the placeholder will be replaced with data from the recordset.

   Next, you'll add the press release summaries to the page.

4 In the table on the page, double-click the word "Summary" to select it.

5 In the Data Binding inspector, select PRShort and click Insert, or drag and drop PRShort onto the text you selected on the page.

   Now you'll add the press release dates to the page.

6 In the table on the page, double-click the word "Date" to select it.

7 In the Data Binding inspector, select PRDate and click Insert, or drag and drop PRDate onto the text you selected on the page.

8 Choose View > Live Data to display the dynamic content.

   The Live Data window opens showing data contained in your recordset. In this window, you can adjust the page's layout, add or delete dynamic content, and even change the page's server behaviors.

9 Choose View > Invisible Elements to remove the highlighting applied to the dynamic content.

   The highlighting may affect how dynamic content is displayed and thus give you an inaccurate picture of the page.

10 Save your work (File > Save).

If you make a mistake, open the Server Behavior inspector (Window > Server Behaviors), select the dynamic content (one of the Dynamic Text items), and click the minus (-) button to delete it.

Add a repeated region to the master page

The next step in building a dynamic page is adding the necessary server behaviors to make the page work. A server behavior is a set of instructions the server executes at run time.

Here's how the master page should work: Users should see the summaries of all the press releases on a single page, and users should be able to view the full text of any press release they choose.

To make the page work like this, you need two server behaviors: one to display more than one record on the same page, and one to open the detail page displaying the full text of a release. (You'll work on the detail page later in the tutorial.)

In this part of the tutorial, you'll give your page the ability to display more than one record by creating a repeating region. To do so, you need the Repeat Region server behavior, which is capable of repeating the page element that holds the data. In this case, the page element is a table.

When the server runs the page, the Repeat Region server behavior will repeat the table to accommodate all the records in the rsReleaseSummaries recordset. Each table will display the content of exactly one record.

1 Deselect Live Data in the View menu to switch to the Document window.

2 Select the table holding the three data placeholders by clicking any placeholder in the table and choosing Modify > Table > Select All.

3 Open the Server Behavior inspector (Window > Server Behaviors).

4 Click the plus (+) button and select Repeat Region.

5 In the Repeat Region dialog box, make sure the rsReleaseSummaries recordset is selected, then select All records.

6 Click OK.

7 Choose View > Live Data to view the repeating effect.

8 Save your work (File > Save).

If you make a mistake, open the Server Behavior inspector (Window > Server Behaviors), select the repeated region from the list, and click the minus (-) button to delete it.

Add a link to open the detail page

The master page should allow users to open a detail page displaying the full text of any press release. (You'll work on the detail page later in the tutorial.) Adding a standard link to open the detail page is not enough: to retrieve the correct record, the detail page needs to know which release the user selected on the master page. In other words, the master page must pass information to the detail page.

Use the following steps to add a link that passes information from the master page to the detail page.

1 Deselect Live Data in the View menu to switch to the Document window.

2 On the master page, click the {rsReleaseSummaries.PRTitle} placeholder to select it.

3 Make sure the Server Behavior inspector is open (Window > Server Behaviors).

4 Click the plus (+) button and select Go to Detail Page from the pop-up menu.

   The Go to Detail Page dialog box appears.

5 In the Detail Page text box, click Browse, select the predesigned detail page, extranet_news_article, and click OK (Windows) or Open (Macintosh).

6 In the Pass URL Parameter text box, enter id.

   You are telling the page to pass a parameter called id to the detail page identifying the press release selected by the user. The server will set the parameter's value to the value specified in the Recordset and Column pop-up menus. In this case, id should be set to the value of the PRID field in the rsReleaseSummaries recordset.

7 Click OK and save your work (File > Save).

When the user clicks the linked release title on the page, not only will the detail page open, but information identifying the record the user chose will be passed to the detail page so it can display the correct press release.

Note: Links don't work in the Live Data window. To view this effect, you must upload the page to your Web server and open it in a browser.

You have finished building the master page of press release summaries. Now you need a detail page that can display the full text of any press release selected on the master page.

Create a detail page

By now you should have created a master page that displays summaries of Scaal press releases. In this part of the tutorial, you'll create a detail page that displays the full text of any press release selected by the user in the master page. The page has already been designed for you; all you need to do is make it dynamic.

As before, creating a dynamic page consists of three steps:

- Defining a recordset.
- Adding dynamic content to the page.
- Adding server behaviors to make the page work. For this page, you'll add three server behaviors: one allowing the page to find and display a specific record, one allowing users to move through records, and one hiding navigation links when they're not needed.

Define a recordset for the detail page

For this page, your recordset only needs two fields: the PRID field, which holds the ID number of each press release, and the PRCopy field, which holds the full text of each press release.

You'll begin by opening the predesigned detail page.

1 Choose Window > Site Files to switch to the Site window.

2 Make sure your Tutorial site is selected, and double-click the extranet_news_article page in the Local Folder pane.

   The News Article page opens.

3 Open the Data Binding inspector (Window > Data Bindings).

4 Click the plus (+) button and select Recordset (Query) from the pop-up menu.

5 If the advanced Recordset dialog box appears, click Simple.

   The simple Recordset dialog box appears.

6 In the Name text box, enter **rsReleaseText**.

   Note: Do not use spaces or special characters in recordset names.

7 Select the connScaal connection from the Connection pop-up menu.

8 In the Table pop-up menu, select PressReleases.

   The pop-up menu displays all the tables in the Scaal Coffee company's database.

9 In the Columns area, click the Selected option to choose only certain columns (or fields) in the PressReleases table.

10 Control-click (Windows) or Command-click (Macintosh) the PRID and PRCopy fields in the list to include them in the recordset.

11 Click Test to test the recordset.

   A recordset appears containing data extracted from the database. Click OK to close it.

12 Click OK and save your work (File > Save).

Add dynamic content to the detail page

After defining the recordset, you can use its fields as sources of dynamic content for your page. The detail page needs only one source of dynamic content: the PRCopy field, which contains the full text of the press releases.

1 Make sure the Data Binding inspector is open (Window > Data Bindings) and lists the rsReleaseText recordset you just defined. Expand the recordset's branch to see the data sources you need—namely, the PRCopy field.

   If the field doesn't appear in the list, click the plus (+) button to define a new recordset. (For instructions, see "Define a recordset for the detail page" on page 42.)

2 In the Data Binding inspector, select PRCopy and drag and drop it into the large, blank table cell to the right of the "news article" table cell.

3 Choose View > Live Data to display the dynamic content.

   The Live Data window displays the text of the first press release in the recordset.

4 Save your work (File > Save).

If you make a mistake, open the Server Behavior inspector (Window > Server Behaviors), select the dynamic content (`Dynamic Text{rsReleaseText.PRCopy}`), and click the minus (-) button.

Enable the page to find and display a release

The next step in building a dynamic page is adding the necessary server behaviors to make the page work.

When the detail page opens in a browser, it should contain the full text of a press release the user selected on the master page. (You worked on the master page earlier in this tutorial.) To make the page work this way, you use the Move to Specific Record server behavior, which finds and displays the press release the user selected on the master page.

1 Make sure the Server Behavior inspector is open (Window > Server Behaviors).

2 Click the plus (+) button and select Move to Record > Move to Specific Record.

  The Move to Specific Record dialog box appears.

3 In the Move to Record In pop-up menu, make sure the rsReleaseText recordset is selected.

4 In the Where Column pop-up menu, make sure the PRID column is selected.

  The previous page you worked on passed the ID number (PRID) of a release to the detail page. By specifying the PRID column, you tell the detail page to look in the PRID column of the current recordset to find an ID number matching the one sent by the master page. When the behavior's server-side script finds a match, it displays the corresponding press release text.

5 Click OK.

Activate the navigation links

The detail page should also allow users to move forward and backward through the press releases. The page has “< Prev” and “Next >” text strings for that purpose. Your job is to activate these text strings so that when a user clicks one, the page displays the text of another press release.

To make this work, you need a pair of server behaviors for moving back and forth through the press releases.

1 Select the text string “< Prev” on the page.

2 In the Server Behavior inspector, click the plus (+) button and select Move to Record > Move to Previous Record.

   The Move to Previous Record dialog box appears.

3 In the Recordset pop-up menu, make sure the rsReleaseText recordset is selected, then click OK.

4 Save your work (File > Save).

Repeat the procedure for the “Next >” text string, only this time select Move to Record > Move to Next Record from the pop-up menu.

If you make a mistake, select the server behavior in the Server Behavior inspector and click the minus (-) button to delete it.

Hide the navigation links

It’s a good idea to hide the “< Prev” link when the user is viewing the first press release. Similarly, when the user is viewing the last press release, you want to hide the “Next >” link because there are no more releases to display. You use the Hide Region server behavior to add this functionality to your page.

1 In the page, select the “< Prev” text string.

2 In the Server Behavior inspector, click the plus (+) button and select Hide Region from the pop-up menu.

3 In the Hide Region dialog box, select the If First Record option.

   This instructs the server to hide “< Prev” if the page is asked to display the first record in the recordset.

4 Click OK.

5 Repeat the procedure, this time selecting the “Next >” text string and the If Last Record option.

6 Save your work (File > Save).

With these finishing touches, the dynamic pages for the Scaal corporate news area are done. In the Site window, select the master and detail pages, click Put to upload them to your server, then open the extranet_news file in your browser.

CHAPTER 2
Dreamweaver UltraDev Basics

The Dreamweaver UltraDev work area is basically the same as Dreamweaver's work area, with a few additions. If you're familiar with Dreamweaver, you already have the skills to get started with Dreamweaver UltraDev. However, to get the most out of UltraDev, you need to become familiar with some basic concepts:

- Simple database concepts such as fields and recordsets
- How dynamic pages work in general
- The workflow involved in creating a dynamic page in UltraDev

This chapter looks briefly at each of these topics. First, here's some basic terminology:

A Web application is a collection of static and dynamic pages that interact with each other and with various resources on a Web server, including databases.

A dynamic page is a Web page modified at run time by the Web server before being sent to a browser.

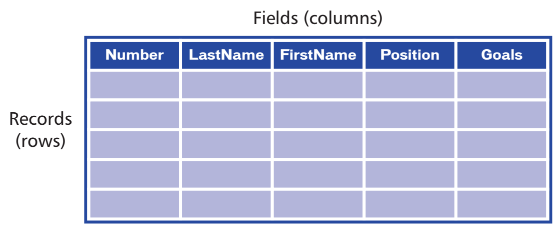
A server technology is a technology such as ASP, JSP, or ColdFusion that gives the Web server the ability to modify a Web page at run time.

A server behavior is the set of instructions the server executes at run time. Server behaviors are inserted in the Web page at design time.

About databases

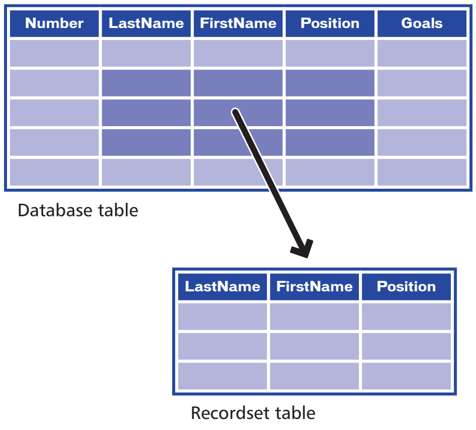
The building block of a database is the record. A record is a collection of related data treated as a single entity. For example, a hockey trading card could be called a record: it brings together the name, photograph, team, and statistics of one player. Using database terms, each of these related pieces of information would be called a field: each hockey card "record" has a name field, a photograph field, a team field, and various statistic fields.

A collection of records that share the same fields is called a table because this kind of information can easily be presented in table format: each column represents a field and each row represents a record. In fact, the word *column* is synonymous with the word *field*, and the word *row* is synonymous with the word *record*.



A database can contain more than one table, each with a unique name. These tables can be related or independent from one another.

A subset of data extracted from one or more tables is called a recordset. A recordset is also a table because it's a collection of records that share the same fields. For example, a hockey team roster listing the names and positions of the players could be called a recordset: it consists of a subset of all the possible information about the players, including goals, assists, penalty minutes, and so on.



To create a recordset, you run a database query. A query consists of search criteria. For example, the query can specify that only certain fields be included in the recordset, or that only certain records be included. For more information, see "Defining a recordset" on page 77.

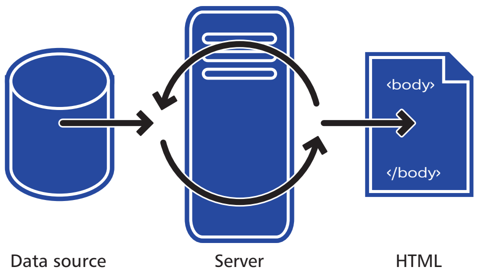
About dynamic pages

A Web application is a collection of static and dynamic pages. Dynamic pages resemble static pages in all aspects except one: where some of their scripts are run. Both kinds of pages are plain text (ASCII) files, contain HTML, and sit on a server waiting to be served up to a Web browser. Both can contain scripts written in languages like VBScript or JavaScript. However, scripts in a dynamic page can run on a server while those in a static page cannot.

Note: Strictly speaking, a "static" page may not be static at all. For example, a rollover image or a Flash movie can make a static page come alive. However, this guide refers to a page as static if it doesn't contain scripts to be executed on a server.

Scripts that run on a server, or server-side scripts, give you the ability to work with resources such as databases on the server. For example, before a page is served up to the browser, a server-side script in the page could instruct the server to extract data from a database and insert it into the page's HTML. In UltraDev, server-side scripts are called server behaviors.

Here's the route data takes to make it into the HTML of your pages:



In effect, the server creates a part of your page during run time and adds it to the parts you designed earlier in UltraDev. The resulting page is then sent to the browser.

UltraDev supports the following server technologies: Microsoft's Active Server Pages (ASP), Sun's JavaServer Pages (JSP), and Allaire's ColdFusion.

About the UltraDev workflow

All dynamic pages start as static pages. You build a static page, then transform it into a dynamic one. For example, you could create a page with a logo, some introductory text, a site map, and a table. Next, you could modify the table to display information from a database.

The workflow for creating a dynamic page consists of four or five steps, depending on whether you want to use a database with your application:

- Build a static page
- Create a recordset (if you're using a database)
- Add dynamic content to the page
- Add server behaviors to make the page work as intended
- Edit and debug the page

This section describes the workflow in general terms. Later chapters describe how to perform the steps in more detail.

To see an animated overview of the UltraDev workflow, choose Guided Tour of UltraDev in the Help menu, and select UltraDev Workflow from the list of Show Me movies.

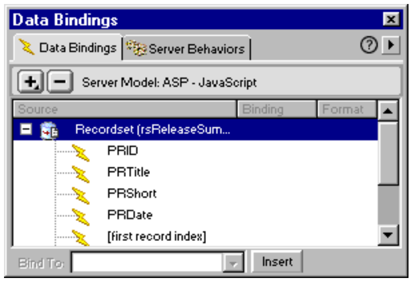
Building a static page

The first step in creating a dynamic page is to create a static one. All of Dreamweaver's design tools are at your disposal. To find out how to use them, see Dreamweaver Help (Help > Using Dreamweaver) or the *Using Dreamweaver* guide.

Defining a recordset

If you decide to use a database with your application, the next step in the process is to define a recordset to temporarily hold data from the database. Your application cannot work directly with a database: it must work through the intermediary of a recordset. For detailed procedures, see "Defining a recordset" on page 77.

Any recordset you define is added to a list in the Data Binding inspector:



You can use this inspector to add dynamic content to your page.

Adding dynamic content

After adding a recordset or other data source to the Data Binding inspector, you can add dynamic content to your page. Dreamweaver UltraDev lets you add dynamic content without worrying about the underlying server-side scripts inserted into the page. You simply need to specify where you want to put the content, and what you want that content to be.

First, you specify where you want to put the dynamic content. In UltraDev, you can put dynamic content anywhere in the page's HTML:

- You can place it at the insertion point.
- You can replace a text string with it.
- You can insert it in an HTML attribute. For example, dynamic content can define the `SRC` attribute of an image or the `VALUE` attribute of a form field.

Second, you specify what you want the dynamic content to be. You can choose from any data source listed in your Data Binding inspector. For example, you could choose a field in a recordset, a value submitted by a requesting page, or a value of a server object. After making your selection, UltraDev inserts a server-side script into the page that instructs the server to transfer the data from the selected data source to the page's HTML code.

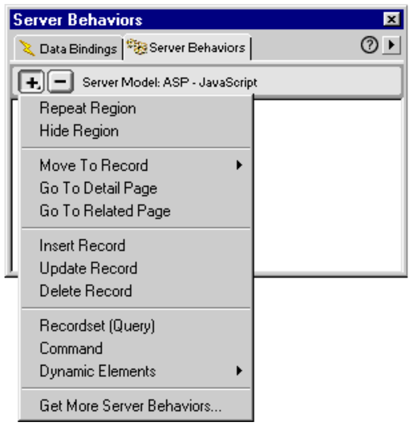
For detailed procedures, see "Adding Dynamic Content" on page 91.

Activating a dynamic page

The next step in the process is supplying the page with the "intelligence" needed to make it work as intended. In many cases, you supply this intelligence by adding server behaviors to the page. A server behavior is a VBScript, JavaScript, Java, or ColdFusion script that runs on a server instead of in a browser.

Dreamweaver UltraDev provides a number of predefined server behaviors to power your user interface. You can also write your own server behaviors or install server behaviors written by other people.

You add server behaviors to your pages using the Server Behavior inspector.



For detailed procedures, see "Activating Dynamic Pages" on page 107.

Editing and debugging the page

The final step in the process is to test the page and make changes to it as required. Dreamweaver UltraDev gives you three editing environments: Dreamweaver's traditional visual editing environment, UltraDev's live-data editing environment, and a source code editing environment.

Dreamweaver's traditional visual editing environment, the Document window, gives you a rough idea of what the page will look like in the browser *before* dynamic content is added to the page. This is an ideal editing environment for pages with no dynamic content. However, since dynamic content may fundamentally alter the way a page looks and works, this environment is not ideal for dynamic pages.

UltraDev's live-data editing environment, the Live Data window, does display dynamic content. More important, it lets you work on the page while the dynamic content is displayed.

Finally, you can edit the HTML source code and the server-side scripts directly in the HTML Source inspector or in the Quick Tag Editor.

For more information, see "Editing and Debugging Dynamic Pages" on page 127.

CHAPTER 3
Laying the Groundwork for an Application

Before you start building your Web application, you need to lay the groundwork for it. You must organize your files and define your site in Dreamweaver UltraDev. You must tell UltraDev what server technology you're using—ASP, JSP, or ColdFusion—so it can insert the proper kinds of server-side scripts and tags into your pages. You must also tell UltraDev what Web server to use to power the Live Data window. Finally, if you plan to use a database with your application, you must create a database connection so your application can find the database and interact with it.

Organizing files and defining sites

Dreamweaver UltraDev gives you the ability to manage your files and to transfer files between your local disk and your Web server at the click of a button. To benefit from these features, you must do the following:

- Create a folder on your local disk to store the files you'll create for your application. You may want to create subfolders to store image files and other assets.
- Define a local site. The local site is the folder you create on your local disk to store your files. If you don't define a local site, UltraDev will not work properly. For more information, see "Defining a local site" on page 56.
- Define a remote site. The remote site is the folder on the Web server destined to hold your files. For more information, see "Defining a remote site" on page 58.

Defining a local site

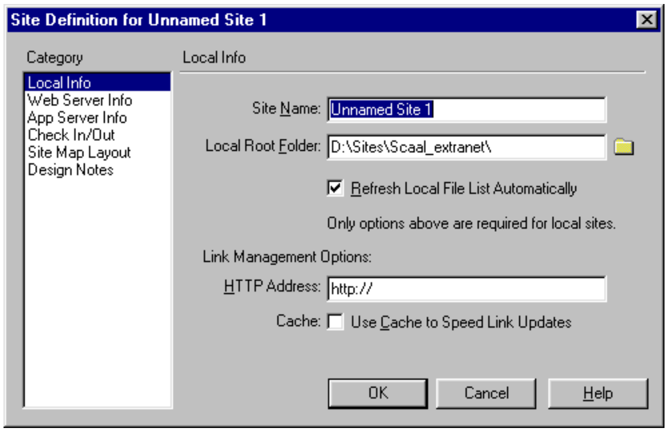
The local site is the folder you use to store the site files on your local disk. If you don't define a local site, Dreamweaver UltraDev will not work properly. Defining a local site is a one-time-only requirement.

If you're a JSP user, you need to include an error-handling file with the rest of the site files on your remote server. This section describes how to create the file and upload it to your remote server.

To define a local site:

1 Choose Site > New Site.

2 In the Site Definition dialog box, select Local Info from the Category list.



Local Info

Site Name: Unnamed Site 1

Local Root Folder: D:\Sites\Scaal_extranet\

☑ Refresh Local File List Automatically

Only options above are required for local sites.

Link Management Options:

HTTP Address: http://

Cache: ☐ Use Cache to Speed Link Updates

3 Enter a name in the Site Name field.

The site name will appear in the Site window and in the Site > Open Site submenu once you're done defining the local site.

4 Specify the folder on your local disk where the application files will be stored. In the Local Root Folder field, enter a path or click the folder icon to browse to and select the folder.

5 Complete the other options in the Local Info dialog box, though these are not required for a local site.

For more information on these options, see Creating a local site in Dreamweaver Help (Help > Using Dreamweaver) or in the *Using Dreamweaver* guide.

6 Click OK.

For more information, see Creating Sites and Documents in Dreamweaver Help or in the *Using Dreamweaver* guide.

To create an error-handling file for JSP:

1. In any text editor, create a plain text (ASCII) file called error.jsp.
2. Type the following code in the file:

   ```
   <%@page LANGUAGE="Java" isErrorPage="true"%>
   <%
       //Put your error-handling code here
   %>
   ```

   For example, the error-handling code could be an exception object:

   ```
   out.println(exception.GetMessage());
   ```
3. Save the file and place it in your local site's root folder to be uploaded to your JSP server with the rest of your site files.

Defining a remote site

Dynamic pages contain scripts that can only run on a server. This means you must upload these pages to a server for the scripts to work. (In the Live Data window, UltraDev uploads a temporary copy of the page to the Web server for you and displays the results.)

You can send files to your server via FTP—you don't need a separate FTP client to do so. You can also transfer files internally to a server located on a local disk or on a network. However, before you can take advantage of either the FTP or file-transfer feature, you must define a remote site in UltraDev.

The remote site you define must be on a server that supports ASP, JSP, or ColdFusion. For more information, see "What you need to run dynamic pages" on page 9.

To define a remote site:

1 Choose Site > Define Sites.

  A dialog box appears listing currently defined sites.

2 Select your site and click Edit. (If you have no currently defined sites, create a local site before going any further. See "Defining a local site" on page 56.)

3 In the Category list on the left, click Web Server Info.

4 In the Server Access text box, select one of the following options: Local/Network or FTP. For more information, see Associating a remote server with a local site in Dreamweaver Help (Help > Using Dreamweaver) or in the *Using Dreamweaver* guide.

5 Complete the other options in the dialog box:

- If you chose FTP, enter the host name of the FTP host.
- Enter the name of the host directory at the remote site. The host directory is where documents visible to the public are stored.
- If you chose FTP, enter the login name and password used to connect to the FTP server.
- Select the appropriate firewall options.

6 Click OK.

For more information, see Setting up a remote site in Dreamweaver Help (Help > Using Dreamweaver) or in the *Using Dreamweaver* guide.

Configuring Dreamweaver UltraDev

Dreamweaver UltraDev needs answers to two questions before you can start building your Web application:

- What server technology are you using?
- What Web server do you want UltraDev to use to display dynamic pages in the Live Data window?

Specifying a server technology

You must tell UltraDev what server technology you're using so it knows what kind of server-side scripts to insert into your pages. If you have a ColdFusion server, UltraDev will insert the necessary ColdFusion tags and scripts in the page. If you have a Web server implementing Sun's JavaServer Pages specification, then UltraDev will insert Java code. If you have a server implementing Microsoft's Active Server Pages specification, you can have UltraDev insert either VBScripts or JavaScripts.

You specify a server technology for a site as a whole, not for individual pages. This ensures that all the pages in your application are compatible.

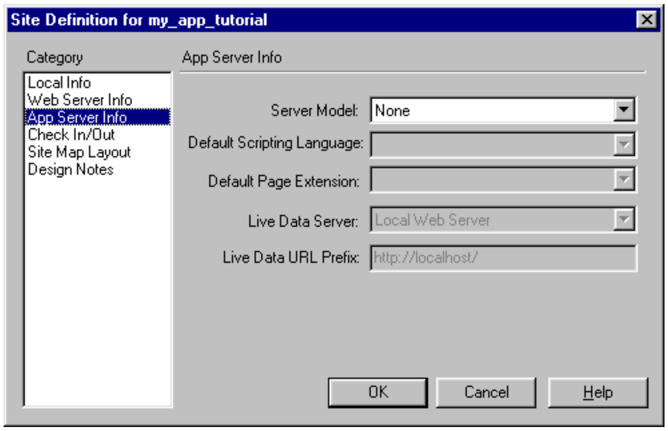
Note: You must define a site before you can specify a server technology. See "Defining a local site" on page 56.

To specify a server technology:

1 Choose Site > Define Sites.

2 In the Define Sites dialog box, select the site you defined for your application and click Edit.

3 In the Site Definition dialog box, select App Server Info from the Category list.



Site Definition for my_app_tutorial

Category: Local Info, Web Server Info, App Server Info, Check In/Out, Site Map Layout, Design Notes

App Server Info

Server Model: None
Default Scripting Language:
Default Page Extension:
Live Data Server: Local Web Server
Live Data URL Prefix: http://localhost/

4 Complete the following options:

- In the Server Model pop-up menu, select ASP, JSP, or ColdFusion.
- If you chose ASP as your server model, set the Default Scripting Language to either VBScript or JavaScript. (If you chose JSP or ColdFusion, this option is set for you.)
- For Default Page Extension, accept the default file extension or enter one of your own. The file extension will be added to every page you create for the site.

Note: The default .asp, .jsp, or .cfm extension will not affect your static pages. However, changing the extension to .htm or .html will disable any dynamic page you create from then on. You will have to manually change the extension of the dynamic pages to .asp, .jsp, or .cfm, as appropriate.

5 Keep the Site Definition dialog box open if you want to configure the Live Data window. Otherwise, click OK, then click Done.

Configuring the Live Data window

Where the Document window uses placeholders to represent dynamic content on the page, the Live Data window displays the actual dynamic content on the page. More significantly, while the dynamic content is displayed you can work on the page.

To get the Live Data window to work, you must specify a Web server capable of executing dynamic pages. Whenever you switch to the Live Data window, a temporary copy of the open document is transferred to the specified Web server for processing. The resulting page is returned and displayed in the Live Data window, and the temporary copy on the Web server is deleted.

You must manually copy any dependent files, such as JSP class files, to the Web server. UltraDev does not automatically copy dependent files to the server.

To specify a server to run pages for the Live Data window:

1. Choose Site > Define Sites.

   A dialog box appears listing currently defined sites.

2. Select your site and click Edit.

3. In the Category list on the left, click App Server Info.

4. In the Live Data Server pop-up menu, select the location of the server you want UltraDev to use to execute dynamic pages when you switch to the Live Data window.

- Choose Remote Web Server to use the server that runs your remote site. The remote site is defined in the Web Server Info category. See “Defining a remote site” on page 58.

  Choose Remote Web Server even if your “remote site” is located on the local disk if the following condition applies: the remote site files are stored in a different folder than the site’s local folder (the root directory in the right pane of the Site window).

- For Windows users, choose Local Web Server if you defined your site’s local folder (the root directory in the right pane of the Site window) as a published directory in your Web server (normally Personal Web Server).

5. In the Live Data URL Prefix text box, enter the URL of your site.

   The URL prefix is the URL for the folder at the root of your site, not the full URL of any subfolder containing your documents.

   If the site is published on your local system, enter the site’s alias (not the full path) at the end of the http://localhost/ string. Example:

   http://localhost/my_site_alias/

6. Click OK, then click Done.

Working with database connections

If you plan to use a database with your Web application, you need to create at least one database connection. Without one, UltraDev won't know where to find the database or how to connect to it.

You may not intend on using a database at all with your application. For example, you may want to use data sources such as user-supplied values or server object values. If you don't plan on using a database, you can skip this section.

A database connection is a set of parameters you define to establish a link to a database. UltraDev places these parameters in your page's server-side scripts if the page is designed to interact with a database.

UltraDev lets you create any number of connections; the program stores them for you until you need them. You can also edit or delete any existing connection.

Note: The connection information is stored in the connections.xml file in the following UltraDev folder on the local disk: Configuration\Connections (Windows) or Configuration:Connections (Macintosh). You can copy this file to other systems running UltraDev to duplicate the connections.

About run-time and design-time connections

UltraDev lets you create a run-time database connection and an optional design-time connection. Because your run-time connection is your deployment connection, you must always create one. You should create a design-time connection if it's impractical or impossible to access the deployment database at design time. If you use a file-based database program like Microsoft Access or Lotus Approach, you can place a copy of the database file on a system that's easily accessible, then create a design-time connection to it. While you work, UltraDev will display information from the database copy—information such as table and field names, stored procedures, and so on.

You must also create a design-time connection if you cannot connect to the database using the run-time technology. For example, Macintosh users developing ASP sites (thus using ADO technology to connect at run time) cannot use ADO to connect at design time because the Macintosh does not support ADO. Macintosh users must use JDBC technology to connect at design time (often connecting to the same database as the run-time connection).

For the run-time connection, you must create the following types of connections:

- If you're using ASP, create an ADO connection. See "Creating an ADO database connection" on page 64.
- If you're using JSP, create a JDBC connection. See "Creating a JDBC database connection" on page 66.
- If you're using ColdFusion, create a ColdFusion connection. See "Creating a ColdFusion database connection" on page 68.

For a design-time connection, you must specify the connection type that's appropriate for your database driver. For example, if you're using the RmiJdbc driver on your Macintosh, create a JDBC connection. If you're using the Microsoft Access driver in Windows, create an ADO connection.

If your database contains a schema or catalog, you can use it to restrict the number of database items displayed in UltraDev at design time. Choose Connections from the Modify menu, then click New or select an existing connection and click Edit. In the Define Connection dialog box, click the Design-Time tab, click Restrict, and enter the name of your schema or catalog.

Creating an ADO database connection

Create an ADO database connection for the run-time connection if you're using ASP. Create an ADO connection for the design-time connection if you're using an ODBC database driver such as the latest Microsoft Access, dBase, or Excel drivers for Windows, or if you're using an OLEDB driver such as the OLEDB driver for SQL Server.

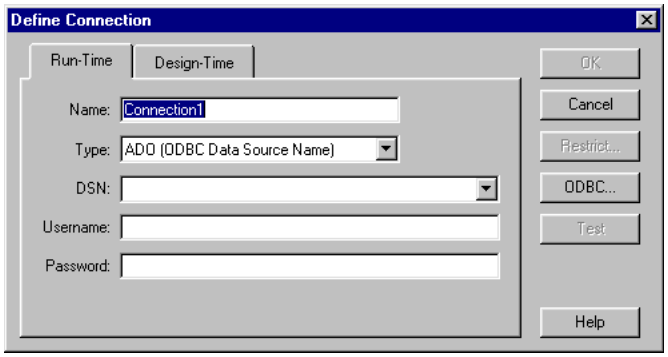
To create an ADO database connection:

1 In the Document or Live Data window, choose Connections from the Modify menu.

The Connections dialog box appears.

2 Click New.

The Define Connection dialog box appears.



Name: Connection1

Type: ADO (ODBC Data Source Name)

DSN:

Username:

Password:

3 Select the Run-Time or Design-Time tab, as appropriate.

If you're not sure which tab to select, see "About run-time and design-time connections" on page 63.

4 Enter a name for the connection.

5 Select a connection type as follows:

- For a DSN connection, select ADO (ODBC Data Source Name) from the Type pop-up menu, select a DSN, and enter a user name and password, if required. See “Setting up a DSN (ODBC)” on page 70.
- For a DSN-less ODBC connection, select ADO (Connection String) from the Type pop-up menu and enter a connection string to the database. For more information, see your ODBC driver vendor’s documentation, or consult your system administrator.
- For an OLEDB connection, select ADO (Connection String) from the Type pop-up menu and enter the OLEDB connection information. For more information, see your OLEDB driver vendor’s documentation, or consult your system administrator.

6 If you want to test the design-time connection, click Test.

7 Click OK.

Your new connection should now appear in the Connections dialog box.

8 Click Done to close the Connections dialog box.

Creating a JDBC database connection

Create a JDBC database connection for the run-time connection if you're using JSP. Create a JDBC connection for the design-time connection if you're using an JDBC database driver such as the RmiJdbc driver or the Oracle Thin JDBC driver.

To create a JDBC database connection:

1 In the Document or Live Data window, choose Connections from the Modify menu.

The Connections dialog box appears.

2 Click New.

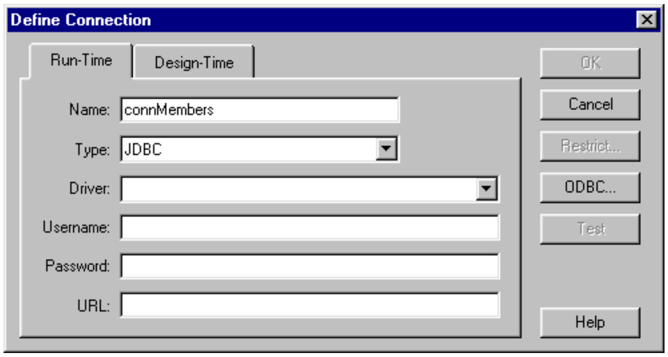
The Define Connection dialog box appears.

3 Select the Run-Time or Design-Time tab, as appropriate.

If you're not sure which tab to select, see "About run-time and design-time connections" on page 63.

4 Enter a name for the connection.

5 In the Type pop-up menu, select JDBC.



Name: connMembers
Type: JDBC
Driver:
Username:
Password:
URL:

6 Complete the Driver, Username, Password, and URL text boxes.

See the examples below. For driver-specific requirements, see the driver vendor's documentation or consult your system administrator.

7 If you want to test the design-time connection, click Test.

8 Click OK.

Your new connection should now appear in the Connections dialog box.

9 Click Done to close the Connections dialog box.

Here are example connections using different JDBC drivers:

- RmiJdbc driver on the Macintosh:

  Driver: RmiJdbc.RJDriver
  Username: my_username
  Password: my_password
  URL: jdbc:rmi://192.186.56.64/jdbc:odbc:orcl

- RmiJdbc driver in Windows:

  Driver: RmiJdbc.RJDriver
  Username: my_username
  Password: my_password
  URL: jdbc:rmi://ANAKIN/jdbc:odbc:orcl

- Oracle Thin JDBC driver on the Macintosh:

  Driver: oracle.jdbc.driver.OracleDriver
  Username: my_username
  Password: my_password
  URL: jdbc:oracle:thin:scott/tiger@192.186.56.64:1251:orcl

- Oracle Thin JDBC driver in Windows:

  Driver: oracle.jdbc.driver.OracleDriver
  Username: my_username
  Password: my_password
  URL: jdbc:oracle:thin:scott/tiger@ATREIDES:1251:orcl

- Inet JDBC driver on the Macintosh:

  Driver: com.inet.tds.TdsDriver
  Username: my_username
  Password: my_password
  URL: jdbc:inetdae:192.176.63.42:1343?database=pubs&user=edu&password=max

- Microsoft JDBC-ODBC Bridge in Windows:

  Driver: com.ms.jdbc.odbc.JdbcOdbcDriver
  URL: JDBC:ODBC:dsn=contacts;UID=admin;PWD=scully;

For more information on creating a database connection using your driver, consult the driver vendor's documentation.

Creating a ColdFusion database connection

Create a ColdFusion connection for the run-time connection if you're using ColdFusion.

To create a ColdFusion database connection:

1 In the Document or Live Data window, choose Connections from the Modify menu.

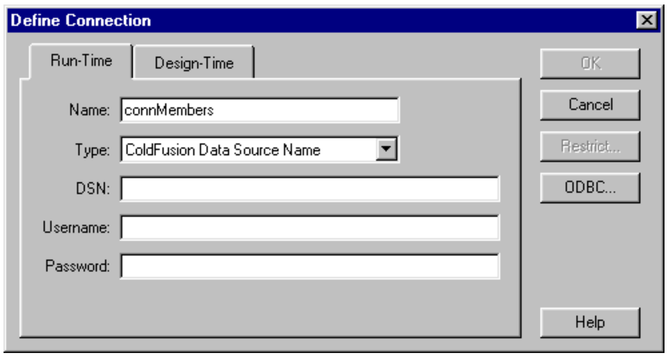
  The Connections dialog box appears.

2 Click New.

  The Define Connection dialog box appears.

3 In the Run-Time tab, enter a name for the connection.

4 In the Type pop-up menu, select ColdFusion.



Name: connMembers

Type: ColdFusion Data Source Name

DSN:

Username:

Password:

5 In the DSN text box, enter the data source name you defined in ColdFusion for the database.

  Note: Do not confuse the ColdFusion DSN with the ODBC DSN described in "Setting up a DSN (ODBC)" on page 70. You define the ColdFusion DSN in the ColdFusion Administrator on your server.

6 If required, complete the Username and Password text boxes.

  Username and password information for each ColdFusion Data Source is usually kept in the ColdFusion Administrator.

7 Click OK.

  Your new connection should now appear in the Connections dialog box.

8 Click Done to close the Connections dialog box.

Editing or deleting database connections

When you edit and delete database connections, you must also update the pages with the out-of-date or obsolete connections.

To edit a database connection:

1 In the Document or Live Data window, choose Connections from the Modify menu.

  The Connections dialog box appears.

2 Select a connection from the list and click Edit.

  The Define Connection dialog box appears.

3 Make the necessary changes to the connection, then click OK.

  Next you'll reestablish the connection in any recordset using the modified connection.

4 Open each page with a recordset or recordsets using the modified connection.

5 In the Data Binding inspector (Window > Data Bindings), double-click the name of the recordset.

6 In the Recordset dialog, select the connection again from the Connection pop-up menu, and click OK.

  The connection is reestablished.

To delete a database connection:

1 In the Document or Live Data window, choose Connections from the Modify menu.

  The Connections dialog box appears.

2 Select a connection from the list and click Delete.

  Next you'll specify a new connection for any recordset using the deleted connection.

3 Open each page with a recordset or recordsets using the deleted connection.

4 In the Data Binding inspector, double-click the name of the recordset.

5 In the Recordset dialog, select a new connection from the Connection pop-up menu, and click OK.

Setting up a DSN (ODBC)

This section applies only if your database is located on a system that supports ODBC data source names (DSNs)—systems such as Windows and Windows NT (but not the Macintosh). A DSN is a sort of shortcut to a database; it contains all the information needed to create a connection to a database.

You must set up DSNs on your local system or remote server before you can use them in UltraDev. Before you begin, make sure your system has the proper driver for your database. For a list of drivers on a Windows 95, 98, or NT system, choose Start > Settings > Control Panel, then double-click the ODBC Data Sources icon. (Depending on your system, the icon could also be called ODBC or 32bit ODBC.) When you click the Drivers tab, you'll see a list of drivers installed on the system. In Windows 2000, choose Start > Programs > Administrative Tools > Data Sources, then click the Drivers tab.

You can set up DSNs for databases created with retail database products such as Microsoft Access or Lotus Approach, or for databases created with industrial-strength database products such as Microsoft SQL Server or Oracle8i. You can even set up DSNs for Excel spreadsheets and text files, both of which are a form of database. In fact, if a driver exists for a given file format, you can create a DSN for that file and use the file as a source of data in your Web application.

To set up a DSN:

1 Open Windows' ODBC Data Source Administrator as follows:

- In Windows 95, 98, or NT, choose Start > Settings > Control Panel, then double-click the ODBC Data Sources icon. Depending on your system, the icon could also be called ODBC or 32bit ODBC.
- In Windows 2000, choose Start > Programs > Administrative Tools > Data Sources.
- In UltraDev's Define Connection dialog box, click the ODBC button.

2 In the ODBC Data Source Administrator dialog box, click the System DSN tab.

The tab displays the list of DSNs currently on your system.

3 Click Add to add a new DSN to the list.

The Create New Data Source dialog box appears listing all the drivers currently loaded on your system.

4 Select a driver from the list, then click Finish.

For example, if your database is a Microsoft Access file, select Microsoft Access Driver (*.mdb). If a driver for your product does not appear in the list, you'll have to download the driver from the vendor's Web site and install it.

5 Follow the onscreen instructions to create the DSN.

Once you're done, the new DSN is added to the list of system DSNs.

4

CHAPTER 4
Page Blueprints

A Web application can consist of many kinds of dynamic pages. The most common are search pages, results pages, detail pages, and record-editing pages (which allow users to insert, update or delete records in a database). Each type of page has different requirements in terms of HTML code, dynamic content, and server behaviors.

To aid your planning, the tables that follow list the building blocks for each kind of page and refer to the sections in this guide to help you meet each requirement.

Building a search page

A search page lets users conduct a database search. The page usually consists of fields for entering search criteria and a form button for submitting the criteria. Here are the minimum requirements for building this type of page:

	Requirements	Getting Help
HTML	A form with a Submit button	Creating Forms topic in the Dreamweaver book or help system “Gathering data in an HTML form” on page 115
Dynamic content	None	
Server behaviors	None	

To see an animated introduction to the process of building a search page in UltraDev, choose Guided Tour of UltraDev in the Help menu, and select Building a Search Page from the list of Show Me movies.

Building a results page

A results page displays records extracted from the database in response to a search. The page usually consists of an HTML table displaying a list of records found. Here are the minimum requirements for building this type of page:

	Requirements	Getting Help
HTML	At designer's discretion, but typically a table	Selected topics in the Dreamweaver book or help system
Dynamic content	Table content bound to a recordset that accepts search criteria	"Defining a recordset for a results page" on page 83
Server behaviors	Repeat Region behavior (if you want to display more than one result per page)	"Displaying more than one record" on page 108
	Move to Record behaviors (if you want record-navigation links)	"Creating record navigation links" on page 111
	Hide Region behaviors (if you want to hide some links on the first and last page)	"Hiding regions" on page 113
	Go to Detail Page behavior (if you want to link to a detail page)	"Going to a detail page" on page 116

To see an animated introduction to the process of building a results page in UltraDev, choose Guided Tour of UltraDev in the Help menu, and select Building a Results Page from the list of Show Me movies.

Building a detail page

A detail page typically shows additional information about a record listed on a results page. Here are the minimum requirements for building this type of page:

	Requirements	Getting Help
HTML	At designer's discretion	Selected topics in the Dreamweaver book or help system
Dynamic content	Page content bound to a recordset	Adding Dynamic Content topic in the UltraDev book or help system
Server behaviors	Move to Specific Record behavior	"Moving to a specific record in a detail page" on page 112
	Other Move to Record behaviors (if you want record-navigation links)	"Creating record navigation links" on page 111
	Hide Region behaviors (if you want to hide some links on the first and last page)	"Hiding regions" on page 113
	Go to Related Page behavior (if you want to go back to the Results page while maintaining state)	"Going to a related page" on page 117

To see an animated introduction to the process of building a detail page in UltraDev, choose Guided Tour of UltraDev in the Help menu, and select Building a Detail Page from the list of Show Me movies.

Building an insert page

An insert page lets users add new records to a database. The page consists of an HTML form for entering the data, and a form button for submitting it. Here are the minimum requirements for building this type of page:

	Requirements	Getting Help
HTML	A form with a Submit button	Creating Forms topic in the Dreamweaver book or help system
Dynamic content	None	
Server behaviors	Insert Record behavior	"Inserting a record in a database" on page 118

To see an animated introduction to the process of building such a page in UltraDev, choose Guided Tour of UltraDev in the Help menu, and select Building a Page to Insert Records from the list of Show Me movies.

Building an update page

An update page lets users edit records in a database. The page consists of an HTML form for modifying a record, and a form button for submitting the changes. This type of page usually works in tandem with a results page. (The user must find the record before he or she can update it.) Here are the minimum requirements for building this type of page:

	Requirements	Getting Help
HTML	A form with a Submit button	Creating Forms topic in the Dreamweaver book or help system
Dynamic content	Form objects bound to a single-record recordset	"Updating a record in a database" on page 119
Server behaviors	Update Record behavior	"Updating a record in a database" on page 119
	Move to Record behaviors (if you want record-navigation links)	"Creating record navigation links" on page 111
	Hide Region behaviors (if you want to hide some links on the first and last page)	"Hiding regions" on page 113

Building a delete page

A delete page lets users delete records from a database. This type of page usually works in tandem with a results page. (The user must find the record before he or she can delete it.) Here are the minimum requirements for building this type of page:

	Requirements	Getting Help
HTML	A form with a Submit button	Creating Forms topic in the Dreamweaver book or help system
Dynamic content	Form objects bound to a single-record recordset	"Deleting a record in a database" on page 122
Server behaviors	Delete Record behavior	"Deleting a record in a database" on page 122
	Move to Record behaviors (if you want record-navigation links)	"Creating record navigation links" on page 111
	Hide Region behaviors (if you want to hide some links on the first and last page)	"Hiding regions" on page 113

CHAPTER 5
Creating a Recordset

If you decide to use a database with your application, you can't work with the database directly: you must work through the intermediary of a recordset. For example, when you bind page attributes to data, you bind them to the data in the recordset, not in the database.

A recordset is a subset of records extracted from a database by a database query. A query consists of search criteria that determine what's included in the recordset and what's not. A query can produce a recordset that includes only certain fields, or only certain records, or a combination of both.

A recordset can also include all the records and fields of a database table. However, because your applications will rarely need to use every piece of data in a database, you should strive to make your recordsets as small as possible. A server temporarily holds the recordset in memory, then discards it when it's no longer needed. Therefore, smaller recordsets use up less memory than larger ones, and the server's performance may improve as a result. The basic guideline when defining recordsets is to include only the data your application needs.

Defining a recordset

A recordset is defined by a query, which is a statement composed of search criteria designed to find and extract information from a database. UltraDev uses the Structured Query Language to build queries. You don't need to know SQL (pronounced "sequel") to define a recordset in UltraDev. However, if you do know SQL, you can use it to define your recordsets.

Note: After you define a recordset, the data it contains won't be immediately visible in the Document window or the Live Data window. Data only becomes visible after you add a field from the recordset to your page and switch to the Live Data window (View > Live Data).

Defining a recordset without using SQL

If you're unfamiliar with SQL, you can define your recordsets using UltraDev's simple Recordset dialog box. Defining a recordset using this dialog can be as easy as selecting a connection and a database table from the pop-up menus.

Note: If you want to work with SQL, use the advanced Recordset dialog box described in "Defining a recordset using SQL" on page 80.

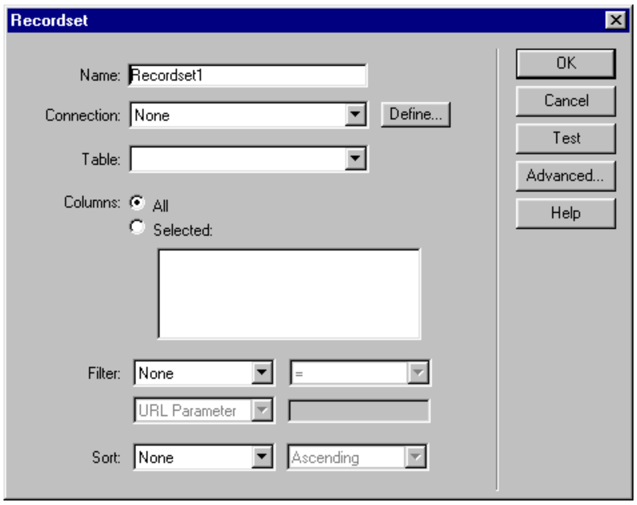
To define a recordset without using SQL:

1 Make sure the page that will use the recordset is open in the Document window or the Live Data window.

2 In the Data Binding inspector (Window > Data Bindings), click the plus (+) button and select Recordset (Query) from the pop-up menu.

3 If necessary, switch to the simple Recordset dialog box by clicking the Simple button.

The simple Recordset dialog box appears.



Name: Recordset1
Connection: None
Table:
Columns: [✓ All] Selected:
Filter: None =
URL Parameter
Sort: None Ascending

4 In the Name text box, enter a name for the recordset.

A common practice is to add the prefix *rs* to recordset names to distinguish them from other object names in your code—for example, rsPressReleases.

Note: Do not use spaces or special characters in recordset names.

5 Select a connection from the Connection pop-up menu.

If no connection appears in the list, click Define to create one. For more information, see “Working with database connections” on page 62.

6 In the Table pop-up menu, select the database table that will provide data to your recordset or receive data from it.

The pop-up menu displays all the tables in the connected database.

Note: UltraDev uses the selected database connection to populate the list of tables. If you specified a separate design-time connection, it uses that connection to populate the list. For more information, see “About run-time and design-time connections” on page 63.

7 To include only some of the table’s fields in the recordset, click Selected and choose the desired fields by Control-clicking (Windows) or Command-clicking (Macintosh) them in the list.

8 To include only some of the table’s records, complete the Filter section as follows:

- From the first pop-up menu, select a field in the table to compare against a benchmark value you define.
- From the second pop-up menu, select a conditional expression to compare the selected value in each record against the benchmark value.
- From the third pop-up menu, select Entered Value.
- In the fourth text box, enter the benchmark value.

If the specified value in a record meets your filtering condition, the record will be included in the recordset.

9 If you want the records to be sorted, select a field on which to sort, then specify whether the records should be sorted in ascending order (1, 2, 3... or A, B, C...) or descending order.

10 If you want, click Test to connect to the database and create an instance of the recordset.

If you created a filter, you will be prompted to provide a test value.

A table appears displaying the data in your recordset. Each row contains a record and each column represents a field in that record. Click OK to close the recordset.

11 If satisfied with your work, click OK.

UltraDev adds the recordset to your list of available data sources in the Data Binding inspector. Expand the recordset branch to view the fields you defined for it. You can use any of these fields as a source of dynamic content for your page. For more information, see “Adding Dynamic Content” on page 91.

Defining a recordset using SQL

If you're familiar with SQL or if you want to learn it, you can define your recordsets using UltraDev's advanced Recordset dialog box, described below. For a short primer on SQL statements, see "Writing simple SQL queries" on page 88.

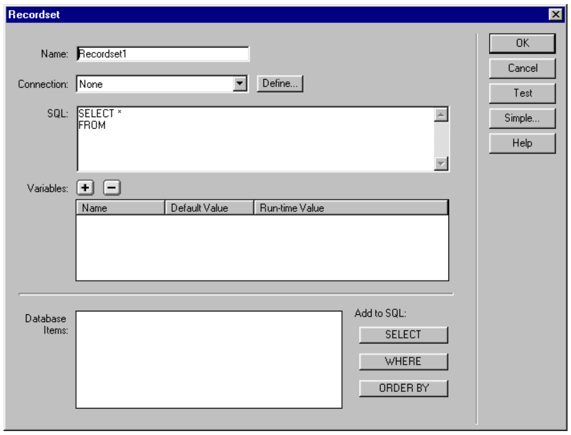
To define a recordset using SQL:

1 Make sure the page that will use the recordset is open in the Document window or the Live Data window.

2 In the Data Binding inspector (Window > Data Bindings), click the plus (+) button and select Recordset (Query) from the pop-up menu.

3 If necessary, switch to the advanced Recordset dialog box by clicking the Advanced button.

The advanced Recordset dialog box appears.


Name: Recordset1
Connection: None
SQL:
```
SELECT *
FROM
```

4 In the Name text box, enter a name for the recordset.

A common practice is to add the prefix *rs* to recordset names to distinguish them from other object names in your code—for example, rsPressReleases.

Note: Do not use spaces or special characters in recordset names.

5 Select a connection from the Connection pop-up menu.

If no connection appears in the list, click Define to create one. For more information, see "Working with database connections" on page 62.

6 Enter the SQL statement in the SQL text area.

To reduce the amount of typing, you can use the tree of database items at the bottom of the advanced Recordset dialog box. To use the tree, first make sure the SQL text area is blank. Next, expand the branches in the tree until you find the database object you need—a field in a table, for example. Select it and add it to your SQL statement by clicking one of the three buttons on the right side of the tree: SELECT, WHERE, and ORDER BY. Each of these buttons adds a clause to the SQL statement.

Note: UltraDev uses the selected database connection to populate the tree of database objects. If you specified a separate design-time connection, it uses that connection to populate the tree. For more information, see "About run-time and design-time connections" on page 63.

7 If you entered variables in the SQL statement, define their values in the Variables area by clicking the plus (+) button and entering the variable's Name, Default Value (the value the variable should take if no run-time value is returned), and Run-Time Value (usually a server object holding a value sent by a browser, such as an ASP Request object).

For example, suppose an HTML form on the requesting page has a field called Name. The run-time value for this field in ASP could be Request("Name"), Request.Form("Name"), or Request.QueryString("Name"), depending on the form method used (GET or POST). The run-time value for ColdFusion would be #Name#. The run-time value for JSP would be request.getParameter("Name").

8 If you want, click Test to connect to the database and create an instance of the recordset.

If successful, a table appears displaying the data in your recordset. Each row contains a record and each column represents a field in that record. Click OK to clear the recordset.

9 If satisfied with your work, click OK.

UltraDev adds the recordset to your list of available data sources in the Data Binding inspector. Expand the recordset branch to view the fields you defined for it. You can use any of these fields as a source of dynamic content for your page. For more information, see "Adding Dynamic Content" on page 91.

Here are two example SQL statements and how to create them using the tools at your disposal in the advanced Recordset dialog box.

SQL statement 1

```
SELECT * FROM Employees
```

1 In the tree of database items at the bottom of the dialog box, expand the Tables branch and select the Employees table.

2 Click the Select button.

3 Click OK to add the recordset to the Data Binding inspector.

SQL statement 2

```
SELECT emplNo, emplName
FROM Employees
WHERE emplJob = 'varJob'
ORDER BY emplName
```

1 In the tree of database items, expand the Tables branch, then expand the Employees branch.

2 Build the SQL statement as follows:

- Select emplNo and click the Select button.
- Select emplNameand click the Select button.
- Select emplJob and click the Where button.
- Select emplNameand click the Order By button.

3 Place the insertion point after WHERE emplJob in the SQL text area and type =**'varJob'**

4 Define the variable 'varJob' as follows:

- In the Variables text area, click the plus (+) button and enter the following values in the Name, Default Value, and Run-Time Value columns: **varJob**, **CLERK**, **Request("job")**.

5 Click OK to add the recordset to the Data Binding inspector.

Defining a recordset for a results page

A results page contains a SQL statement that finds the results of a database search. The SQL statement uses search criteria passed to it by a search page on a Web browser. The search criteria are stuffed in variables in the SQL statement, which is then used to create the recordset.

For example, you may want to give your field sales staff the ability to view customers with incomes above a certain level. The sales associate would enter a minimum income level in a form on a search page and send the value to a server by clicking the Submit button. On the server, the value would be passed to the results page's SQL statement, which would then create a recordset containing only customers with incomes above the level specified by the sales associate.

You don't need to know SQL to define the recordset for a results page in UltraDev. This section describes how to define the recordset using the simple Recordset dialog box.

Note: If you have more than one search condition, you must use the advanced Recordset dialog box to define your recordset. The simple Recordset dialog box only supports one search condition.

If you prefer to use SQL to define the recordset for a results page, you can use the advanced Recordset dialog box. For more information, see "Defining a recordset using SQL" on page 80.

To define a recordset for a results page:

1 Define a new recordset or open an existing one.

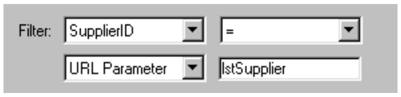
For instructions on defining a new recordset, see "Defining a recordset" on page 77. To open an existing recordset, open the Data Binding inspector (Window > Data Bindings) or the Server Behavior inspector (Window > Server Behaviors) and double-click the name of the recordset in the list.

2 Make sure the simple Recordset dialog box is open. If the advanced Recordset dialog box opens, click the Simple button.

3 Complete the Filter section as follows:

- From the first pop-up menu, select a field in the table to compare against the value sent by the search page.
- From the second pop-up menu, select a conditional expression to compare the selected value in each record against the value sent by the search page.
- If the HTML form on the search page uses the GET method, select URL Parameter from the third pop-up menu. If the HTML form uses the POST method, select Form Variable. If the HTML form method is set to default, it will use the GET method, so select URL Parameter.
- In the fourth text box, enter the exact name of the HTML field providing the value sent to the results page.

For example, suppose you want to create a recordset that includes only products from a specific supplier. Assume you have a field in the table called SupplierID. Also assume the HTML form on your search page uses the GET method and contains a search list box called lstSupplier ("lst" stands for list box). Here's how your Filter section should look:



Filter: SupplierID =
URL Parameter lstSupplier

4 If you want, click Test, enter a test value, and click OK to connect to the database and create an instance of the recordset. The test value simulates the value that would otherwise have been returned from the search page. Click OK to close the recordset.

5 If you're satisfied with the recordset, click OK.

Copying a recordset to another page

You can copy a recordset from one page to another in your site.

To copy a recordset to another page:

1 Select the recordset in either the Data Binding inspector or the Server Behavior inspector.

2 Click the arrow button in the top-right corner of the inspector and choose Copy from the pop-up menu.

3 Open the other page.

4 Click the arrow button in the top-right corner of the Data Binding inspector or the Server Behavior inspector, and choose Paste from the pop-up menu.

Invoking a stored procedure

A recordset can be defined by a stored procedure, which consists of one or more SQL statements residing in a database (as opposed to residing in the source code of your dynamic pages). Stored procedures can return one or more recordsets, though UltraDev only supports stored procedures that return one recordset or no recordsets.

To invoke a stored procedure to define a recordset:

1 Open the page that needs the recordset.

2 In the Data Binding inspector, click the plus (+) button and select Recordset (Query) from the pop-up menu.

3 If the simple Recordset dialog box opens, click the Advanced button to open the advanced Recordset dialog box.

4 In the advanced Recordset dialog box, enter a name for your recordset and select the connection to the database containing your stored procedure.

5 In the tree of database items at the bottom of the dialog box, expand the Stored Procedures branch, select the stored procedure you want, and click the Procedure button.

6 If the stored procedure takes parameters, define their default and run-time values in the Variables area.

7 Click OK.

Creating a stored procedure object

You can create a stored procedure server object, then invoke it in your Web application. The stored procedure object is called a Command in ASP, a Callable in JSP, and a Stored Procedure in ColdFusion.

A stored procedure server object can hold a recordset as well as other data such as output parameters. In contrast, a recordset object defined by a stored procedure, as described in "Invoking a stored procedure" on page 85, can only hold a recordset.

To create a stored procedure server object:

1 Open any dynamic page in UltraDev.

2 In the Data Binding inspector, click the plus (+) button and select one of the following items from the pop-up menu:

- in ASP, select Command (Stored Procedure)
- in JSP, select Callable (Stored Procedure)
- in ColdFusion, select Stored Procedure

3 Enter a name for the stored procedure, then select a connection from the Connections pop-up menu to specify the database containing the stored procedure.

4 If you're using ASP, select Stored Procedure in the Type pop-up menu.

5 Click the Return Recordset Named option and enter a name for the recordset to be returned.

6 Select a stored procedure from the tree of database items at the bottom of the dialog box.

7 Click OK.

The stored procedure object is added to the list of data sources in the Data Binding inspector.

Editing or deleting a recordset

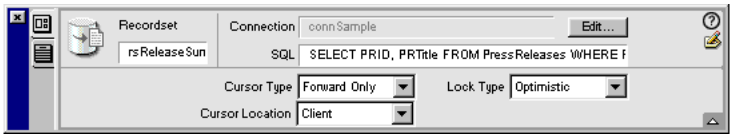
You can edit or delete any recordset on your page.

To edit a recordset:

1 In either the Data Binding inspector or the Server Behavior inspector, double-click the name of the recordset you want to edit.

2 In the simple or advanced Recordset dialog box, make your changes, then click OK.

You can also use the Property inspector to edit your recordsets. Open the Property inspector (Window > Properties), then select the recordset in the Server Behavior inspector (Window > Server Behavior). Here's the Property inspector for a recordset:


Recordset rsReleaseSun
Connection connSample
SQL SELECT PRID, PRTitle FROM PressReleases WHERE F
Cursor Type Forward Only
Lock Type Optimistic
Cursor Location Client

If you edit a recordset in the Live Data window with the Auto Refresh option not selected, you must refresh the page to see your changes. To refresh the page, click the Refresh button or choose View > Refresh Live Data.

To delete a recordset:

1 In either the Data Binding inspector or the Server Behavior inspector, select the recordset you want to delete.

2 Click the minus (-) button.

Writing simple SQL queries

You can write your own SQL query to define a recordset, then cut and paste the query into your pages. This section presents a short primer on writing simple SQL queries to create recordsets.

The most common SQL statement for creating a recordset is the SELECT statement, which extracts specified fields (columns) from one or more tables to build the recordset. Here's the basic syntax for a SELECT statement:

```
SELECT FieldName FROM TableName
```

You can also add line breaks, tabs, and other white space to your statements to clarify the logic: SQL ignores all white space. For example, the following is a valid statement:

```
SELECT PaidDues
    FROM Members
```

Including an entire table

To include the full contents of a table in your recordset, use the asterisk (*) as a wildcard character to include all the fields in the table. For example, suppose you have a table called Customers. To extract all the fields, you would type the following SELECT statement:

```
SELECT * FROM Customers
```

Limiting the number of fields

Suppose you only need the data contained in two fields of the Customers table: the YearBorn field and the DateLastPurchase field. To create a recordset containing only the data from these two fields, you would type the following SELECT statement:

```
SELECT YearBorn, DateLastPurchase FROM Customers
```

Limiting the number of records

Use a WHERE clause to limit the number of records in the recordset. For example, you may want to include only those customers who earn more than $50,000 a year. Assume you have a field in your table called Earnings that tells you how much each customer earns. Your SELECT statement would read as follows:

```
SELECT YearBorn, DateLastPurchase FROM Customers
WHERE Earnings > 50000
```

Here's a list of conditional operators you can use in a WHERE clause:

Operator	Meaning
=	Equal to (case sensitive)
LIKE	Equal to (case insensitive)
<>	Not equal to (case sensitive)
NOT LIKE	Not equal to (case insensitive)
<	Less than
>	Greater than
<=	Less than or equal to
>=	Greater than or equal to

If the item being compared is text, place it in single quotes as in this example:

```
...WHERE Country = 'Germany'
```

If the item being compared is a date, and you're working with a Microsoft Access database, enclose it with # signs:

```
...WHERE DateOfBirth < #01/01/1970#
```

Other databases have their own date conventions. Consult the database vendor's documentation.

You can use wildcard characters in conditional expressions. The standard wildcard character is the percentage sign (%):

```
...WHERE LastName LIKE 'Mc%'
```

For an Access database, the asterisk (*) also works as a wildcard character:

```
...WHERE CompanyName = '*soft'
```

Sorting the records

Use the ORDER BY clause to sort the records in your recordset. For example, suppose you want to sort the records in the recordset by customer earnings, from the lowest to the highest. In the SQL statement you would order the records as follows:

```
SELECT LastName, FirstName, Earnings FROM Customers
ORDER BY Earnings
```

By default, the ORDER BY clause sorts records in ascending order (1, 2, 3... or A, B, C...). If you want to sort them in descending order, from the highest earnings to the lowest, you would use the DESC keyword as follows:

```
...ORDER BY Earnings DESC
```

6

CHAPTER 6
Adding Dynamic Content

Adding dynamic content is a three-step process. First, define at least one data source to provide the dynamic content. Data sources can include a field in a recordset, a value submitted by an HTML form, the value contained in a server object, and other data.

Second, specify where you want to put the dynamic content. In UltraDev, you can put dynamic content almost anywhere in the page or its HTML source code:

- You can place it at the insertion point.
- You can replace a text string with it.
- You can insert it in an HTML attribute. For example, dynamic content can define the SRC attribute of an image, or the VALUE attribute of a form field.

Third, add the dynamic content by choosing one of the data sources you defined. UltraDev inserts a server-side script in the page instructing the server to transfer the data from the data source to the page's HTML source code.

There is often more than one way to make a given page element dynamic. For example, to make an image dynamic you can use the Data Binding inspector, the Property inspector, or the Image command in the Insert menu. This chapter describes the most efficient ways of making various page elements dynamic.

By default, an HTML page can display only one record at a time. To display the other records in the recordset, you can add a link to move through the records one at a time (see "Creating record navigation links" on page 111), or you can create a repeated region to display more than one record on a single page (see "Displaying more than one record" on page 108).

After adding dynamic content to a page, you can make changes to it. For more information, see "Making basic changes to pages" on page 128.

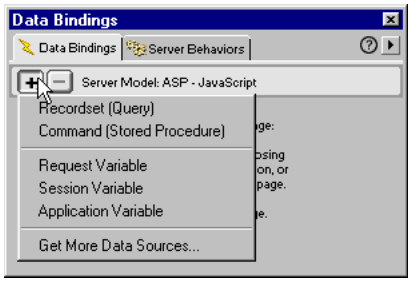
Defining data sources

Before you can add dynamic content to your pages, you must define at least one data source to provide the dynamic content. A data source can be a field in a recordset, a value submitted by an HTML form, or a server object such as a session or application object.

To define a data source:

1. Open the Data Binding inspector by choosing Window > Data Bindings.
2. Click the plus (+) button and select a type of data source from the pop-up menu.

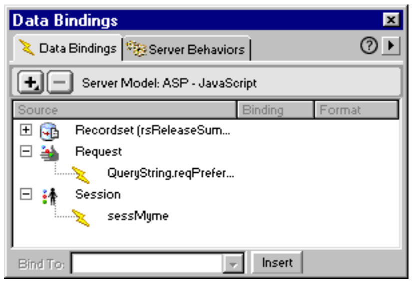
   For example, here are your choices in ASP:



3. Define the data source you selected.

- If you select Recordset or Stored Procedure from the pop-up menu, see "Defining a recordset" on page 77 for more information. (Stored Procedure is also called Command or Callable, depending on your server model.)
- If you select another data source such as Session Variable from the pop-up menu, enter the name of the variable to be used by the associated server object and click OK. For example, suppose you want to use the Session object to store the visitor's first name in order to display it on succeeding pages. In the Session dialog box, enter a variable name such as FirstName and click OK. (This action is the same as writing Session("FirstName")in the source code.)
- If you're an ASP user and you selected the Request Variable, select the type of Request variable (QueryString, Form, Cookie, and so on), enter the name of the variable to be used by the Request object, and click OK. For more information, see "About server objects" on page 94.

The newly defined data source appears in the inspector's data sources list:



To delete a data source from your list in the Data Binding inspector:

1 In the Data Binding inspector, select the data source from the list.

2 Click the minus (-) button.

Note: Deleting a data source from the inspector will not delete the corresponding dynamic content on your page. For instructions on deleting dynamic content on a page, see "Replacing or removing dynamic content" on page 129.

About server objects

Server objects can be a source of data for your applications. Common server objects include the Request, Session, and Application objects.

- Request objects (ASP, JSP) store information the browser submits to the server. In ASP, the Request object has five collections: Request.QueryString, Request.Form, Request.ServerVariables, Request.Cookie, and Request.ClientCertificates. Here are two examples. Request.Form("LastName") contains the value a user entered on a browser—specifically, the value in the form field called LastName in an HTML form using the POST method. Request.ServerVariables("HTTP_USER_AGENT") contains information about the user's browser, such as Mozilla/4.07 [en] (WinNT; I)
- Session objects (ASP, JSP, ColdFusion) store information about a user's current session. For example, a Session object can keep track of the user's name, which allows you to personalize subsequent pages requested by the same user. A different Session object is created for each user and is maintained for a set period of time or until the object is explicitly ended.
- The Application object (ASP, ColdFusion) stores information available application-wide. Only one Application object is created for the application. (An application is defined as all the files in a virtual directory and its subdirectories.)

For more information on these and other server objects, consult your server technology's documentation or visit the following Web sites:

- For ASP documentation, visit Microsoft's server technology Web site at http://msdn.microsoft.com/workshop/server/toc.htm
- For JSP documentation, visit Sun's JSP Web site at http://java.sun.com/products/jsp/docs.html
- For ColdFusion documentation, visit Allaire's ColdFusion Web site at http://www.allaire.com/Documents/cf4docs.cfm

Making text dynamic

You can replace existing text with dynamic text, or you can place dynamic text at a given insertion point on the page.

Dynamic text adopts any text formatting applied to the existing text or to the insertion point. For example, if a CSS style affects the selected text, the dynamic content replacing it is also be affected by the style. You can also add or change the text format of dynamic content by using any of Dreamweaver's text formatting tools.

You can also apply a data format to dynamic text. For example, if your data consists of dates, you can specify a particular date format such as 04/17/00 for U.S. visitors, or 17/04/00 for Canadian visitors.

To make text dynamic:

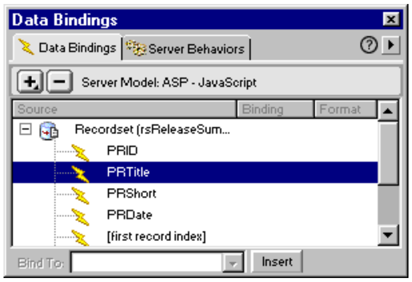
1 Open the Data Binding inspector by choosing Window > Data Bindings.

2 Make sure the Data Binding inspector lists the data source you want to use.

   The data source should contain plain text (ASCII text). For example, the data source could contain full-fledged HTML, which is plain text. If no data sources appear in the list, or if the available data sources don't meet your needs, click the plus (+) button to define a new data source. See "Defining data sources" on page 92.

3 In the Document or Live Data window, select text on the page, or click where you want to add dynamic text.

4 In the Data Binding inspector, select a data source from your list. If you select a recordset, specify the field you want in the recordset.



5 Click Insert, or drag and drop the data source onto the page.

   The dynamic content appears on the page if you're working in the Live Data window. In the Document window, a placeholder appears instead. (If you selected text on the page, the placeholder replaces the text selection.) The placeholder for a recordset data source uses the syntax {RecordsetName.ColumnName} where Recordset is the name of the recordset and ColumnName is the name of the field you chose from the recordset.

If desired, specify a data format for the dynamic text. For example, if the price data in a record reads 10.989, you can display the price on your page as $10.99 by selecting the Currency - 2 Decimal Places format in the pop-up menu. This format takes a number and displays it with two decimal places. If the number has more than two decimal places, the data format rounds the number to the closest decimal; if the number has no decimal places, the data format adds a decimal point and two zeros.

To apply a data format to dynamic text:

1 Select the dynamic content (Live Data window) or its placeholder (Document window) on the page.

2 In the Data Binding inspector (Window > Data Bindings), click the arrow button in the Format column.

3 Select a format from the pop-up menu.

   Make sure the data format is appropriate for the data. For example, the Currency formats work only if the dynamic text consists of numbers. Also, you cannot apply more than one format to the same data.

To edit existing data formats or create one of your own, see "Editing and creating data formats" on page 136.

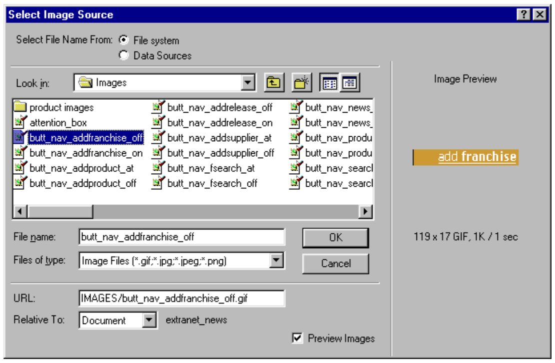
Making images dynamic

You can make images on your page dynamic. For example, suppose you design a page to display items for sale at a charity auction. Each page would include descriptive text and a photo of one item. The page's general layout would remain the same for each item, but the photo (and descriptive text) could change.

To make an image dynamic:

1 Place the insertion point where you want the image to appear on the page, then select Insert > Image.

The Select Image Source dialog box appears.



Select File Name From: (●) File system ( ) Data Sources

Look in: Images

product images, attention_box, butt_nav_addfranchise_off, butt_nav_addfranchise_on, butt_nav_addproduct_at, butt_nav_addproduct_off, butt_nav_addrelease_off, butt_nav_addrelease_on, butt_nav_addsupplier_at, butt_nav_addsupplier_off, butt_nav_fsearch_at, butt_nav_fsearch_off, butt_nav_news_, butt_nav_news_, butt_nav_produ, butt_nav_produ, butt_nav_searcl, butt_nav_searcl

File name: butt_nav_addfranchise_off

Files of type: Image Files (*.gif;*.jpg;*.jpeg;*.png)

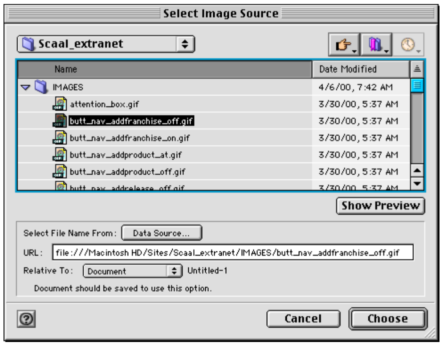
URL: IMAGES/butt_nav_addfranchise_off.gif

Relative To: Document extranet_news

☑ Preview Images

119 x 17 GIF, 1K / 1 sec

On the Macintosh, the dialog box differs slightly:



Scaal_extranet

Name	Date Modified
IMAGES	4/6/00, 7:42 AM
attention_box.gif	3/30/00, 5:37 AM
butt_nav_addfranchise_off.gif	3/30/00, 5:37 AM
butt_nav_addfranchise_on.gif	3/30/00, 5:37 AM
butt_nav_addproduct_at.gif	3/30/00, 5:37 AM
butt_nav_addproduct_off.gif	3/30/00, 5:37 AM
butt_nav_addrelease_off.gif	3/30/00, 5:37 AM

Select File Name From: Data Source...

URL: file:///Macintosh HD/Sites/Scaal_extranet/IMAGES/butt_nav_addfranchise_off.gif

Relative To: Document Untitled-1

Document should be saved to use this option.

2 Click the Data Sources option (Windows) or the Data Source button (Macintosh).

A list of data sources appears.

3 Select a data source from the list.

The data source should be a recordset containing the paths to your image files. Depending on the file structure of your site, the paths can be absolute, document relative, or root relative. For more information, see About document locations and paths in the “Linking and Navigating” chapter of the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

Note: UltraDev does not currently support binary images stored in a database.

If no recordsets appear in the list, or if the available recordsets don’t meet your needs, click the plus (+) button to define a new recordset. For instructions, see “Defining a recordset” on page 77.

4 Click OK.

Making form objects dynamic

You can create an HTML form for displaying records in your database. For example, you could design a form to display suppliers’ contact information.

You can only display one record at a time in a form. To give users the ability to view other records, you can add links to move through the records one at a time. (See “Creating record navigation links” on page 111.)

The most common dynamic form objects are text fields, image fields, checkboxes, and radio buttons. You can also use a data source to populate the options in a List Menu object.

To make text fields dynamic:

1 Open the Data Binding inspector by choosing Window > Data Bindings.

2 Make sure the Data Binding inspector lists the data source you want to use.

The data source should contain textual information. If no data sources appear in the list, or if the available data sources don’t meet your needs, click the plus (+) button to define a new data source. For instructions, see “Defining data sources” on page 92.

3 In the Document window or Live Data window, select a text field in your HTML form.

4 In the Data Binding inspector, select a data source from your list of data sources.

5 In the Bind To text box, make sure the VALUE attribute (input.value) is selected.

6 Click Bind.

To make image fields dynamic:

1 Place the insertion point where you want the image field to appear on the page, then select Insert > Image.

   The Select Image Source dialog box appears.

2 Click the Data Sources option at the top of the dialog box.

   A list of data sources appears.

3 Select a data source from the list.

   The data source should be a recordset containing the paths to your image files. Depending on the file structure of your site, the paths can be absolute, document relative, or root relative. For more information, see About document locations and paths in the “Linking and Navigating” chapter of the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

   Note: UltraDev does not currently support binary images stored in a database.

   If no recordsets appear in the list, or if the available recordsets don’t meet your needs, click the plus (+) button to define a new recordset. For instructions, see “Defining a recordset” on page 77.

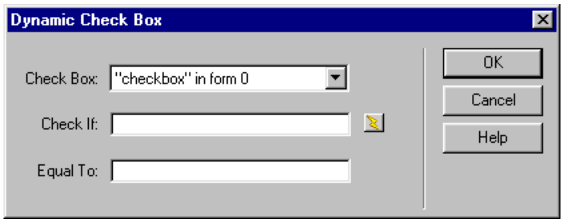
4 Click OK.

To make checkboxes dynamic:

1 Select a checkbox in the HTML form on your page.

2 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and select Dynamic Elements > Dynamic Check Box from the pop-up menu.

The Dynamic Check Box dialog box appears.



Check Box: "checkbox" in form 0

Check If:

Equal To:

3 If you want the checkbox to be selected when a field in a record equals a certain value, do the following:

- Click the lightning bolt icon beside the Check If text box and select the field from the list of data sources.

  Typically, the chosen field contains Boolean data such as Yes and No, or TRUE and FALSE.

- In the Equal To text box, enter the value the field must have for the checkbox to appear selected.

  For example, if you want the checkbox to appear selected when a specific field in a record equals Yes, enter Yes in the Equal To text box.

  Note: This value will also be returned to the server if the user clicks the form's Submit button.

4 Click OK.

The checkbox will appear selected or unselected—depending on the data—when the form is viewed in a browser.

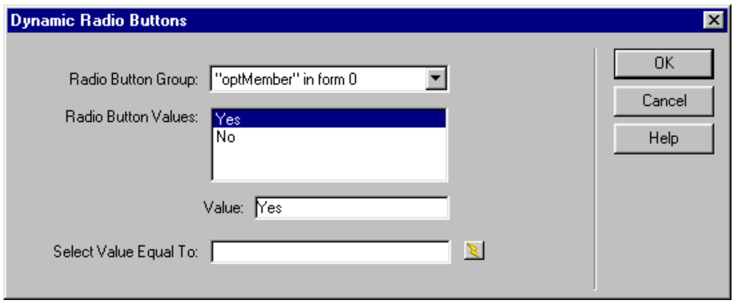
To make radio buttons dynamic:

1 Make sure the page has at least one group of radio buttons.

  You create a group of radio buttons by giving all the radio buttons in a group the same name. For more information, see Creating Forms in the *Using Dreamweaver* book or in Dreamweaver Help (Help > Using Dreamweaver).

2 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and select Dynamic Elements > Dynamic Radio Button from the pop-up menu.

  The Dynamic Radio Buttons dialog box appears.


Dynamic Radio Buttons

Radio Button Group: "optMember" in form 0

Radio Button Values: Yes, No

Value: Yes

Select Value Equal To:

3 In the Radio Button Group pop-up menu, select a group of radio buttons on your page.

4 You can specify the value of each radio button in the group by selecting a radio button in the Radio Button Values list, then entering a value for the radio button in the Value text box.

  Note: The value of the currently selected radio button will be returned to the server if the user clicks the form's Submit button.

5 If you want one of the radio buttons to appear selected when a field in a record equals that radio button's value, click the lightning bolt icon beside the Select Value Equal To text box and select the field from the list of data sources.

  The chosen field should contain data matching the radio buttons' values—that is, the values appearing in the Radio Button Values list.

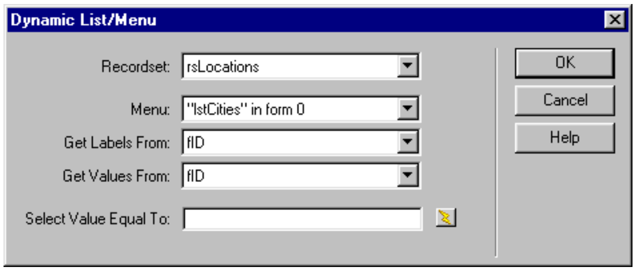
6 Click OK.

To make a List/Menu object dynamic:

1 Select the List/Menu object in the HTML form on your page.

2 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and select Dynamic Elements > Dynamic Menu from the pop-up menu.

The Dynamic Menu dialog box appears.



Recordset: rsLocations
Menu: "lstCities" in form 0
Get Labels From: fID
Get Values From: fID
Select Value Equal To:

3 In the Recordset pop-up menu, select the recordset containing the menu information.

4 In the Get Labels From pop-up menu, select the field containing the labels for the menu items.

5 In the Get Values From pop-up menu, select the field containing the values of the menu items.

6 If you want a particular menu item to be selected when the page opens in a browser, enter a value equal to the menu item's value in the Select Value Equal To text box.

7 Click OK.

Making HTML attributes dynamic

You can dynamically change the appearance of your page by binding HTML attributes to data. For example, you could change the background image of a table by binding the table's BACKGROUND attribute to a field in a recordset.

You can bind HTML attributes with the Data Binding inspector or with the Property inspector.

To make HTML attributes dynamic with the Data Binding inspector:

1 Open the Data Binding inspector by choosing Window > Data Bindings.

2 Make sure the Data Binding inspector lists the data source you want to use.

The data source should contain data that's appropriate for the HTML attribute you want to bind. If no data sources appear in the list, or if the available data sources don't meet your needs, click the plus (+) button to define a new data source. For instructions, see "Defining data sources" on page 92.

3 In the Document window or the Live Data window, select an HTML object.

4 In the Data Binding inspector, select a data source from your list.

5 In the Bind To text box, select an HTML attribute to bind to the data source.

6 Click Bind.

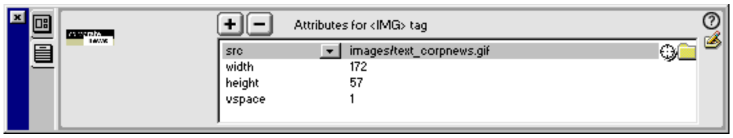
To make HTML attributes dynamic with the Property inspector:

1 In the Document or Live Data window, select an HTML object and open the Property inspector (Window > Properties).

2 If the attribute you want to bind has a folder icon next to it in the inspector's Standard view, click the folder icon to open a file selection dialog box, then click the Data Sources option to display a list of data sources. Skip to step 6.

3 If the attribute you want to bind does not have a folder icon next to it in the Standard view, click the List tab (the lower of the two tabs) on the left side of the inspector.

The Property inspector's List view appears.



src	images/text_corpnews.gif
width	172
height	57
vspace	1

4 If the attribute you want to bind is not listed in the List view, click the plus (+) button, then enter the attribute's name or click the small arrow button and select the attribute from the pop-up menu.

5 To make the attribute's value dynamic, click the attribute then click the lightning-bolt icon or folder icon at the end of the attribute's row.

If you clicked the lightning-bolt icon, a list of data sources appears.

If you clicked the folder icon, a file selection dialog box appears. Click the Data Sources option to display a list of data sources.

6 Select a data source from the list of data sources.

The data source should hold data that's appropriate for the HTML attribute you want to bind. If no data sources appear in the list, or if the available data sources don't meet your needs, click the plus (+) button to define a new data source. For instructions, see "Defining data sources" on page 92.

7 Click OK.

Making ActiveX, Flash, and other object parameters dynamic

You can make the parameters of Java applets and plug-ins dynamic, as well as the parameters of ActiveX, Flash, Shockwave, Director, and Generator objects.

Before starting, make sure the fields in your recordset hold data that's appropriate for the object parameters you want to bind.

To make object parameters dynamic:

1 In the Document window, select an object and open the Property inspector (Window > Properties).

2 Click the Parameters button.

The Parameters dialog box appears.

3 If your parameter does not appear in the list, click the plus (+) button and enter a parameter name in the Parameter column.

4 Click the parameter's Value column, then click the lightning-bolt icon to specify a dynamic value.

A list of data sources appears.

5 Select a data source from the list.

The data source should hold data that's appropriate for the object parameter you want to bind. If no data sources appear in the list, or if the available data sources don't meet your needs, click the plus (+) button to define a new data source. For instructions, see "Defining data sources" on page 92.

6 Click OK.

CHAPTER 7
Activating Dynamic Pages

The next step in the process of creating a dynamic page is supplying the page with the "intelligence" needed to make it work as intended. In many cases, you supply this intelligence by adding server behaviors to the page. A server behavior is a VBScript, JavaScript, Java, or ColdFusion script that runs on a server instead of in a browser.

For example, after creating an HTML form to insert records in a database, you must add a server behavior to do the actual work of taking data entered in the form and inserting it in the database.

UltraDev comes with a number of predefined server behaviors to activate your pages. You can also write your own server behaviors or install server behaviors written by other people.

The server behaviors you need to make a page work will depend on the kind of page you're building. For a list of common pages and the HTML, dynamic content, and server behaviors needed to make each of them work, see "Page Blueprints" on page 71.

To modify server behaviors after adding them to a page, see "Making changes to server behaviors" on page 129.

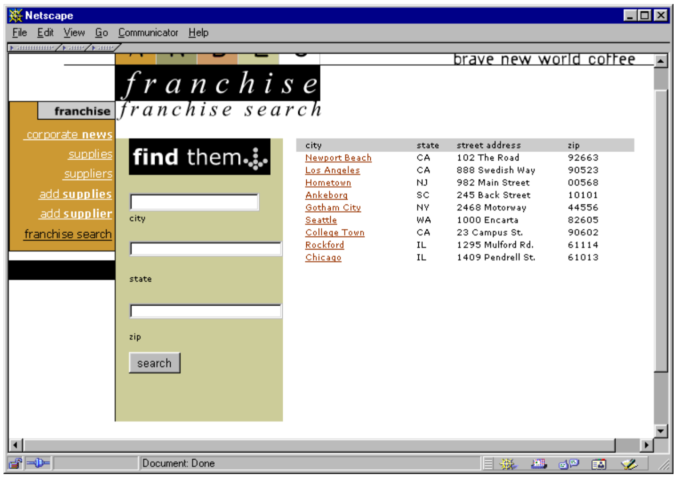
Displaying multiple records

The Repeat Region server behavior lets you display more than one record per page. As well, the Data Binding inspector provides recordset statistics you can use to build a record counter on any page displaying multiple records.

Displaying more than one record

To display more than one record on a single page, you must designate a selection containing dynamic content as a repeated region. Any selection can be turned into a repeated region; the most common are a table, a table row, or a series of rows.

For example, you could design a table to display all the franchises of the Scaal Coffee company. Each row in the table would display a different franchise, and each column would display a different piece of information about the franchises:



city	state	street address	zip
Newport Beach	CA	102 The Road	92663
Los Angeles	CA	888 Swedish Way	90523
Hometown	NJ	982 Main Street	00568
Ankeborg	SC	245 Back Street	10101
Gotham City	NY	2468 Motorway	44556
Seattle	WA	1000 Encarta	82605
College Town	CA	23 Campus St.	90602
Rockford	IL	1295 Mulford Rd.	61114
Chicago	IL	1409 Pendrell St.	61013

You build this kind of table by applying the Repeat Region server behavior to a table row containing dynamic content. At design time, the repeated region consists of a single row. At run time, the row is repeated a number of times and a different record is inserted in each new row.

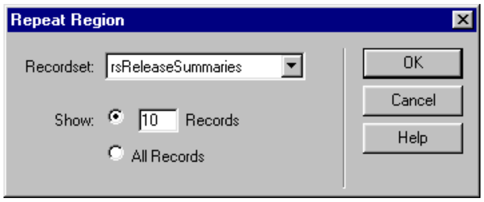
To create a repeated region:

1 Select a region that contains dynamic content.

  The selection can be anything, including a table, a table row, or even a paragraph of text.

2 Open the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button, and select Repeat Region.

  The Repeat Region dialog box appears.



Recordset: rsReleaseSummaries

Show: [✓ 10 Records] All Records

3 Specify the recordset containing the data to populate the repeated region.

4 Specify the number of records to display per page.

  If you specify a limited number of records per page and it's possible the number of records requested will exceed it, add record navigation links to let users display the other records. See "Creating record navigation links" on page 111.

5 Click OK.

In the Document window, a thin, tabbed gray outline appears around the repeated region. In the Live Data window (View > Live Data), the gray outline disappears and the selection expands to display the number of records you specified.

Building a record counter

Use the Data Binding inspector to build a record counter such as “Displaying Records 1 - 8 of 31.” Record counters are especially useful in results pages that may display many records.

The following procedure builds a typical counter. You can use the same tools to build other types of counters.

To build a record counter on a results page:

1. Type the counter’s text as follows:

   Showing records - of

2. Place the insertion point at the end of the text string.
3. Open the Data Binding inspector (Window > Data Bindings), expand the branch of the recordset to be monitored, select [total records]from the list of data sources, and click Insert. You can also drag [total records]onto the page.

   Here’s how the record counter should look in the Document window:

   Showing records - of {*myRecordset*_total}

4. Place the insertion point after the word *records*.
5. In the Data Binding inspector, select [first record index]from the list of data sources and click Insert. You can also drag [first record index]onto the page.

   Here’s how the counter should look:

   Showing records {*myRecordset*_first} - of {*myRecordset*_total}

6. Place the insertion point after the hyphen.
7. In the Data Binding inspector, select [last record index]from the list of data sources and click Insert. You can also drag [last record index]onto the page.

   Here’s how the counter should look:

   Showing records {*myRecordset*_first} - {*myRecordset*_last} of {*myRecordset*_total}

If you view the page in the Live Data window (View > Live Data), the counter should read something like this:

Showing records 1- 5 of 16

If the results page has a navigation link to move to the next records, clicking the link would display the next five records in the recordset and the counter would read as follows:

Showing records 6 - 10 of 16

Note: Links don’t work in the Live Data window. To test them, you can use UltraDev’s Preview in Browser feature. Make sure the Preview Using Live Data Server option selected in Preferences (Edit > Preferences > Preview in Browser), then select File > Preview in Browser.

Moving through records

You can give users the ability to move through records in a recordset by attaching the appropriate server behaviors to a link. For example, after designing a page to display five records at a time, you might want to add links such as “Next Records” or “Previous Records” to let users see the next or previous five records.

UltraDev lets you create four types of navigation links to move through records: First, Previous, Next, Last. A single page can contain any number of these links, provided they all work on a single recordset. In other words, you can’t add links to move through a second recordset on the same page.

UltraDev has a special server behavior for detail pages that moves to a record specified in a master page. See “Moving to a specific record in a detail page” on page 112.

Creating record navigation links

You can create navigation links to move to the first record, the last record, the next record (or set of records), and the previous record (or set of records) in a recordset.

To create a link to move through records:

1. On the page, select the text string or image to act as your link.

   You don’t have to create a link for the text string or image: UltraDev will create one for you.

2. Open the Server Behavior inspector (Window > Server Behaviors) and click the plus (+) button.
3. Choose Move to Record from the pop-up menu, then choose one of the listed server behaviors.

   Note: If the recordset contains a large number of records, the Move to Last Record server behavior can take a long time to run.

4. In the Recordset pop-up menu, select the recordset containing the records.
5. Click OK.

Moving to a specific record in a detail page

Your application may contain a detail page to display more information about a record listed on a master page (usually a results page). UltraDev provides a a server behavior that moves to the record identified by the master page. For more information on the server behavior that sends information from the master page, see "Going to a detail page" on page 116.

Often a detail page simply displays information. A detail page can also be used to update or delete records. For more information, see "Updating a record in a database" on page 119 and "Deleting a record in a database" on page 122.

To move to a specific record in a detail page:

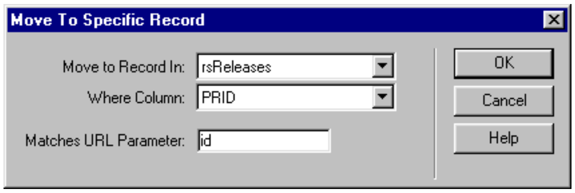
1 Open the detail page in the Document window.

   Make sure the page contains dynamic content displaying the detail information.

2 Open the Server Behavior inspector (Window > Server Behaviors) and click the plus (+) button.

3 Choose Move to Record from the pop-up menu, then choose Move to Specific Record.

   The Move to Specific Record dialog box appears.



Move to Record In: rsReleases
Where Column: PRID
Matches URL Parameter: id

4 In the Move to Record In pop-up menu, select the recordset containing the records to be displayed.

5 In the Where Column pop-up menu, select the field that contains the value passed by the master page.

   For example, if the master page passes a record ID number, choose the field containing record ID numbers.

6 In the Matches URL Parameter text box, enter the name of the value passed by the master page.

   The name appears in the Go to Detail Page server behavior on the master page.

7 Click OK.

Hiding regions

UltraDev has a server behavior that lets you hide a region such as a record navigation link from users when the region is not needed. For example, after adding "Previous records" and "Next records" links to a results page, you can specify that the "Previous records" link should be hidden on the first page of results and the "Next records" link should be hidden on the last page. Or if a query returns an empty recordset, you can hide the result list.

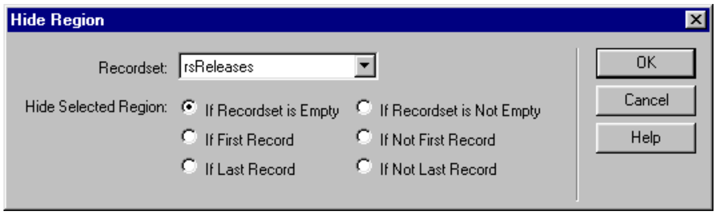
To hide a region when it's not needed:

1 Select the region.

2 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button.

3 Choose Hide Region form the pop-up menu.

The Hide Region dialog box appears.


Hide Region

Recordset: rsReleases

Hide Selected Region: [✓ If Recordset is Empty] If Recordset is Not Empty If First Record If Not First Record If Last Record If Not Last Record

4 In the Hide Region dialog box, choose your recordset, then select a condition specifying when to hide the region.

5 Click OK.

Passing information between pages

You can pass information, or “parameters,” from one page to another in your application. This ability is useful when you want one page to tell another what record to display, or when you want to preserve information from one page to the next and thus avoid having the user repeatedly enter the same information (search criteria, for example).

UltraDev provides two server behaviors for this purpose: Go to Detail Page, and Go to Related Page.

Parameters are gathered in an HTML form and passed to the server in one of two ways:

- If the form uses the POST method to send information to the server, the parameters are sent as part of the body of the message.
- If the form uses to GET method, the parameters are appended to the URL specified in the form’s Action attribute.

If the form uses the GET method, a question mark is appended to the URL to distinguish the address information from the parameter information, and the parameter information follows the question mark. For example, the following URL opens and passes information to a detail page:

```
http://www.mysite.com/search_details.asp?model_number=3701
```

The first part of the URL, http://www.mysite.com/search_details.asp, opens the detail page. The second part, ?model_number=3701, tells the detail page what record to find and display. The string model_number is the name of the field object on the HTML form, and 3701 is the value the user entered in the field.

Gathering data in an HTML form

Web applications make extensive use of HTML forms to gather data from users. Make sure your forms have a Submit button to submit the form data to your server.

If you don't know how to add a form to a page, see Chapter 15, Creating Forms, in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

To configure a form to gather information from users:

1. After adding a form to the page, select the form by clicking the `<form>` tag on the status bar in the lower left corner of the Document window.
2. Open the Property inspector (Window > Properties).

   The form's properties are displayed.
3. Type a name in the Form Name text box.
4. In the Action text box, specify the ASP, JSP, or ColdFusion page that will process the submitted form data on the server. Click the folder icon to locate the file, or enter the file's path.
5. Choose one of the following methods to define how the form sends data to the server:

- `Get` sends the form data by appending it to the URL. Because URLs are limited to 8192 characters, don't use the `GET` method with long forms.
- `Post` sends the form data in the body of a message.
- `Default` uses the browser's default method (usually `GET`).

Going to a detail page

In your Web application, you can have a master page (such as a results page) listing records that, when one record is clicked, opens a detail page displaying more information about the record. Using a standard link on the master page does not work: the link must also tell the detail page what record the user selected.

The Go to Detail Page server behavior creates a special link that passes information from a master page to a detail page.

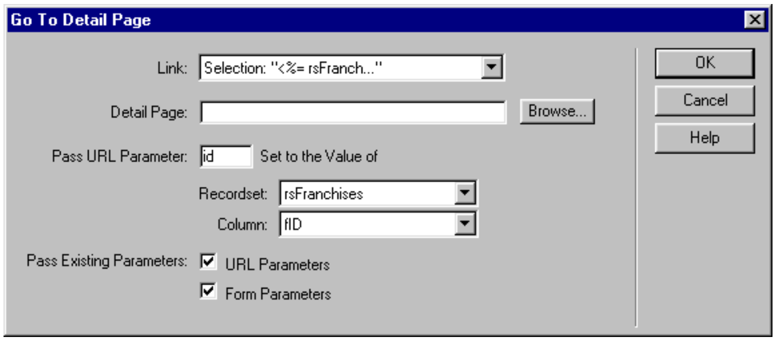
To create a link that passes information from a master page to a detail page:

1 On the master page, select the dynamic content to double as a link.

  The dynamic content is normally a data placeholder in a repeated region. You don't have to place a link on the dynamic content: UltraDev will create the link automatically.

2 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and choose Go to Detail Page from the pop-up menu.

  The Go to Detail Page dialog box appears.


Go To Detail Page

Link: Selection: "<%= rsFranch..."

Detail Page:

Pass URL Parameter: id Set to the Value of

Recordset: rsFranchises

Column: fID

Pass Existing Parameters: ☑ URL Parameters ☑ Form Parameters

3 In the Detail Page text box, click Browse and locate the detail page file.

  If the current page submits data to itself, enter the current page's file name.

4 Specify what information you want to pass to the detail page by selecting a recordset and a column from the Recordset and Column pop-up menus.

  Typically, the information is a record's ID.

5 If the master page contains an HTML form that submits data to the server using the POST method, you can also pass this data to the detail page by making sure the Form Parameters option is selected.

6 Click OK.

Going to a related page

You can create a link that opens a page other than a detail page and passes certain information to that page. For example, you can pass search criteria from one page to another to save the user from entering the search criteria again.

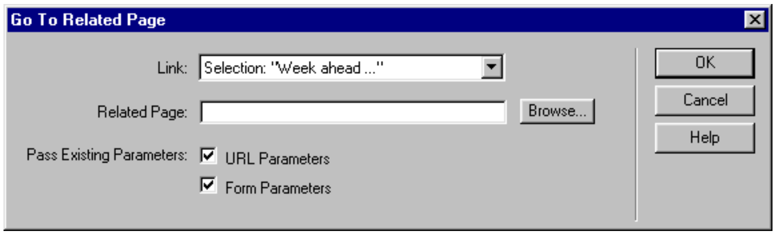
To create a link that passes information to a related page:

1 On the page, select the text string or images to act as a link to the related page.

You don't have to create a link for the text string or image: UltraDev will create one automatically.

2 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and choose Go to Related Page from the pop-up menu.

The Go to Related Page dialog box appears.



Link: Selection: "Week ahead ..."

Related Page:

Pass Existing Parameters: ☑ URL Parameters ☑ Form Parameters

3 In the Related Page text box, click Browse and locate the related page file.

If the current page submits data to itself, enter the current page's file name.

4 If the page contains an HTML form that submits data using the `POST` method, you can also pass this data to the related page by making sure the Form Parameters option is selected.

5 Click OK.

Editing Records

UltraDev comes with a set of server behaviors that let users update, delete, and add records in a database from their browsers.

Inserting a record in a database

Your application can contain a page that allows users to insert records in a database.

To activate an HTML form to insert records in a database:

1 Open the insert page in the Document window.

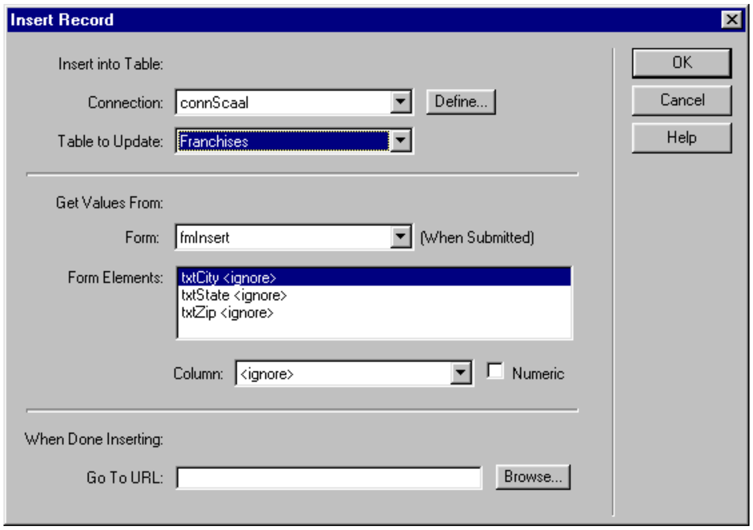
The page should contain an HTML form that has a Submit button.

2 Make sure the HTML form has a name.

Select the `<form>` tag on the status bar in the lower left corner of the window to select the form, open the Property inspector (Window > Properties), and enter a name in the Form Name text box.

3 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and choose Insert Record from the pop-up menu.

The Insert Record dialog box appears.



Connection: connScaal
Table to Update: Franchises
Form: fmInsert (When Submitted)
Form Elements: txtCity <ignore> txtState <ignore> txtZip <ignore>
Column: <ignore> Numeric
Go To URL:

4 Use the Connection and Table to Update pop-up menus to specify the database table into which the record should be inserted.

5 In the Form pop-up menu, specify the HTML form used to enter the data.

6 Specify what each form element will update in the database table by selecting a form element, then selecting a field from the Column pop-up menu. Select the Numeric option if the field accepts only numeric values.

7 In the Go to URL text box, enter the page to open after the record has been inserted into the table.

8 Click OK.

Updating a record in a database

Your application can contain a page that lets users update records in a database. This kind of page is usually a detail page working in tandem with a master page. The master page lets the user choose a record to update, then passes the choice to the detail page (see “Going to a detail page” on page 116).

A page that updates records performs two distinct operations. First, it displays the existing data so it can be modified by the user. Second, it updates the database with the modified data.

To display the existing data so it can be modified by the user:

1 On the update page, create an HTML form to display the data.

Make sure the HTML form has a name and a Submit button. Select the <form> tag on the status bar in the lower left corner of the window to select the form, then open the Property inspector (Window > Properties) and adjust the form's properties as required. You can also relabel the Submit button, if you like.

Next, you'll create a recordset to hold the record to be displayed.

2 In the Data Binding inspector (Window > Data Binding), click the plus (+) button and choose Recordset (Query).

If the advanced Recordset dialog box appears, click the Simple button to open the simple Recordset dialog box.

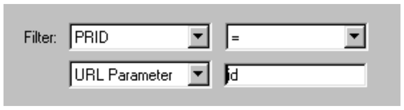
3 Name the recordset and specify where the data you want to display is located using the Connection and Table pop-up menus.

4 Click the Selected option and select a key column (usually the record ID field) and the columns (fields) containing the data to be displayed.

The columns should have a corresponding form object on your page allowing users to view and modify the data.

5 Configure the Filter area so that value of your key column equals the value of the corresponding URL parameter passed by the master page.

This kind of filter creates a recordset that contains only the record—the one specified by the master page. For example, if your key column contains record ID information and is called PRID, and if the master page passes the corresponding record ID information in the URL parameter called *id*, then here's how your Filter area should look:



Filter: PRID =
URL Parameter id

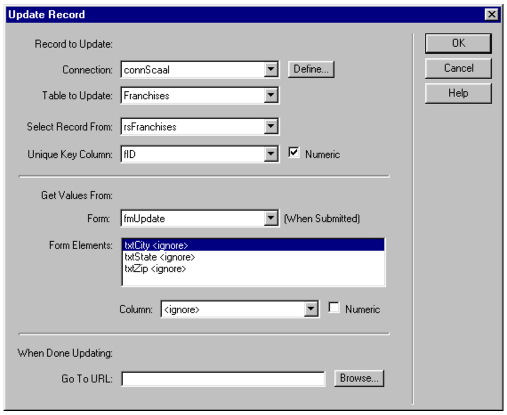
6 Bind each HTML form object to its corresponding recordset field.

For instructions, see "Making form objects dynamic" on page 98.

To update the database with the modified information:

1 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and choose Update Record from the pop-up menu.

The Update Record dialog box appears.



Record to Update:
Connection: connScaal
Table to Update: Franchises
Select Record From: rsFranchises
Unique Key Column: fID ☑ Numeric

Get Values From:
Form: fmUpdate (When Submitted)
Form Elements:
txtCity <ignore>
txtState <ignore>
txtZip <ignore>
Column: <ignore> ☐ Numeric

When Done Updating:
Go To URL:

2 Use the Connection and Table to Update pop-up menus to specify the database table containing the records to be updated.

3 In the Select Record From pop-up menu, specify the recordset that contains the record displayed in the HTML form.

4 In the Unique Key Column pop-up menu, select a key column (usually the record ID field) to identify the record in the database table. Select the Numeric option if the value is a number.

5 In the Form pop-up menu, specify the HTML form used to edit the record data.

6 Specify what each form element will update in the database table by selecting a form element, then selecting a field from the Column pop-up menu. Select the Numeric option if the field accepts only numeric values.

7 In the Go to URL text box, enter the page to open after the record has been updated in the table.

8 Click OK.

Deleting a record in a database

Your application can contain a page that allows users to delete records in a database from their browsers.

This kind of page is usually a detail page working in tandem with a master page. The master page lets the user choose a record to delete, then passes the choice to the detail page (see “Going to a detail page” on page 116).

A page that deletes records performs two distinct operations. First, it displays the existing data so the user can see the data about to be deleted. Second, it deletes the record from the database.

To display the existing data before it can be deleted by the user:

1 On the delete page, create an HTML form to display the data.

   Make sure the HTML form has a name and a Submit button. Select the <form> tag on the status bar in the lower left corner of the window to select the form, then open the Property inspector (Window > Properties) and adjust the form's properties as required. You can also relabel the Submit button, if you like.

   Next, you'll create a recordset to hold the record to be displayed.

2 In the Data Binding inspector (Window > Data Binding), click the plus (+) button and choose Recordset (Query).

   If the advanced Recordset dialog box appears, click the Simple button to open the simple Recordset dialog box.

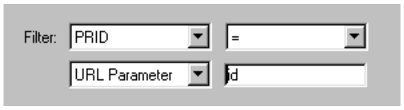
3 Name the recordset and specify where the data you want to display is located using the Connection and Table pop-up menus.

4 Click the Selected option and select a key column (usually the record ID field) and the columns (fields) containing the data to be displayed.

   The columns should have a corresponding form object on your page allowing users to view the data.

5 Configure the Filter area so that the value of your key column equals the value of the corresponding URL parameter passed by the master page.

   This kind of filter creates a recordset that contains only the record—the one specified by the master page. For example, if your key column contains record ID information and is called PRID, and if the master page passes the corresponding record ID information in the URL parameter called *id*, then enter the following information in the Filter area:



Filter: PRID =
URL Parameter id

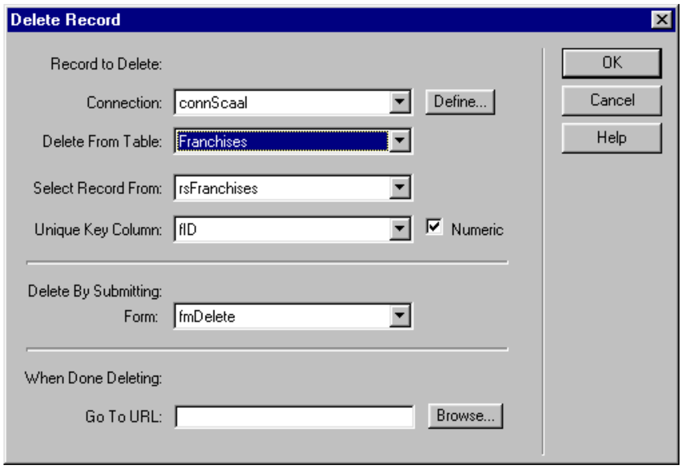
6 Bind each HTML form object to its corresponding recordset field.

   For instructions, see "Making form objects dynamic" on page 98.

To delete the displayed record from the database:

1 In the Server Behavior inspector (Window > Server Behaviors), click the plus (+) button and choose Delete Record from the pop-up menu.

   The Delete Record dialog box appears.



Delete Record

Record to Delete:

Connection: connScaal

Delete From Table: Franchises

Select Record From: rsFranchises

Unique Key Column: fID ☑ Numeric

Delete By Submitting:

Form: fmDelete

When Done Deleting:

Go To URL:

2 Use the Connection and Table to Update pop-up menus to specify the database table containing the records to be deleted.

3 In the Select Record From pop-up menu, specify the recordset containing the records displayed in the HTML form.

4 In the Unique Key Column pop-up menu, select a key column (usually the record ID field) to identify the record in the database table. Select the Numeric option if the value is a number.

5 In the Form pop-up menu, specify the HTML form used to display the record data.

6 In the Go to URL text box, enter the page to open after the record has been deleted from the table.

7 Click OK.

Installing more server behaviors

To give your Web application more functionality, you can install more server behaviors. For example, you can download and install a server behavior from the UltraDev Exchange (Help > UltraDev Exchange).

If you're proficient with JavaScript, VBScript, Java, or ColdFusion, you can write your own server behaviors. For more information, see the *Extending Dreamweaver and UltraDev* book or help system (Help > Extending Dreamweaver and UltraDev).

To install an extension in UltraDev, launch the Extension Manager by selecting Commands > Manage Extensions, then select File > Install Extension in the Extension Manager. For more information, see the help system that comes with the Extension Manager.

CHAPTER 8
Editing and Debugging Dynamic Pages

After creating static pages, adding dynamic content, and adding server behaviors, the final step in building a Web application is making sure it looks and works as planned. UltraDev gives you several ways of editing and debugging dynamic pages:

- You can work in the Document window.
- You can work in the Live Data window, which displays dynamic content.
- You can preview the page in a browser to test how the pages of your application interact.
- You can use the HTML Source inspector to edit HTML source code or server-side scripts, or use the Quick Tag Editor within either the Document or the Live Data window to edit single HTML tags.

You can also edit the UltraDev data formats, or create new ones.

Making basic changes to pages

You can make basic changes to your pages in either the Document window or the Live Data window. For more information on working in the Live Data window, see “Editing in a live data environment” on page 130.

Basic changes you can make include changing the content of existing recordsets, replacing the data sources that supply dynamic content to your page, and changing how the server behaviors on your page work.

Changing the content of a recordset

You can alter the definition of an existing recordset to make additional fields available to the application, or to change the number of records in the recordset. For example, you may decide you want to display the cellular phone numbers of your sales staff on a results page. If your existing recordset doesn’t have a field listing the cellular numbers, you can change the recordset’s definition to include one.

To change the content of an existing recordset:

1 In either the Data Binding inspector or the Server Behavior inspector, double-click the name of the recordset you want to edit.

2 Make your changes in the simple or advanced Recordset dialog box.

   For more information, see “Defining a recordset” on page 77.

3 Click the Test button to view the contents of the updated recordset, then click OK to close the test recordset.

4 If satisfied with your work, click OK.

Replacing or removing dynamic content

After adding dynamic content to a page, you can change the data source providing the dynamic content. For example, you can use the same field in an updated recordset, use a different field in the same recordset, or use a different field in a different recordset. You can also remove dynamic content from a page.

To replace the data source providing the dynamic content:

1 In the Server Behavior inspector, double-click the dynamic content in the list of server behaviors.

   A list of available data sources appears.

2 Select another data source.

3 Click OK.

To remove dynamic content from a page:

Select the dynamic content on the page and press Delete.

Making changes to server behaviors

After adding a server behavior to a page, you can delete it or change its properties. For example, you can make a repeated region display more records per page.

To change the properties of a server behavior:

Double-click the server behavior in the Server Behavior inspector, change the properties in the dialog box, then click OK.

To delete a server behavior:

Select the server behavior in the Server Behavior inspector and click the minus (-) button.

Editing in a live data environment

You can use UltraDev's Live Data window to make changes to your pages in a live data environment. Unlike the Document window, the Live Data window displays a page's dynamic content. More significantly, while the dynamic content is displayed you can do the following:

- Adjust the page's layout using Dreamweaver's standard design tools.
- Add, edit, or delete dynamic content.
- Add, edit, or delete server behaviors. (You can also edit the behaviors' server-side scripts directly in the HTML Source inspector.)

You can toggle between the Document window and the Live Data window by choosing Live Data from the View menu. If a page expects data from the user—for example, the ID number of a record selected in a master page—you can provide the page with that data yourself in the Live Data Settings dialog box.

You must upload all the necessary files, including server-side includes and dependent files, to the server used by the Live Data window. (When UltraDev opens a page in the Live Data window, it only uploads a temporary copy of the page, not the dependent files.)

Links don't work in the Live Data window. To test your links, use UltraDev's Preview in Browser feature. For more information, see "Testing the application's links" on page 134.

To work on a page in the Live Data window:

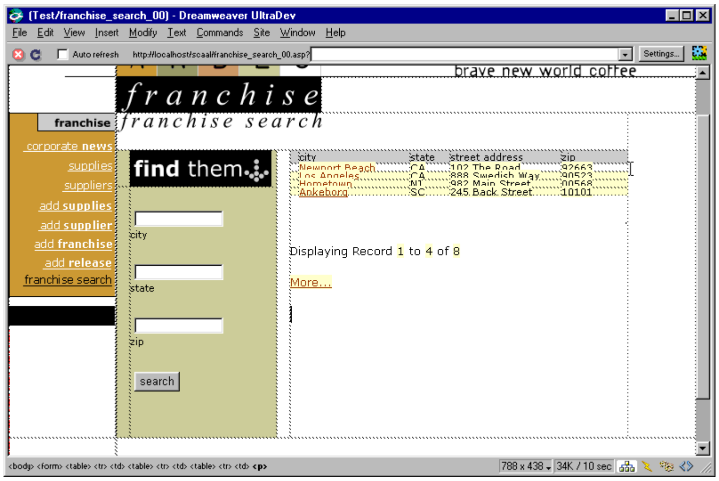
1 Make sure the Document window displays a dynamic page.

In the Document window, placeholders are used for all dynamic content.

2 Choose View > Live Data to switch to the Live Data window.

UltraDev must run a temporary copy of the page on a server before displaying the page and its dynamic content. The process may take a few seconds. To cancel the process, click the Stop button (the button with a white X on a red background).

The Live Data window appears with dynamic content displayed on the page.



(Test/franchise_search_00) - Dreamweaver UltraDev
File Edit View Insert Modify Text Commands Site Window Help
Auto refresh
http://localhost/scaal/franchise_search_00.asp?
Settings...
brave new world coffee
franchise
franchise search
franchise
corporate news
supplies
suppliers
add supplies
add supplier
add franchise
add release
franchise search
find them.
city
state
zip
search
<body> <form> <table> <tr> <td> <table> <tr> <td> <table> <tr> <td> <p>
788 x 438
34K / 10 sec

city	state	street address	zip
Newport Beach	CA	102 The Road	92663
Los Angeles	CA	888 Swedish Way	90523
Hometown	NJ	982 Main Street	00568
Ankeborg	SC	245 Back Street	10101

Displaying Record 1 to 4 of 8

More...

3 If desired, choose View > Invisible Elements to remove the highlighting applied to dynamic content.

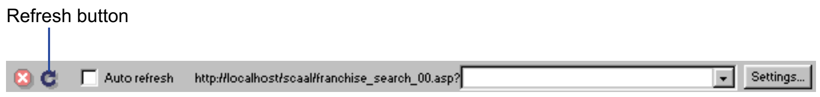
The highlighting may affect how dynamic content is displayed and thus give you an inaccurate picture of the page.

4 If desired, select the Auto refresh option on the toolbar.

The page will refresh whenever you make a change affecting dynamic content. If you have a slow database connection, you may want to ignore this option when working in the Live Data window.

5 Make the necessary changes to the page.

6 If your page expects values from an HTML form using the GET method, enter the values in the text box on the toolbar and click the Refresh button.



Note: A text box for entering values appears only if you specified the GET method in the Live Data Settings dialog box (View > Live Data Settings).

Enter the test data in the following format:

name=value;

where *name* is the variable name expected by your page and *value* is the value held by that variable.

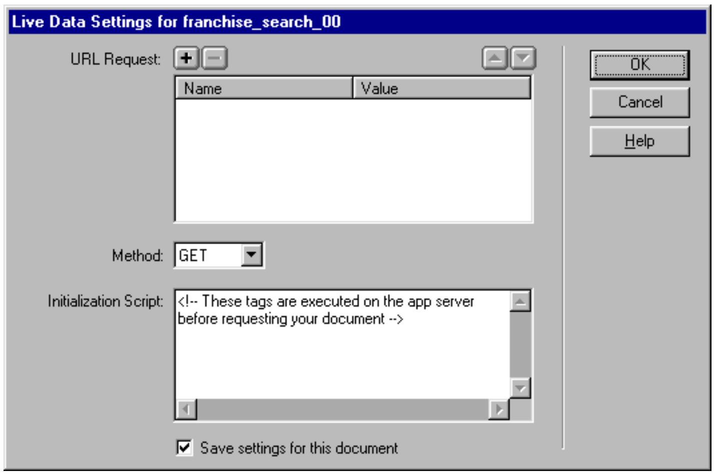
You can also define name/value pairs in the Live Data Settings dialog box (View > Live Data Settings) and save them with the page.

7 Click the Refresh button if your page needs refreshing.

To provide the page with data expected from users:

1 In the Document window, choose Live Data Settings from the View menu.

  The Live Data Settings dialog box appears.



Method: GET

Initialization Script: <!-- These tags are executed on the app server before requesting your document -->

☑ Save settings for this document

2 In the URL Request area, click the plus (+) button to enter a variable your page expects. Specify a name and a test value for each variable.

  Note: You can also specify variables in the Live Data window if you choose the GET method. Enter the values directly in the text box on the Live Data window's toolbar and click the Refresh button.

3 In the Method pop-up menu, select the HTML form method your page expects: POST or GET.

4 In the Initialization Script text area, include any source code you want to insert at the top of the page before it runs. The code usually consists of one or more tags that initialize session variables.

5 To save your settings for the current page, click Save Settings for this Document.

6 Click OK.

Testing the application’s links

Because links don’t work in the Live Data window, this window is not the ideal environment to test how your pages interact. Use the Preview in Browser feature to test this aspect of your application.

Using Preview in Browser with dynamic pages

UltraDev offers a Preview in Browser command that lets you preview documents in a browser at any time. By default, UltraDev takes the document from the local file system and creates a temporary copy to display in your browser. However, because dynamic pages must be run on a server, UltraDev must run the temporary copy on a server before displaying it in the browser. (UltraDev then deletes the temporary file from the server.)

To configure Preview in Browser for dynamic pages:

1 Choose Edit > Preferences, then select Preview in Browser.

2 Select the Preview Using Live Data Server option.

UltraDev uses the same server used to display pages in the Live Data window. See “Configuring the Live Data window” on page 61.

3 Click OK.

4 Upload any related pages, server-side includes, and dependent files to the server.

Preview in Browser only uploads a temporary copy of the page to the server. It does not upload related pages (such as a results or a detail page), dependent files such as image files, or server-side includes. To upload a file, choose Site > Site Files to open the Site window, select the file in the Local Folder pane, and click Put to upload it to your remote site. To define a remote site, see “Defining a remote site” on page 58.

To open a page using Preview in Browser:

Open the page in either the Document window or the Live Data window, then choose File > Preview in Browser, or press F12.

About testing strategies

When testing your application, try to follow the workflow of the application. For example, if you want to test a page that processes data received from an HTML form on another page, open the form page first, fill it out, and click Submit. Otherwise, the test page might not receive the data it needs to work properly.

If you find problems on a page, make the necessary corrections in UltraDev, then test the page again in a browser. For more information, see “Making basic changes to pages” on page 128 and “Editing in a live data environment” on page 130.

Editing the source code

In UltraDev, you can edit the page’s HTML source code and the behaviors’ server-side scripts directly using the HTML Source inspector, or you can use the Quick Tag Editor to edit single HTML tags without leaving the Document or Live Data window.

To use the Quick Tag Editor, select an object, text, or tag, then press Control+T (Windows) or Command+T (Macintosh). For more information on the Quick Tag Editor, see “Editing an HTML tag in the Document window” in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

To open the HTML Source inspector, choose HTML Source from the Window menu. For more information on the HTML Source inspector, see “Editing HTML” in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

Note: Changes made to static content in the HTML Source inspector are reflected in the Live Data window only when you click in the window. Changes made to dynamic content are reflected in the Live Data window only when you click the Refresh button in the window.

You may prefer to use your favorite text editor to hand-code large amounts of HTML, JavaScript, or VBScript. UltraDev lets you use any external text editor, including Notepad (Windows), SimpleText (Macintosh), BBEdit, HomeSite, and other editors. For more information, see “Using external text editors” in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

Editing and creating data formats

A data format is applied to dynamic text on a page to display the data in more user-friendly ways. For example, you can make a date in your recordset appear as March 29, 2000 on the page. For more information on applying data formats, see "Making text dynamic" on page 94.

You can edit the various data formats available in UltraDev, or create new ones.

To edit an UltraDev data format:

1. Open a page containing dynamic text.
2. Select any dynamic text by clicking its placeholder.
3. Make sure the Data Binding inspector is open (Window > Data Bindings), and click the down arrow in the Format column.
4. Select Edit Format List from the pop-up menu.

   The Edit Format List dialog box appears.
5. Double-click any of the listed formats.
6. Make your changes and click OK.
7. Click OK to close the Edit Format List dialog box.

To create a new UltraDev data format:

1. Open a page containing dynamic text.
2. Select any dynamic text by clicking its placeholder.
3. Make sure the Data Binding inspector is open (Window > Data Bindings), and click the down arrow in the Format column.
4. Select Edit Format List from the pop-up menu.

   The Edit Format List dialog box appears.
5. Click the plus (+) button and select a format type—for example, Currency.
6. Define the format and click OK.
7. Enter a name for the new format in the Name column.
8. Click OK to close the Edit Format List dialog box.

A

APPENDIX A
UltraDev for Drumbeat Users

Dreamweaver UltraDev presents a major advance to Web development, combining the popular design features of Macromedia Dreamweaver with the powerful database connectivity that Macromedia Drumbeat first made accessible to Web developers.

UltraDev includes several new features that Drumbeat users have long requested. With UltraDev you can do the following:

- Visually design your Web applications without sacrificing control over source code
- Rapidly build tables with predefined and editable table formats and include complementary colors and layouts to create appealing, browser-safe table designs
- Preview and edit live server-side data in the workspace
- Easily connect to any ODBC, JDBC, or ADO database; connect to, browse, and test against industry standard databases such as Microsoft SQL Server, Oracle8i, Sybase, Informix, and IBM DB2
- Build ASP, JSP, or ColdFusion applications in a single design environment
- Streamline team development with collaborative features such as check in/check out and Design Notes
- Maximize productivity with a library of built-in server behaviors, HTML table editing, CSS styles, and the History palette
- Easily work with existing HTML and ASP sites using UltraDev's integrated site map, which allows you to connect to any directory structure to manage your files
- Easily integrate content from Microsoft Office, Macromedia Fireworks, Flash, Shockwave, and other leading applications

- Tune UltraDev for maximum productivity by customizing the entire application, including menus, using JavaScript and XML
- Extend UltraDev's functionality with new server objects and server behaviors (in addition to writing commands, behaviors and more) using UltraDev's powerful JavaScript API (application programming interface) which builds on the existing Dreamweaver API

This appendix will help you make a quick, smooth transition to UltraDev and will help you convert your existing Drumbeat applications to UltraDev applications.

Understanding the UltraDev terminology and interface

Most of the familiar Drumbeat features can be found in UltraDev. However, the location and terminology may have changed to accommodate UltraDev's extensibility features and to integrate with the design features in Dreamweaver.

The following table shows the UltraDev equivalents of some Drumbeat features and their locations in UltraDev.

Drumbeat feature	UltraDev equivalent
SmartElements toolbar	Object palette, with tabs for different types of objects. Appears by default to the left of the Document window. Show or hide the Object palette by using Window > Objects. You can also insert many objects from the Insert menu.
Attributes tab	Property inspector. Appears by default below the Document window and shows the properties for the selected object. Show or hide the Property inspector with Window > Properties or Control+F3.
Style builder	CSS Style palette (for CSS styles, 4.0 and later browsers). HTML Style palette (for non-4.0 browsers or inline styles).
Interactions Center	Behavior inspector (for client-side scripts). Select Window > Behaviors to show or hide the Behavior inspector. Server Behavior inspector (for server-side scripts). Show or hide the Server Behavior inspector by clicking the Server Behaviors button on the Launcher, or by choosing Window > Server Behaviors.

Drumbeat feature	UltraDev equivalent
Query Manager Basement objects	Data Binding inspector. This inspector shows the dynamic data sources for the current page, much as the Basement shows recordsets, request objects, and session objects in Drumbeat. It allows you to add or delete data sources. Edit recordsets through the Server Behavior inspector. Show or hide the Data Binding inspector by clicking the Data Bindings button on the Launcher, or by choosing Window > Data Bindings.
SQL Query Builder	Recordset dialog box allows you to create your own SQL statement or select from tables, columns, views, and stored procedures to build your SQL statement. When you insert a new recordset (from the Data Binding inspector), the advanced Recordset dialog box appears, from which you can click the simple button if you want the simple Recordset dialog box. Before you create a query, you need to define a connection in the Connection Manager. You can do this either by clicking the Define button in the Recordset dialog box or by choosing Modify > Database Connections.
Publishing a site	Site window To publish a site, switch to the Site window (choose Window > Site Window if it is not already displayed) and choose from the publish options Get/Put, Check In/ Check Out, or Site > Synchronize, as described in “Publishing to the server” on page 160.

Converting your Drumbeat site to UltraDev

If you have already created a site in Drumbeat that you want to convert to UltraDev, you have three choices:

- Convert your existing pages to UltraDev designs and clean up the HTML and page designs as desired.
- Rebuild your pages from scratch using UltraDev's design tools.
- Keep your original pages with their code intact and add them to your site map in UltraDev. You can then add new pages and set up links to the old pages from any new pages you add to the site. Although you will not be able to use the Live Data window to preview the Drumbeat pages with dynamic data, UltraDev can transfer them to the server. (See "Publishing to the server" on page 160.)

Your choice will depend on whether you want to modify the existing pages and how extensive your changes are.

To use the UltraDev design tools most effectively, and to work with the dynamic data tools in UltraDev, you will have to reconstruct your pages to some degree.

Converting your Drumbeat site to UltraDev may take several steps, depending on the complexity of the site, the number of supporting elements you may need to integrate, and the amount of customization you have done in your site.

Creating your site

The first step in creating your UltraDev site is to copy the published pages from your Drumbeat site into a new folder that you'll use in UltraDev. Copy any other assets you use, such as image folders and external scripts, into this folder. You may also want to include the scripts that Drumbeat has generated; although most of these will have to be recreated, you can use them for reference. Any subfolders you use will also be included in the site.

Next, you'll define a new site in UltraDev. Defining a new site is similar to setting up your site preferences and publish settings in Drumbeat and presents the same types of choices.

The following steps assume that you want to convert an ASP or JSP site built in Drumbeat to UltraDev.

To define your UltraDev site:

1 Choose Site > New Site.

2 Select Local Info in the Category list, and choose the local root folder in which your site pages and assets are located.

This is the folder to which you copied your published pages and site assets.

3 Select Web Server Info in the Category list, and provide the location of the Web server that will host your site (and to which you will eventually publish):

- Choose Local/Network if your server is on a network drive (Windows), or if you are running a Web server on your local system.
- Choose FTP if you connect to your Web server using FTP.

4 Complete the other options in the dialog box:

- If you chose Local/Network server, click the folder icon to browse to and select the remote folder where your site files are stored on the server.
- If you chose FTP, enter the host name of the FTP server, the host directory, login and password, and firewall options (if required).

For more information, see "Associating a remote server with a local site" in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

5 Select App Server Info in the Category list, and choose your server model and the server scripting language:

- In the Server Model pop-up menu, select ASP 2.0 or JSP 1.0. If your site does not include dynamic data, choose None.
- If you chose ASP 2.0 as your server model, set the Default Scripting Language to either VBScript or JavaScript.

Note that you can also change the default page extension, if necessary, but the option defaults to the extension that matches your server model choice.

- For Live Data Server, select the location of the server to use to preview your pages in the Live Data window. If you are using Personal Web Server, select Local Folder and enter the URL to access your publishing directory (http:/localhost/sitename/). Select Remote if you are using a remote server, such as the one that will host your remote site or a different staging server on your network.

For more information, see “Configuring the Live Data window” on page 61.

6 Select Site Map Layout in the Category list, and choose the home page for your site.

For more information on configuring your site, see “Laying the Groundwork for an Application” on page 55.

Managing assets in UltraDev

Assets for an UltraDev file are stored in the folder you choose when you define a site. If you want to use assets located elsewhere, you can copy them to your site folder. When you place an object in the Document window and choose to populate it with a file (such as an image) that is not in your site folder, UltraDev asks you if you want to copy it to your site folder. If you want UltraDev to upload it to the server, you should copy it to your site folder.

Working with Drumbeat-generated layouts

UltraDev handles positioning with layers and CSS differently than Drumbeat does, which may cause some problems if you try to work with Drumbeat pages that are designed for 4.0 browsers in UltraDev.

If you open a page that uses layers in Drumbeat (which includes any page designed with the target browser setting at Any 4.0), you will not be able to convert it to a table in UltraDev because of the extensive nesting of layers.

If you want to be able to redesign your pages in UltraDev and use the layout tools most effectively, it is recommended that you set the target browser settings in your Drumbeat site (or a snapshot of it) to Navigator 3.0 and republish your pages. That way Drumbeat will publish table code rather than layers. You can then open the page in UltraDev and convert your table code back to layers, if you need to do further design work. See Layers in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

Once you have converted your pages to Navigator 3.0 and brought them into your UltraDev site, you can import the style sheets and reapply styles if necessary. (See “Importing style sheets” on page 144.) If you have other elements on your pages that are specific to 4.0 and later browsers, you can also add these elements again.

After importing your pages into UltraDev and adjusting the design, you can use File > Check Target Browsers in UltraDev to verify that they will work in the browsers you choose.

Note: Some SmartElements specific to Microsoft Internet Explorer 4 may not be detected by the Check Target Browsers command. Test their functionality in the browser and recreate them in UltraDev if necessary.

Preserving Drumbeat SmartPages

A SmartPage in Drumbeat is a collection of pages designed for different target browsers, along with an HTML file that contains a script to redirect users to the appropriate browser version.

Instead of SmartPages, UltraDev has a reporting feature that tests your site against specific browsers. Use this feature if you want to design separate pages for different browsers, or pages that will satisfy requirements for more than one browser. The advantage of this approach is that you can test your pages against a wider variety of browser versions and are not limited to just the four browser types defined in Drumbeat.

To check your page against specific browser versions:

1. Select File > Check Target Browsers.
2. Choose the browser or browsers you want to test against, and click Check.

   UltraDev generates an HTML report page that lists and describes any errors found.

You can also make any page viewable in a 3.0 browser by using File > Convert > 3.0 Browser Compatible. You can choose to convert layers to tables and CSS styles to HTML markup.

If you have several versions of a page for different target browsers, you must then create a script to redirect users to the correct version. (You can use and modify an existing Drumbeat SmartPage script for this.)

If you want to simply preserve the various browser-specific pages you have already created in Drumbeat, add them to your new UltraDev site, along with the SmartPage, which contains the redirection script.

Importing style sheets

A style sheet created in Drumbeat can be imported into UltraDev for use in your UltraDev site. You can import all the styles in Drumbeat or only selected ones. If you want only selected styles, first export the styles you want from Drumbeat's style builder using the Export feature to create a separate style sheet file.

In UltraDev you can use two types of styles: HTML styles and CSS styles. These correspond to Drumbeat's inline styles (applied with the formatting toolbar) and container styles (applied with the Style attribute). HTML styles (or inline styles) use the FONT tag to apply the style to an element and are recognized by all browsers. CSS styles (or container styles) use a style sheet, which allows a style to be defined once and applied to any number of elements. A change to a style in the style sheet affects all the elements with that style at once. CSS styles are recognized only by 4.0 and later browsers.

If you have converted your pages to Netscape 3 before adding them to your UltraDev site, you must reapply any CSS styles. First, remove the HTML style from the text, then apply the CSS style.

To import a style sheet into UltraDev:

1 Open the CSS Style palette (Window > CSS Styles).

2 Click the Open Style Sheet button on the bottom of the palette.

3 In the Edit Style Sheet dialog box, click Link.

4 In the Link External Style Sheet dialog box, select Add As: Import.

5 Click Browse to locate the style sheet (*.css) you want to import, and click OK.

   If you want all the Drumbeat styles from your original site (both the global and site-specific styles), use the CSS file that Drumbeat publishes in the publish directory for your site. If you chose Any 4.0 as your target browser, Drumbeat has created two style sheets: one for Internet Explorer (IE) and one for Netscape Navigator. Use the IE style sheet, because the nomenclature is more straightforward and you will be able to use the same style names that you used in Drumbeat.

6 Click Done. The styles in the linked style sheet are added to the CSS Style palette.

   If you want to check or change the style characteristics, you can edit the imported styles in UltraDev's style editor. For more information on creating and editing styles, see Formatting text with CSS style sheets in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

To remove an HTML style from a text selection:

1 Open the HTML Styles palette.

2 Select the text.

3 Click Clear Selection Style.

To apply a CSS style:

1 Open the CSS Style palette.

2 Select the text to which you want to apply the style.

3 In the CSS Style palette, select the style to apply.

About UltraDev templates

You can create templates in UltraDev just as you do in Drumbeat. However, the two kinds of templates work very differently. In UltraDev, a template is a separate file that you create and save as part of your site assets. You then apply the template to each page that you want to inherit those elements.

In a UltraDev template, you create editable and noneditable regions on the page. Noneditable regions contain the elements that you want all the pages to inherit. Editable regions may often contain placeholder text or images as a layout guide. Content in the editable regions can then be replaced on each page individually.

When you change a template and save it, you are given the opportunity to update the pages that use the template; you can choose to update all the pages or only specific pages. You must then republish the pages. See Modifying templates and updating the site in the *Using Dreamweaver* guide or Dreamweaver Help (Help > Using Dreamweaver).

You can create new templates in UltraDev to match your Drumbeat templates. Although it's possible to import a Drumbeat page into UltraDev, convert it to a template, and then modify it, this is not advisable unless you are already very familiar with Dreamweaver layout tools and templates. For more information about designing and using templates, see Creating templates in the *Using Dreamweaver* guide or Dreamweaver Help (Help > Using Dreamweaver).

A possible alternative for creating a template is to use a graphics editor to take a screen capture of one of your Drumbeat pages. You can then use this as a tracing image to design a template in UltraDev. For more information on using tracing images, see Using a tracing image in the *Using Dreamweaver* guide or Dreamweaver Help (Help > Using Dreamweaver).

Using dynamic data in UltraDev

UltraDev refers to database-driven Web pages as dynamic pages. A dynamic page is any page that is modified at run time by the server before it is sent to a requesting browser. Dynamic data includes recordsets, session variables, request objects, and command objects.

UltraDev expands the server models available in Drumbeat and exposes the scripting interface so that you can more easily inspect and add to the server behaviors you use in UltraDev. Experienced programmers can easily modify existing server scripts and write their own scripts.

UltraDev supports the same database applications as Drumbeat, including desktop database applications such as Microsoft Access and Lotus Approach, as well as industrial-strength databases such as Microsoft SQL Server, Oracle8i, IBM DB2, and Informix.

Setting up data connections

After you have defined your site, you can set up the data connections and data sources for your site. If you have imported pages from Drumbeat, you can open a document that contains dynamic data in UltraDev and reconnect the data sources so you can work with it.

To create dynamic pages in UltraDev, you must first set up the data connections for your database using the Connections dialog box.

If you want to use a sample database for design purposes, you can also define a test connection to use instead of connecting to the full database at design time. This is useful if you have a very large database, or if you do not want to risk changes to your existing database during testing.

After you set up your data connection, you must set up the data sources you want to use by creating recordsets on each page; see "Using recordsets" on page 148.

To set up the data connection in the Connection Manager:

1 Choose Modify > Connections to open the Connection dialog box.

2 Click New.

The Define Connection dialog box appears.

3 In the Run-Time tab, enter a name for the connection. If you are using an ODBC database with a data source name (DSN) that has already been defined on your system, enter the name of the DSN.

4 Choose a server data source type:

- For ASP, choose ADO (ODBC Data Source Name) if you want to use an existing DSN on your system. Choose ADO (Connection String) if you want to define your own connection string to connect to the database.
- For JSP, choose JDBC.

5 Define the connection.

- If you chose ADO (ODBC Data Source Name), select the database driver from the DSN pop-up menu.
- If you chose ADO (Connection String), enter your connection string, or click the ODBC button to choose from the existing ODBC data sources on the system, then edit the string as necessary.
- If you are using a JDBC data source, specify the URL for the database and the driver. (If you have already defined these for your Drumbeat site, you can copy and paste the appropriate strings from the JDBC Connections dialog box in Drumbeat.)
- Enter the username and password, if they are required to access the database.

For more information, see "Working with database connections" on page 62.

6 Click Test to test the connection. If the connection is made, the message "Connection was made successfully" appears. Click OK. If the the test is not successful, revise your information as necessary until the connection is successful.

7 If you want to use a sample database for design purposes, click the Design-Time tab. Deselect Same as Run-Time, then choose the server type and define the connection string as in step 5.

8 In the Define Connection dialog box, click OK. The new connection should now appear in the Connection dialog box. Click Done.

Data connections that have been defined for the local system are available to use in any site. You can see the available data connections on the system by choosing Modify > Connections to open the Connection dialog box.

Using recordsets

Recordsets in UltraDev are not site-level elements, as they are in Drumbeat. A recordset is specific to the page on which it is defined. To use dynamic data on a page, you must set up the recordsets you want to use for the page.

To set up a recordset as a data source for a page, you use the plus (+) button in the Data Binding inspector, then choose the data or define your SQL statement using either the simple or advanced Recordset dialog box. For detailed instructions, see "Defining a recordset" on page 77.

Any elements on the page originally bound to a recordset in Drumbeat will have to be rebound to the new recordset in UltraDev.

- To bind form elements to a recordset, see "Making form objects dynamic" on page 98.
- To bind text, see "Making text dynamic" on page 94.
- To bind images, see "Making images dynamic" on page 97.

After you have created the data sources you want to use on the page, they will appear in the Data Binding inspector and the Server Behavior inspector. You can edit the recordset from either inspector.

Although recordsets are specific to the page on which they are created, you can reuse them by copying and pasting a recordset from one page to another.

To edit a recordset:

Double-click the recordset in the Data Binding inspector or the Server Behavior inspector to display the Recordset dialog box.

To copy a recordset to another page:

1. Select the recordset in either the Data Binding inspector or the Server Behavior inspector.
2. Click the arrow button in the upper-right corner of the inspector and choose Copy from the pop-up menu.
3. Open the page where you want to copy the recordset.
4. Click the arrow button in the upper-right corner of the inspector and choose Paste from the pop-up menu.

Remember that session, request, and command objects are also data sources. If you use any of these on your pages, you must set up these sources, too. You can use the Data Bindings inspector to add them (see "Defining data sources" on page 92).

About dynamic pages

When you work with database content during design time in Drumbeat you see a representation of your live data on the layout (usually the first record in a recordset or the default values from a SQL parameter).

UltraDev offers two ways to work with dynamic data. When you place dynamic data in the Document window (the design and editing environment), a placeholder icon represents the dynamic content. To see the actual dynamic data, you can use the Live Data window.

The Live Data window displays sample data on the page, so that you can adjust your design around it. You can't edit or interact with the dynamic data, but you can manipulate other static areas of the page. See "Editing in a live data environment" on page 130.

When you're ready to test the interactions with the data on your page, use UltraDev's File > Preview in Browser command. See "Testing the application's links" on page 134.

Setting up a search page

Because a recordset is a different type of object in UltraDev than in Drumbeat, you use a different process for setting up search and results pages in UltraDev.

- In Drumbeat you put the recordset you want to filter on the search page. Then you set up an interaction to filter the recordset by the value of the form element and redirect to the results page.
- In UltraDev, the search function is handled entirely through the results page. You do not need a recordset on the search page (unless you are displaying results on the same page). You simply set up the action of the form to go to the results page and choose the correct method for your server (GET or POST). The value of the form element is submitted to the server when the Submit button is clicked. The results page retrieves this value and uses it to filter the recordset on the results page.

To set up a simple search page in UltraDev:

1. On the Form tab of the Object palette, select the Form object to add a form to the page.
2. Select the Text Field object on the Object palette to add a text field to the form.

   This is the text field in which the user enters search criteria. You can add instructional text for the user next to the text field, as necessary.
3. In the Property inspector, give the text field object an appropriate name.

   You'll use this name later when you set up the results page to retrieve the value from the server.
4. Select the Button object on the Object palette to add a Submit button to the form.
5. Select the form by clicking the dotted outline.
6. In the Property inspector, set the Action attribute of the form to the page that will display the results by entering the page name or browsing to locate the page. If you enter the page name manually, be sure to include the page extension (.asp or .jsp).
7. Choose the Method that matches your server (GET or POST).

   If you choose Default for the method, the method used will be the default method on the server. However, you must know which method is used to retrieve the value, or use a simplified retrieval statement (see "Setting up a results page" on page 151).

For more information, see "Building a search page" on page 71.

Setting up a results page

A results page displays the results of a search. To create a results page, you must first set up the display of the results as desired, using text or a table with a repeating region, and bind the dynamic elements to the recordset fields that you want to display. (See "Using tables to display data" on page 152).

After you have designed the results display, you must define a SQL variable to filter the recordset. The search page sends the value of this parameter to the server, which then passes it to the results page. To edit the recordset, double-click it in the Server Behavior inspector.

To append a variable to filter the recordset by a value from a form element:

1 In the Recordset dialog box, click the Variables plus (+) button to add a new variable.

2 Enter a name for the variable in the Name column.

3 Enter a default value for the variable.

   Do not enclose the value in quotation marks; use the single quotation mark in your SQL statement instead. To use a wildcard in the expression, you can use % to represent a string of zero or more characters, or an underscore (_) to indicate one character.

4 For the run-time value, enter the value to retrieve from the server, formatted as a request:

- If the server method is POST, use the syntax **Request**.Form("ElementName".)
- If the method is GET, use the syntax Request.QueryString("ElementName".)
- If your server can handle the default scenario, you may use the simplified syntax Request("ElementName".)

   Replace ElementNamewith the name you gave to the text field that contained the search criteria on the Search page.

5 Add the variable to your SQL statement as appropriate.

   For example, to filter a database of names by an entry made in a text field on a form, you can add:

   ```
   where lastname like 'varname%'
   ```

   If the user enters *a* for the search criteria, the search will replace 'varname%' with 'a%' and the search result will be all names beginning with *a*. You must use single quotation marks around the variable that will be replaced in the search.

For more information, see "Building a results page" on page 72.

Using tables to display data

AutoTables are used in Drumbeat to display data from a content table. The content table may contain static data that was added manually, it may have been created from an external CSV file, or it may have a recordset as its source. In UltraDev, a distinction is made between these three types of tables.

- To create a table with static data that you want to add manually, use an HTML table (choose Insert > Table). Then populate the table as you want. See Inserting a table and Adding content to a table cell in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).
- To create a table from a CSV file, use the Insert > Tabular Data command. See Importing table data in *Using Dreamweaver* or in Dreamweaver Help. Tabular data imported from a static CSV file is not dynamic (it will not change if the original CSV file changes).
- To create a table from a recordset, such as a table used for a result set on the Result List page in Drumbeat, use Insert > Table, populate each column with dynamic data, and designate the row as a repeated region.

If you have a Result List page or an AutoTable display of a result set that you created in Drumbeat, you will have to recreate that result set in UltraDev.

First, make sure you have set up the data source you will be using (see “Defining a recordset” on page 77).

To set up the result set display in a table:

1. Choose Insert > Table. You can also drag the Table object from the Object palette’s Common panel onto the Document window.
2. In the Insert Table dialog box, for the Row option, choose 2 if you want to show table headings, or 1 if you do not want headings. Choose the number of columns that matches the number of data fields you want to display.
3. Enter and format the column headings in the first row, if you have chosen to display headings.
4. Open the Data Binding inspector (Window > Data Bindings) if it is not already open. Select the data source you want to use, expanding it to show the available columns.
5. Click inside a table cell in which you want to insert data from the recordset. In the Data Binding inspector, select the field you want to display in the column, and click Insert.
6. Repeat step 5 for each column of data you want to display.

To define the repeated region in the table:

1. Select the table row with the dynamic data that is to be repeated.
2. In the Server Behavior inspector, click the plus (+) button and choose Repeat Region from the pop-up menu.
3. In the Repeat Region dialog box, choose the recordset and the number of records to display at one time, and click OK.

For more information, see “Displaying more than one record” on page 108.

After you have created your table with the repeated region, you can view sample data in the Live Data window. You can then add navigation buttons if necessary. You can also create a link from a column in the table to a detail page. For instructions on how to apply the server behaviors to create both these functions, see “Applying server behaviors” on page 157.

Creating a detail page

Often you will want your results page to include a link to a detail page—a page that displays more details about a selected record. Creating a detail page is simply a matter of adding the elements to display the data and binding them to the appropriate fields in the recordset. However, to maintain the state of a selected record, you must also add the server behavior that locates the selected record when requested from the results page.

To create a link to a detail page:

1. Select the dynamic data that you want to link (the name that displays in the result list).
2. In the Server Behavior inspector, click the plus (+) button and choose Go to Detail Page.
3. In the Go to Detail Page dialog box, enter the name of the detail page or browse to select it.

   Use this server behavior only when you are not using record navigation on your detail page.
4. Enter a name for the URL parameter to pass (or accept the default).
5. Select the recordset and the unique column (this must be included in the recordset SQL statement)
6. Deselect the boxes to pass existing URL and form parameters.

To create a detail page and maintain the state of the requested record:

1. Define the recordset for the page. In the SQL statement, include the unique column that will be used to locate the record.
2. Place the text, image, and form elements on the page for displaying the data.
3. Bind each element to the appropriate field in the database.
4. In the Server Behavior inspector, click the plus (+) button and choose Move to Record > Move to Specific Record. In the dialog box, select the recordset and the unique column. Make sure that the name of the URL parameter to match is the same as the parameter passed from the referring page.

Note: The recordset on the referring page need not be identical to the one on the receiving page (that is, the SQL `SELECT` statement may be different), as long as the unique column in the database is included on both pages and is correctly specified.

For more information, see “Building a detail page” on page 73.

To set up buttons for navigating through records, see “Applying server behaviors” on page 157.

Setting up update and insert pages

You can set up update and insert pages in UltraDev for editing your database. To use an insert or update page you created in Drumbeat, you must define the new recordset for the page and then bind the data fields to the new recordset.

An update page needs to get its record parameter from a referring page. The referring page typically contains a button linked to the update page to allow users to update the current record. Two server behaviors are involved in this process:

- The server behavior Go to Related Page is applied on the referring page.
- The server behavior Move to Record > Move to Specific Record is applied on the update page; it retrieves the value passed from the referring page and displays the matching record.

On an insert page, you do not need to apply the Move to Specific Record behavior, since the page allows users to insert a new record, rather than locate and edit a current record.

For more information, see "Updating a record in a database" on page 119 and "Inserting a record in a database" on page 118.

To set up an update page:

1. Define the recordset for the page. (See "Defining a recordset" on page 77.)
2. Add the form elements to the page to display the data. Add a Submit button and text labels for the dynamic fields as desired.
3. Bind the form elements to the appropriate fields in the recordset. Select the form element, then in the Data Binding inspector, expand the Recordset to show the columns. Select the column you want to bind the form element to and click Bind To. (See "Making form objects dynamic" on page 98.)
4. In the Server Behavior inspector, click the plus (+) button and select Move to Record > Move to Specific Record. In the dialog box, select the recordset, the unique column and the URL parameter.
5. Click the plus (+) button and select Update Record. In the dialog box, in the Record to Update section, select the connection, recordset and table to update, and the unique key column
6. In the Get Values From section, select the form (if more than one form is on the page), and check the bindings for each form element. You can change or assign the bindings of any form element by selecting the form element from the form elements list, then choosing the correct column from the column list.
7. Select the URL to go to when done updating. Then click OK.

To set up an insert page:

1 Follow steps 1–3 in the previous procedure for creating an update page.

2 In the Server Scripts inspector, click the plus (+) button and select Insert Record. In the dialog box, in the Record to Update section, select the connection, recordset and table to update, and the unique key column

3 In the Get Values From section, select the form (if more than one form is on the page). Then assign each form element to the appropriate column in the recordset by selecting the form element from the form elements list, then choosing the correct column from the column list.

   Assigning the form elements to the correct column will be easier if you name your form elements appropriately first.

4 Select the URL to go to when done updating. Then click OK.

Scripting in UltraDev

In Drumbeat, you apply both client-side and server-side scripts to elements on the layout using interactions. UltraDev distinguishes between client-side and server-side scripts and manages them differently.

Client-side scripts (written in JavaScript) are called behaviors and are applied through the Behavior inspector (Window > Behaviors). Behaviors can be sorted by browser compatibility on the Behavior inspector.

Server scripts, called server behaviors, are scripts that are processed by the server, and include all interactions involving dynamic data. For ASP, they are written in JavaScript or VBScript; for JSP they are written in Java. Server behaviors are applied and managed through the Server Behavior inspector (Window > Server Behaviors). As in Drumbeat, a number of prewritten server behaviors are available and can easily be applied through the Server Behavior inspector.

You can also extend the available scripts in UltraDev with your own custom scripts. For more information, see *Extending Dreamweaver and UltraDev*.

About client-side scripts

Drumbeat publishes client-side scripts as external scripts, linked from the main document file. To access these scripts in UltraDev, you need to copy them over from your published directory on the server to your local UltraDev site folder.

If you are not sure what external scripts are referenced in your file, you can see them by choosing View > Head Content. The scripts referenced by your document are indicated by the script icon at the top of the Document window. Select a script icon to view the script's properties in the Property inspector, including the source (the referenced file), language, and type.

To edit scripts that are included in the file, click Edit in the Property inspector to display the Script Properties editor. If a script is in a linked file, you must edit it outside UltraDev, or copy it into the <SCRIPT> tag in your UltraDev document.

For simple client-side scripts with UltraDev behavior equivalents, it may be best to delete the link to the old script from your document and apply new behaviors in UltraDev. Custom scripts you have written can be added as extensions to UltraDev. For more information see *Extending Dreamweaver and UltraDev*.

Applying server behaviors

To apply a server behavior (server script) you use the Server Behavior inspector. A number of prewritten behaviors are available for common database operations such as navigating sets of records.

For instance, you can use predefined server behaviors to do the following:

- Add navigation buttons to a page to display the next or previous set of records in a table or a data loop
- Set up navigation links between recordset pages, maintaining the state of the recordset (see "Setting up update and insert pages" on page 155.)
- Set up a link from a column in a dynamic table to a detail page (see "Creating a detail page" on page 154.)

These behaviors are similar to the interactions you use in Drumbeat.

To apply navigation buttons to display the next or previous set of records:

1. Place Next and Previous image buttons in the Document window.
2. Select the Next button. In the Server Behavior inspector, click the plus (+) button and choose Move to Record > Next.
3. In the dialog box that appears, choose the recordset from the pop-up menu. Then click OK.
4. Select the Previous button. In the Server Behavior inspector, click the plus (+) button and choose Move to Record > Previous.
5. In the dialog box that appear, choose the recordset from the pop-up menu. Then click OK.

Viewing and editing server behaviors

In UltraDev's Document window, server behaviors appear as placeholder icons. If you open a page you created in Drumbeat that has server behaviors applied, you will see an ASP or JSP icon where dynamic content is inserted, instead of actual data.

To view sample data, you can switch to the Live Data window. However, in the Document window, you can view and edit the server behaviors on your page that access the dynamic data. You cannot edit behaviors in the Live Data window.

To view or edit the behaviors associated with an icon in the Document window:

1 Select the script icon (ASP or JSP).

2 In the Property inspector, click Edit.

The Edit Contents dialog box appears, in which you can view and edit the existing script.

Converting custom Drumbeat contracts to server behaviors

Server behaviors you have written for use in Drumbeat can easily be reused in UltraDev. You can add your own custom scripts to the predefined server behaviors.

For information on developing your own objects, commands, property inspectors, behaviors, and translators, see *Extending Dreamweaver and UltraDev*.

Drumbeat SmartElements and UltraDev objects

UltraDev's Object palette is similar to Drumbeat's SmartElements toolbar. The Object palette contains several panels, each of which can be customized. Most of the default SmartElements found on the Drumbeat SmartElements toolbar can be found in two panels:

- The Common panel provides standard elements such as images, rollover images, tables, and plug-in objects.
- The Form tab provides form elements, such as text fields (edit boxes), drop-down lists, and form buttons.

You can drag objects from the Object palette to the Document window, just as you do SmartElements in Drumbeat. To set up an object's properties (attributes), you use the Property inspector.

If you've developed custom SmartElements in Drumbeat, you can convert them to UltraDev objects and even include them on your Object palette. However, this requires some programming knowledge.

For information on developing your own objects, commands, property inspectors, behaviors, and translators, see *Extending Dreamweaver and UltraDev*.

Publishing to the server

When you publish your site in Drumbeat, Drumbeat creates the pages for you and any supporting files (such as JavaScript and style sheet files) and transfers them to the server using the publish settings you have provided. However, no local copies of these files exist, unless you've used your Staging settings to publish your files to a temporary location for testing or safekeeping. Coordinating changes you make to the Drumbeat EDF file and these different sets of published files can sometimes be a problem.

UltraDev allows you to publish and keep track of changes on your site in several ways. Since UltraDev always maintains a copy of your published files in your local site folder, you can work on the local copies without affecting the copies on the server. When you're ready to transfer them to the server, you can do so from the Site window:

- In the Site window, use the Get option to transfer files from the server to your local directory, or use the Put option to transfer files from your local directory to the server.
- If you are working in a collaborative environment, you may want to enable the check in/check out feature, which locks the file on the server and keeps track of who has it. Enable this feature in the Define Site dialog box (Site > Define Sites). Once enabled, the Check In and Check Out buttons are added to your Site window and can be used in place of Get and Put.

For more about these procedures, see Getting and putting files and Using the checkin/checkout system in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

Another convenient feature is the Synchronize Site option, which automatically synchronizes the content of the local site folder and the server site. UltraDev looks for the latest copy of each file in the site; if the local and remote copies are different, UltraDev copies the latest version to the appropriate location.

To synchronize local and remote versions of your site:

1 Choose Site > Synchronize in the Site window.

2 In the dialog box, choose the option you want from the synchronize pop-up menu (entire site or selected files only).

3 Choose an option from the direction pop-up menu. You can get newer files from the remote server, put newer files to the remote server, or get and put newer files wherever they are at once.

4 Click OK.

5 In the dialog box that appears, choose which files you want to include in the synchronize operation, and click OK.

For more information, see Synchronizing the files on your local and remote sites in the *Using Dreamweaver* guide or in Dreamweaver Help (Help > Using Dreamweaver).

INDEX