User Manual

TextWrangler

THE Text Editor for Anyone who Types

Bare Bones Software, Inc.

TextWrangler™ 2.1.1

Product Design	*Rich Siegel, Patrick Woolsey, Jim Correia, Steve Kalkwarf*
Product Engineering	*Jim Correia, Jon Hueras, Steve Kalkwarf, Rich Siegel*
Engineers Emeritus	*Chris Borton, Tom Emerson, Pete Gontier, Jamie McCarthy, John Norstad, Jon Pugh, Mark Romano, Rob Vaterlaus*
Documentation	*Stephen Chernicoff, Caroline Rose, Jeff Mattson, Jerry Kindall, John Gruber, Philip Borenstein, Simon Jester, Rich Siegel, Patrick Woolsey*
Additional Engineering	*Polaschek Computing http://www.polaschek-computing.com/*
Icon Design	*Ultra Maroon Design http://www.ultramaroon.com/*
PCRE Library Package	*written by Philip Hazel and © 1997-2000 University of Cambridge, England*
PopupFuncs™ Technology	*Eric Slosser*

License Agreement:

You, the Licensee, assume responsibility for the selection of the program TextWrangler to achieve your intended results, and for the installation, use, and results obtained from the program. Breaking the package seal and installing the program constitutes your acceptance of these terms and conditions. If you do not accept these terms and conditions, then do not break the package seal or install the software, and return the package for a full refund.

License:

You may use the program and documentation on a single machine and copy the program and documentation into any machine-readable or printed form for backup or support of your use of the program and documentation on the single machine, provided that no more than one copy of the program and documentation may be used on a single machine by no more than one user at a time. You may not use, copy, modify, or transfer the program or documentation, or any copy thereof, in whole or in part, except as provided in this Agreement. If you use, copy, modify, or transfer the program or documentation, or any copy thereof, in whole or part, except as expressly provided for in this agreement, your license is automatically terminated.

The license is effective on the date you accept this Agreement, and remains in effect until terminated as indicated above or until you terminate it. If the license is terminated for any reason, you agree to destroy the program and documentation, together with all copies thereof, in whole or in part, in any form, and to cease all use of the program and documentation.

Limited Warranty and Limitation of Remedies:

The program, documentation and any support from Bare Bones Software, Inc., are provided "as is" and without warranty, express and implied, including but not limited to any implied warranties of merchantability and fitness for a particular purpose. In no event will Bare Bones Software, Inc. be liable for any damages, including lost profits, lost savings, or other incidental or consequential damages, even if Bare Bones Software, Inc. is advised of the possibility of such damages, or for any claim by you or any third party.

General Terms:

This Agreement can only be modified by a written agreement signed by you and Bare Bones Software, Inc. and changes from the terms and conditions of this Agreement made in any other manner will be of no effect. If any portion of this Agreement shall be held invalid, illegal, or unenforceable, the validity, legality, and enforceability of the remainder of the Agreement shall not in any way be affected or impaired thereby. This Agreement shall be governed by the laws of The Commonwealth of Massachusetts, without giving effect to conflict of laws provisions thereof. As required by United States export regulations, you shall not permit export of the program or any direct products thereof to any country to which export is then controlled by the United States Office of Export Administration, unless you have that agency's prior written approval.

Use, duplication and disclosure by the United States Government of the program or any part thereof is subject to restrictions as set forth in subdivision (g)(3) of Rights in Data - General Clause in Federal Acquisition Regulations ("FAR") 52.227.14, the Commercial Computer Software Restricted Rights Clause in FAR 52.227.19. and/or subdivision c(1)(ii) of Rights in Technical Data and Computer Software Clause in Department of Defense FAR 252.227.7013.

Acknowledgment:

You acknowledge that you have read this agreement, understand it, and agree to be bound by its terms and conditions. You further agree that it is the complete and exclusive statement of the agreement between you and Bare Bones Software, Inc. which supersedes all proposals or prior agreements, oral or written, and all other communications between you and Bare Bones Software, Inc. relating to the subject matter of this agreement.

Contents

Chapter 8   Searching with Grep   129

Chapter 9   Browsers   159

Chapter 10   Preferences   165

CHAPTER

1

Welcome to TextWrangler

This chapter introduces you to TextWrangler, a high-performance text editor for the Macintosh.

In this chapter

Getting Started

Thank you for selecting TextWrangler, a high-performance text editor for the Macintosh.

We recommend that you read at least Chapters 1 through 4 of this manual to familiarize yourself with the installation and basic operation of TextWrangler. You may also wish to read or skim any other chapters that cover features you frequently use. After you have installed TextWrangler, the best way to learn it is to use it.

What Is TextWrangler?

TextWrangler is a high-performance text editor. Unlike a word processor, which is designed for preparing printed pages, a text editor focuses on providing a means of producing and changing content. Thus, TextWrangler does not offer fancy formatting capabilities, headers and footers, graphics tools, a thesaurus, and other staples of feature-laden "office" software. Instead, it focuses on helping you manipulate text in ways that word processors generally cannot.

In service of this goal, TextWrangler offers powerful regular expression–based ("grep") search and replace, multi-file search, sophisticated text transformations, intelligent text coloring, and other features not usually found (or missed) in word processors.

TextWrangler also has features that make it easier to edit specific kinds of text, such as source files for programming languages.

How Can I Use TextWrangler?

Use TextWrangler any time you need to create or edit source code or text documents of any kind. Whether you need to find (or change!) all the occurrences of a particular word, phrase, or string of text in a set of files, or modify or reformat large text files of any sort, TextWrangler is the right tool for the job.

Editing Source Code

TextWrangler is a powerful tool for editing numerous types of source code, with the following features:

- Syntax coloring helps you read your code and find simple errors.
- The function pop-up menu lets you can quickly find the functions in your files.
- Find Differences lets you compare two versions of a text file and merge the differences.
- Find in Reference lets you look up documentation in the Developer Help Center.

Editing Text Files

TextWrangler is a full-featured text editor that makes it easy to create, edit, and search any sort of text file, such as release notes, articles, books, or TeX documents. It's also an excellent tool for pre- and post-processing any files that contain textual data, such as database exports, PostScript files, or server logs.

- Grep searching lets you find and change text that matches a set of conditions that you specify.
- Multi-file search and replace lets you quickly search and modify text files anywhere on your computer.
- Numerous text manipulation and processing commands allow you to reformat or rearrange the information in text files, e.g. add or remove hard line breaks, change the case of selected text, or remove duplicate lines from a list.
- Extensive AppleScript support allows you to combine multiple processing steps for reuse, and enables you to easily transfer data into and out of TextWrangler to other applications
- International text support lets you edit Unicode files (UTF-8 and UTF-16), as well as files saved in most non-Roman single-byte scripts.
- FTP and SFTP support lets you open and save text files located on remote servers.
- BBEdit plug-in support gives you access to a wide variety of plug-in tools that add functionality to TextWrangler.

Human Interface Notes

TextWrangler enhances the behavior of its menus and dialogs as described in this section.

Dynamic Menus

Many of TextWrangler's pull-down menus are dynamic: if you hold down the Shift or Option key while a menu is open, you can see some of the items change. The illustration below shows what the File menu looks like normally (left) and when you hold down the Option key (right).
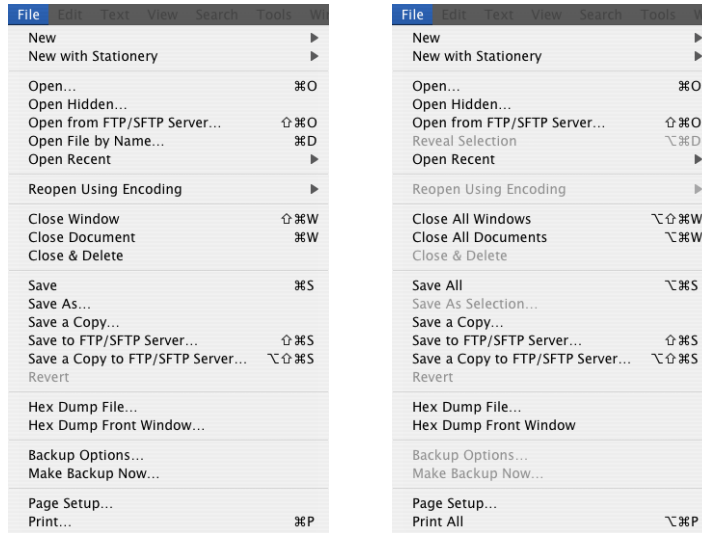
You can use the Shift and Option keys when you choose an item from a menu or when you use the Command-key equivalents.

Bypassing Options Dialogs

You may have noticed that commands that require additional settings to be made before they are performed appear on the menu with ellipses after their names. To bypass this step and use the command with its most recent settings, hold down the Option key while selecting the menu item. For example, "Zap Gremlins…" in the Text menu becomes "Zap Gremlins" when the Option key is pressed and, when chosen, will zap gremlins in the frontmost text document using the current settings.

Keyboard Shortcuts for Commands

Many of TextWrangler's commands have keyboard shortcuts. TextWrangler lets you reassign the shortcuts for any menu item to suit your own way of working. To change the keyboard shortcut for any menu item, choose the Set Menu Keys command in the TextWrangler (Application) menu.

Many other TextWrangler features can have keyboard shortcuts assigned as well. Here's how to set them:

Feature	Set Keys in…
Menu commands	Set Menu Keys (in the TextWrangler menu)
Plug-ins	Plug-In Tools palette (in the Window menu)
Stationery	Stationery palette (in the Window menu)

To display any of TextWrangler's floating palette windows, use the Palettes submenu in the Window menu.

Contextual Menus

When you Control-click on selected text or at the insertion point in a text window, TextWrangler's contextual menu provides quick access to a few relevant and frequently used commands, so you do not have to hunt around in the menu bar for them.

Snappy Palettes

When you move or resize palettes (floating windows), they will "snap" to the edges of the screen and the edges of other palettes. You can override this behavior by holding down the Shift key while dragging or resizing.

Dialog Box Key Equivalents

You can use keys to click buttons or select options in most of TextWrangler's dialog boxes. Certain keys have the same meaning in all dialogs:

- Pressing either the Return or Enter key is the same as clicking the default button.
- Typing Command-period or pressing the Escape key is the same as clicking the Cancel button.
- You can use the Cut, Copy, Paste, Clear, and Select All commands (either from the Edit menu or with their Command-key equivalents) in any dialog that has a text field.

To see the other key equivalents for a particular dialog, hold down the Command key. After a brief delay, the key equivalents appear next to the buttons in the dialog. For example, this is the Find & Replace dialog without the Command-key equivalents showing:
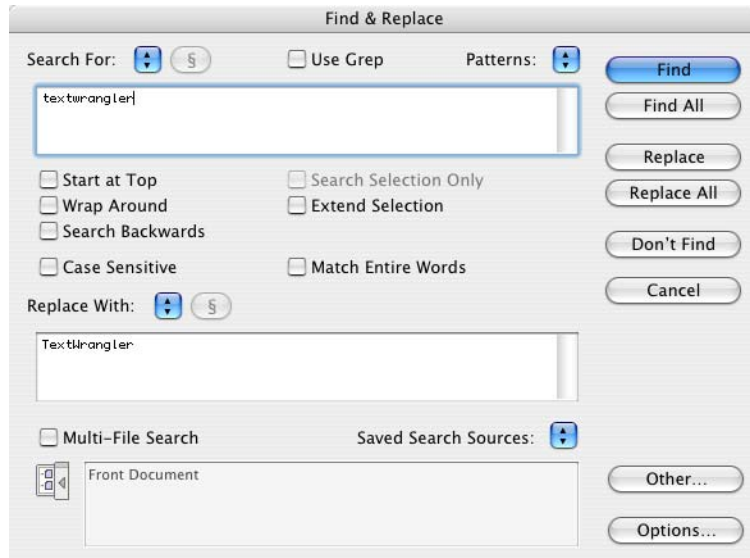
This is the Find & Replace dialog with the Command-key equivalents visible:
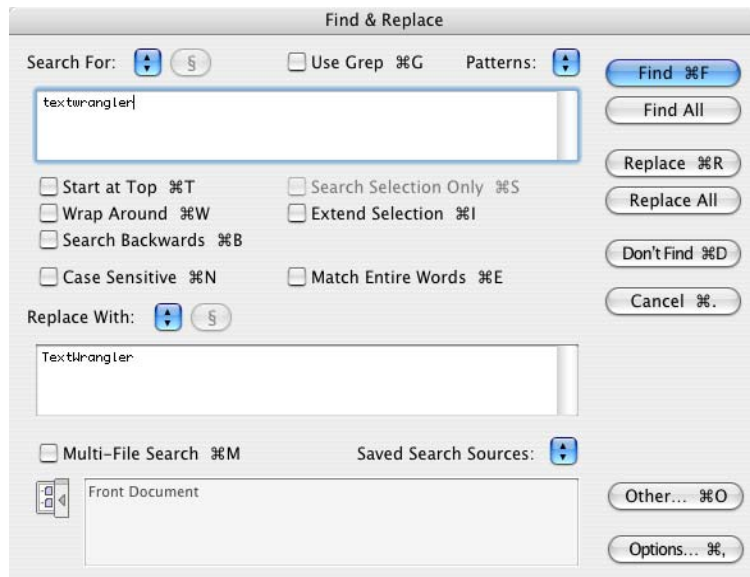
TextWrangler waits briefly before displaying the Command-key equivalents so that you can type a sequence quickly without making the dialog flicker.

What's New in this Version

TextWrangler 2 raises the bar for Macintosh text editors. Here are a few of the many features and enhancements offered by this version:

- Support for opening and managing multiple text documents via a single window interface.

- Enhanced multi-file searching, offering threaded searches and diverse search sources.

- Use of ATSUI for text rendering greatly improves handling of Unicode files, particularly those containing glyphs from multiple disparate languages.

- Runs Text Factories created with BBEdit, allowing you to apply multiple text processing actions to diverse source documents.

- Shebang (#!) menu offers integrated access to Unix scripting environments such as Perl, Python, shell scripts, and more.

- Command line tool enhancements including support in the "edit" tool for opening files from FTP/SFTP servers, and a new "twdiff" tool.

- Text menu commands expanded and enhanced: imported capabilities of former plug-in tools, sheet support, improved recordability, and more.

- Spell checking now uses the Mac OS X system spelling checker.

- Streamlined application support folder configuration supports both global and local add-in items.

- New visual Page Guide and tab stop indicators.

- Current line highlighting preference.

Support Services

The Bare Bones Software web site provides up-to-date information on TextWrangler, as well as BBEdit, Mailsmith, and other Bare Bones Software products, including FAQs, product updates, plug-ins, sample scripts, and other support and informational materials.

```
http://www.barebones.com/
```

For support help and information, follow the links to the Support area, where you'll find a wide range of information including:

- Frequently Asked Questions (FAQ) — Information and answers for commonly encountered questions and problems. We strongly recommend you read the TextWrangler FAQ before resorting to any other means of inquiry.

- Product Updates — The latest maintenance versions of our products will always be available for download.

- Technical Support — If you have a registered version of TextWrangler, and you can't find the information you need on our web site, send email to: <support@barebones.com>

Note   We do not offer telephone support for TextWrangler. Please refer to the support resources available on our web site for information and assistance. Registered users may also contact us via email.

Installing TextWrangler

This chapter tells you how to install TextWrangler on your Macintosh. It also describes the files TextWrangler creates, where it puts them, and how to install or remove optional components of TextWrangler.

In this chapter

Basic Installation

TextWrangler is supplied as a single application file. Specific system requirements and installation instructions are described below, and the organization of TextWrangler's supporting files is described in subsequent sections.

System Requirements

IMPORTANT TextWrangler 2 requires Mac OS X 10.3.5 or later. The software will not run on Mac OS 9, or any earlier versions of Mac OS X.

Installing TextWrangler

TextWrangler is distributed in the standard Mac OS X disk image format (a ".dmg" file), or may be available on demo CD-ROMs. To install TextWrangler, just drag the "TextWrangler" application file from the disk image or CD-ROM to the Applications folder (or other desired location) on your hard drive.

Updating an Existing Copy

IMPORTANT   In order to update TextWrangler when future versions become available, you need only quit the TextWrangler application, and replace it with the updated version. The first time you launch a newer version of the software, TextWrangler will prompt you should any further actions, such as updating the command-line tools, be needed.

Upgrading from a Previous Version

IMPORTANT   In addition to installing the TextWrangler application, if you are upgrading from a previous version, you will need to manually copy across any items you added to that version's "TextWrangler Support" folder into TextWrangler's new application support folder. You should **not** simply rename your existing "TextWrangler Support" folder. (See "TextWrangler's Application Support Folders" on page 25.)

Please carefully read the remainder of this chapter, since the organization of TextWrangler's supporting files has changed considerably. We have provided specific suggestions and tips for transferring your customized support items in each category.

Welcome Dialog and Registration

The first time you launch TextWrangler, it will display the "Welcome to TextWrangler" dialog. This dialog allows you to choose whether to register your copy of TextWrangler, and whether to install the current command-line tools.

You are not required to register to use TextWrangler. However, if you do not register, you will not be eligible for discounted cross-upgrade pricing, free technical support, or other benefits and offers.

If you wish to register, enter your name and a valid email address in the appropriate fields, choose whether you wish to be notified of future upgrades and special offers by selecting or deselecting the Notify Me checkbox, and press the Register Now button. (You may view our privacy policy before registering by pressing the View Privacy Policy button.) If you do not wish to register, press the Skip Registration button You may also press the Later button, and TextWrangler will prompt you to register the next time you launch it.

TextWrangler's Application Support Folders

TextWrangler makes use of an application support folder to store and organize a variety of items that add functionality, such as plug-in tools, scripts, and more. Such items are kept in subfolders according to their purpose (described below).

IMPORTANT   TextWrangler 2 does not make use of the "TextWrangler Support" folder that previous versions employed, nor can you can place the application support folder alongside TextWrangler itself. Instead, TextWrangler's application support folder must be present in either or both of the following locations:

- Global (items available to all users): /Library/Application Support/ TextWrangler/

- Local (user-specific items): ~/Library/Application Support/TextWrangler/

Note   In the above path to the local application support folder, ~ is the customary Unix shorthand to indicate the user's home directory. If written out as a full path, this would normally be "/Users/<username>/Library/Application Support/TextWrangler/".

Using the Global Application Support Folder

You can use the global application support folder to provide a common set of supporting items in TextWrangler to each user of the machine.

Users whose accounts do not have admin privileges will not be able to modify the contents of the global application support folder, since it resides in the system hierarchy. This arrangement can be advantageous when configuring the software for use in shared-machine environments, such as labs or common-area workstations.

However, if such an arrangement is not desirable for your purposes, you should not create a global application support folder. Instead, each user can maintain their own local application support folder for TextWrangler, which they may add items to, or remove items from, at will.

Using a Local Application Support Folder

If a local application support folder does not exist when TextWrangler starts up, TextWrangler will create one for you, and populate it with a set of factory default examples. Although none of these default items are essential for doing basic tasks with TextWrangler, and you can remove any or all of them, they provide additional functionality that you may wish to retain.

Application Support Folder Contents

TextWrangler's application support folders contain various subfolders, each of which holds a specific type of support item. As indicated, items in some subfolders can be loaded from both the global and local application support folders; other items may only be used from a specific location.

If there are multiple copies of any plug-ins or language modules, TextWrangler will use the latest version regardless of location. For all other items, TextWrangler lists the global and local sets separately.

To prevent alias loops, TextWrangler will not follow aliases to folders that are placed inside any of the subfolders within the application support folder. We also recommend that you do not try to share plug-ins or scripts between TextWrangler and other applications, and that you not make aliases to plug-ins or scripts located on remote (server) volumes.

Language Modules

[Global, Local]

This folder contains plug-in modules that tell TextWrangler about new programming, scripting, or markup languages so that it can colorize them appropriately and generate function pop-up menu listings for them. Three modules are supplied with the default installation—Python, Setext, and TeX. A list of additional modules from third-party developers is available on our web site.

Upgrading   You should move or copy over any compatible third-party language modules that you wish to preserve.

Menu Scripts

[Local only]

This folder contains scripts that are attached to TextWrangler menu items. See Chapter 11 for more information on creating and using menu scripts.

Note   Only items in the instance of this folder which resides in the local application support folder will be loaded.

Upgrading   You should move or copy over any menu scripts that you wish to preserve.

Plug-Ins

[Global, Local]

The Plug-Ins folder contains third-party code modules that add features to TextWrangler. Each plug-in adds a command to TextWrangler's Tools menu and you can run the plug-in by choosing its name from that menu.

To install a plug-in, drag and drop it directly onto the TextWrangler application icon in the Finder. TextWrangler will open, if necessary, and present an alert asking you to confirm that you want to install the plug-in. If there is already a plug-in with the same name in your Plug-Ins folder, you will be further prompted whether to replace it with the version you are dragging. If you confirm the operation, the plug-in you dragged will be placed at the top level of your Plug-Ins folder and the one it replaced will be moved to the Trash. You will need to quit and relaunch the TextWrangler application in order to use the newly installed plug-in.

Various plug-ins and information are available from our web site, as well as the BBEdit Plug-In SDK if you are interested in writing your own plug-ins.

Upgrading If you have any compatible third-party plug-ins that you wish to use with TextWrangler 2, you may move or copy them into the Plug-Ins folder.

IMPORTANT You should **not** copy any of the factory-supplied plug-ins from previous versions of TextWrangler to use with TextWrangler 2. To determine whether a plug-in was factory-supplied, select it in the Finder, choose the Get Info command, and check its version information.

You will also **not** be able to use any third-party plug-ins that have not been updated to function on Mac OS X. Contact the developers of your plug-ins or visit the Bare Bones Software web site for more information on the availability of updated plug-ins.

Scripts

[Global, Local]

The Scripts folder contains compiled AppleScripts that appear in the Scripts menu (left). You can place scripts in this folder and use the Scripts menu to run the script. Scripts may be placed within subfolders (up to four levels deep) to organize them. A floating Scripts palette lets you activate scripts with a double-click and assign keyboard shortcuts to any script. Several basic scripts are supplied in this folder.

You can hide, or show, all items included from the global folder by using the menu item "Hide/Show Library Scripts".

Upgrading You should move or copy over any customized scripts that you wish to preserve. Note that some scripts written for use with older versions of TextWrangler may not work with TextWrangler 2. (Please see Chapter 11 for more details and tips on modifying your existing scripts.)

Shutdown Items

[Local only]

The items in this folder are opened when you quit TextWrangler. Usually, this function will be used to run scripts of some sort. This folder is installed empty by default.

Shutdown items are now run after all windows have been closed, and only if TextWrangler is actually quitting. Previously, shutdown items were run before all windows were closed, and were run whenever the application was told to quit (either by the Quit menu command or via the scripting interface), regardless of whether it actually quit or not. Thus, if you wish to run any items as the immediate result of a Quit command, you should write a menu script attached to TextWrangler•Quit.

Upgrading You should move or copy over any shutdown items that you wish to preserve.

Startup Items

[Local only]

When launched, TextWrangler will open any items it finds in this folder. This folder is installed empty by default.

If the items present are documents of a type that TextWrangler knows how to handle (such as text files), TextWrangler will open them directly. If you place a compiled OSA (AppleScript, or any other OSA-compliant scripting language) script in this folder, TextWrangler will execute the script. If you place a folder alias here, TextWrangler will open a disk browser window based at that folder. If you place other types of items in this folder, TextWrangler will ask the Finder to open them.

Upgrading You should move or copy any file or application aliases that you wish to preserve. If you have any AppleScripts startup items, please see the preceding upgrade note for the Scripts folder about script compatibility.

Stationery

[Global, Local]

This folder contains stationery files for use with TextWrangler's New with Stationery command, and the Stationery List palette. Stationery files may be placed within subfolders (up to four levels deep) to organize them.

You can hide, or show, all items included from the global folder by using the menu item "Hide/Show Library Stationery".

Upgrading You should move or copy over any stationery documents that you wish to preserve.

Text Factories

[Global, Local]

This folder contains text factory documents which you can invoke via the Text Factories menu, the Apply Text Factory command in the Text Menu, or the Text Factories palette. If this folder does not already exist, TextWrangler will create it the first time you choose Open Text Factories Folder from the Text Factories menu. This folder is installed empty by default. For more information on using text factories,

Unix Support

[Global, Local]

This folder contains the Unix Scripts and Unix Filters folders, which are used to build the Shebang menu and the floating Unix Scripts and Unix Filters palettes. You can place scripts and filters within subfolders (up to four levels deep) of their respective folders to organize them. Some example Perl, Python, and shell scripts and filters are supplied with the standard installation. The Unix Script Output file stores output from scripts, and the Unix Script Logs folder stores output logs for specific source files. See Chapter 12, "Unix Scripting and the Command Line," for more information on this folder.

You can hide, or show, all items included from the global folder by using the menu items "Hide/Show Library Scripts" or "Hide/Show Library Filters".

Upgrading   You should move or copy over any Unix scripts or filters that you wish to preserve.

Preference Files and Folders

When you start up TextWrangler, it may create the files and folders noted in this section.

TextWrangler Preferences File

All of TextWrangler's basic preference settings are stored in the file "~/Library/ Preferences/com.barebones.textwrangler.plist", which is created and maintained using standard OS services.

This change brings several benefits, including improved durability of the preferences file, better performance when accessing remote storage (e.g. with a network home folder), and the ability to modify preference settings outside of TextWrangler by using appropriate "defaults write" commands (see "Optional settings via 'defaults write'" on page 192).

Upgrading   TextWrangler 2 will import relevant preference settings from previous versions of TextWrangler, if a suitable "TextWrangler Prefs Data" file is available, i.e. can be found within a "TextWrangler Preferences" folder in either the old TextWrangler Support folder or in ~/ Library/Preferences.

TextWrangler Preferences Folder

The folder "~/Library/Preferences/com.barebones.textwrangler.PreferenceData/" contains ancillary preference and settings files and folders for TextWrangler. Its standard contents are as follows.

Document State.plist

TextWrangler stores state information for individual documents in this file.

File Filters

TextWrangler stores user-defined file filter patterns in this file.

FTP Bookmarks.xml

TextWrangler stores user-defined FTP and SFTP bookmarks in this file, which supersedes the previous "FTP Bookmarks" file. You should not attempt to directly edit the contents of this file; instead, use the FTP Bookmarks preferences panel to add, remove, or modify your bookmarks.

Grep Patterns.xml

TextWrangler stores user-defined search patterns in this XML file, which is located in your active TextWrangler Preferences folder. You should not attempt to directly edit the contents of this file. Instead, use the Patterns pop-up menu in the Find dialog to add entries, or use the Text Search preferences panel to add, remove, or modify stored grep patterns.

Upgrading   If you have created and saved any custom grep patterns in a previous version of TextWrangler, these patterns will be imported; otherwise, a set of factory default patterns will be created.

Recent Files & Favorites

This folder contains aliases to the most recent local files that you have opened, or URL clippings for any files opened from FTP or SFTP servers. The items stored in this folder are used to create the Open Recent list in TextWrangler's File menu, and the file lists in the Find Differences dialog.

To set the number of items shown in the Open Recent list, use the "Remember the (#) most recently used items" option on the Application panel of the Preferences window.

Note   You may lock items in this folder to have them persist as Favorites. To do this, open the Recent File & Favorites folder and use the Finder's Get Info command to open an info window for each item (alias) you wish to lock; then turn on the Locked option. Locked items will be displayed at the bottom of the list below a separator line, and are not counted against the specified item limit.

Recent Folders & Favorites

This folder contains aliases to folders that have recently been searched, or compared in a Find Differences operation. The items stored in this folder are used to create the folder search pop-up menu in the Find & Replace dialog, and the folder lists in the Find Differences dialog.

To set the number of items shown in these lists, use the "Remember the (#) most recently used items" option on the Application panel of the Preferences window.

Note   You may lock items in this folder to have them persist as Favorites. To do this, open the Recent Folder & Favorites folder, and use the Finder's Get Info command to open an info window for each item (alias) you wish to lock; then turn on the Locked option. Locked items will be displayed at the bottom of the list below a separator line, and are not counted against the specified item limit.

Working with Files

This chapter discusses how to use TextWrangler to manipulate text files.

In this chapter

Launching TextWrangler

To launch TextWrangler, double-click the TextWrangler application icon or a TextWrangler document. Holding down the following keys at launch has the indicated effects, overriding any options set in the Startup panel of the Preferences window. When one of these keys is held down, TextWrangler will beep after it finishes launching.

Modifier	Function
Option	Suppress startup items only
Shift	Disable all plug-ins, tools, external services, and startup items

Startup Items

When launched, TextWrangler will look for a folder named Startup Items in its application support folder (see page 25). If this folder is found, TextWrangler will open any items it finds in the folder.

If the items present are documents of a type that TextWrangler knows how to handle (such as text files), TextWrangler will open them directly. If you place a compiled AppleScript in this folder, TextWrangler will execute the script. If you place a folder alias here, TextWrangler will open a disk browser window based at that folder.

If you place other types of items in this folder, TextWrangler will ask the Finder to open them. If you often edit shell scripts, for instance, you may want to place an alias to the Terminal application in the TextWrangler Startup Items folder so that it will start up automatically whenever you run TextWrangler.

If you wish, you may place the actual Startup Items folder in any convenient location, create an alias to it, and place the resulting alias in TextWrangler's application support folder. Be sure to name the alias "Startup Items" so that TextWrangler can locate it.

Creating and Saving Documents

To create a new text document or special-purpose window within TextWrangler, pull down the File menu and open the New submenu. Since TextWrangler uses different kinds of documents for specific purposes, you will see several options, as follows:
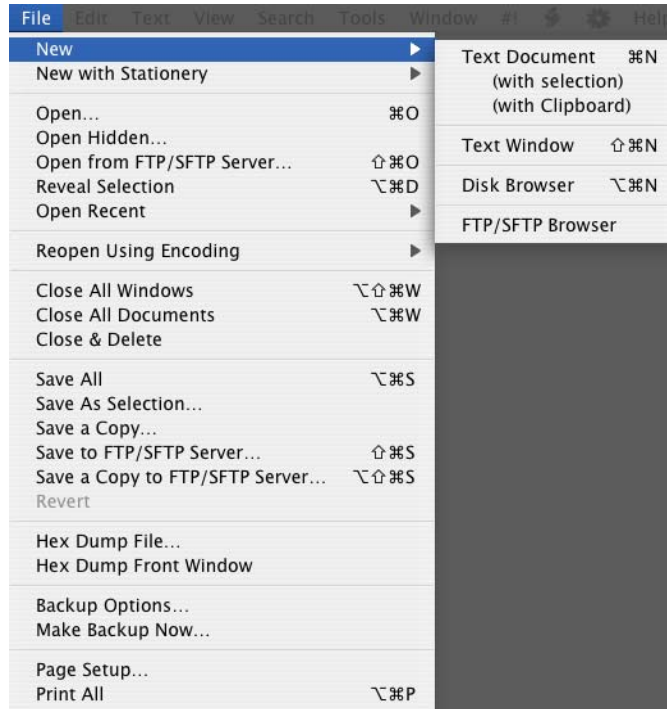
The available commands and their effects are as follows:

- Text Document: Opens an empty text document.

- (with selection): Opens a new text document containing any text selected in the active document and having the same display font, saving you the trouble of copying and pasting it.

- (with Clipboard): Opens a new text document and automatically pastes the contents of the current clipboard into it.

- Text Window: Opens a new text window (see "Text Windows" later in this chapter for more information).

- Disk Browser: Opens a new disk browser (see Chapter 9 for more information).

- FTP/SFTP Browser: Opens a new FTP/SFTP browser (see later in this chapter for more information).

You can also create a new text document by selecting text in any application which supports Mac OS X Services, and choosing the New Window with Selection command from the TextWrangler submenu of the Services menu. TextWrangler will open a new text window containing a copy of the selected text.

When you want to save a new text document:

1 Choose the Save or Save As command from the File menu.

TextWrangler opens the Save dialog:
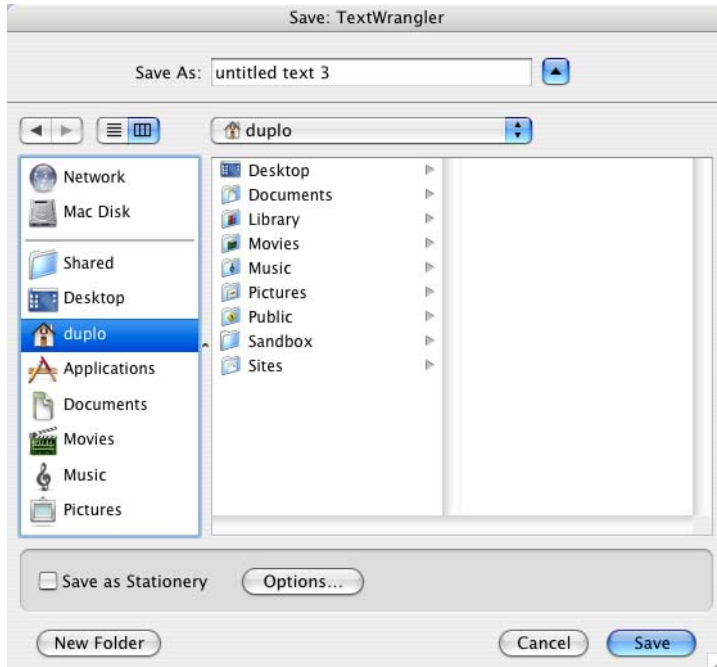
2 Give the file a name.

3 Change any desired options (see below).

4 Click Save to save the file.

You can also create a new document from the selected text in any open window with TextWrangler's contextual menu. Simply Control-click the selected text and choose New (with selection) or Save Selection from the menu that appears. Depending on which command you choose TextWrangler will either create a new editing window containing the selected text, or display the Save dialog and allow you to create a new file containing the selected text. The new file will use the same options (see "File Saving Options," below) as those of the original parent document.

If you want to save a copy of a file using the currently selected text as the file name, hold down the Option key and choose Save As Selection from the File menu. TextWrangler displays the Save dialog with the selected text already entered as the file's name.

Saving a Copy of a File

You can save a copy of a file with TextWrangler's Save a Copy command in the File menu. Just like the Save As command, the Save a Copy command displays a Save dialog and lets you choose a name and location for the file. However, unlike the Save As command, where TextWrangler will start working with the new file you saved in place of the original, when you use Save a Copy, you create a new file in the designated location, but keep working with the original file.

For example, say you are editing a document called Test.c and use the Save a Copy command to save a document called Backup-Test.c. The next time you choose the Save command, TextWrangler saves the changes to Test.c and not to Backup-Test.c.

File Saving Options

TextWrangler's Save dialog is the standard Macintosh Save dialog with these additions:

- Save As Stationery: When this option is selected, TextWrangler saves the document as a stationery pad. When you later open this document, TextWrangler uses it as the basis of a new untitled document. The new document will inherit the contents and display settings of the stationery document, but TextWrangler will prompt you for a name when you save it.

- Options button: This button opens a dialog box that lets you control what kind of line breaks and encoding the file should have.

Setting the Line Breaks

The line breaks settings let you choose what kinds of line breaks TextWrangler writes when you save the file. Choose:

- Unix line breaks (ASCII 10) for most purposes, including use with most Mac OS X applications, or for files being saved to a Unix file server.

- Macintosh line breaks (ASCII 13) if you will be using the file with Classic Macintosh applications or will be sending it to another Macintosh user.

- DOS line breaks (ASCII 13/10) if the file resides on a DOS or Windows file server or if you will be sending it to someone who uses a DOS or Windows system.

- Unicode line breaks for a file saved in any Unicode format. (See "Saving Unicode Files" on page 43.)

Setting the Encoding

TextWrangler lets you save documents using any character set encoding supported by Mac OS X, including a variety of Unicode formats (see "Saving Unicode Files" on page 43). To select an encoding, choose its name from the Encoding pop-up menu. The list of available encodings is controlled by your preference settings (see "Text Encodings Preferences" on page 182).

When you select an encoding that requires a Unicode file format, you can also choose "Unicode" as an option from the Line Breaks pop-up menu in this dialog. (Unicode has its own line-ending standard.)

Files saved as Unicode from TextWrangler are given a type of 'utxt'—the standard for Unicode text files on the Mac. UTF-8 files are given a type of 'TEXT' for compatibility with other applications; however, TextWrangler will also recognize such files with type 'UTF8'.

Note You can choose which encodings appear in the Encoding pop-up menu in the Text Encodings panel of the Preferences window.

File State

You can control whether TextWrangler saves document states by means of the Save Document State option in the Text Files: Saving panel of the Preferences window.

TextWrangler no longer saves document state in the resource fork of the document's file, and does not distinguish between the "MPW" and "TextWrangler" state information created by older versions. This change offers a number of advantages, including: no longer creating resource forks for files, better compatibility with source-control systems, and easier personalization (when working with shared files, other users' display options do not affect you).

Long File Names

TextWrangler fully supports the use of "long" and Unicode file names. Such file names can be up to 255 characters long when stored on disks formatted as HFS Plus.

Saving with Authentication

TextWrangler supports saving files that require Administrator privileges, if you possess the necessary user and password information to enable this. For example, you can edit and save files that are owned by, and only readable by, the "root" user. Authenticated saving is particularly useful in conjunction with TextWrangler's Open Hidden command, which allows you to see and open files in hidden folders (like /bin and /usr).

When you open a file for which you do not have write privileges, TextWrangler will display a slash through the pencil icon in the status bar. To edit the file, click the pencil icon. TextWrangler will prompt you to confirm whether you wish to unlock the file. (Option-click the pencil icon to skip the confirmation dialog.)

When you are finished editing, simply choose Save from the File menu. TextWrangler will prompt you to authenticate as a user with Administrator privileges. Type a suitable user name and password to save the file.

Opening Existing Documents

There are several ways to open existing documents with TextWrangler.

- Double-click any file with a TextWrangler document icon.

- If TextWrangler is running, choose the Open, Open Hidden, or Open Recent command from the File menu.

- Select the name of a file in a TextWrangler editing window; then use the Open Selection command in the File menu.

- Double-click a file name in a browser's file list (see Chapter 9, "Browsers").

- Drag a file's icon to the Windows palette (see Chapter 6, "Working with Windows").

- Drag a file's icon to the document drawer of any text window (see Chapter 4, "Window Anatomy).

- Drag a file's icon to the TextWrangler icon or to an alias of the icon.

- Select a file in the Finder, and use the Open File command from the TextWrangler submenu of the Services menu.

TextWrangler can natively open files with type 'TEXT', 'utxt', 'UTF8', or 'PICT'. (Of course, you cannot edit PICT files in TextWrangler, but it will display them.)

You can view any image files or movie files whose format is supported in QuickTime by opening them with TextWrangler. You can also have TextWrangler ask QuickTime to try to open non-text files (by selecting the appropriate setting in the Application panel of the Preferences window). TextWrangler can then open image files and sounds in their own windows, rather than displaying them as text. If this preference is not active, TextWrangler will open the files in their "raw" condition as if they were text documents.

Choosing the Encoding for a Document

When you open a document, TextWrangler will automatically examine its contents for any indication of the proper encoding, and attempt to handle it appropriately. If TextWrangler cannot determine the proper encoding, and you opened the file with the Open command, it uses the encoding specified in the Read As pop-up menu on the Open dialog. Otherwise, it uses the encoding specified by the "If the file's encoding can't be guessed, use" preference setting in the Text Files: Opening panel of the Preferences window.

Note   You can choose which encodings appear in both the Read As and the "If the file's encoding can't be guessed, use" pop-up menus by using the Text Encodings preferences panel.

Here are the details of the steps that TextWrangler goes through to determine the proper encoding for a file:

1 If the file is well-formed HTML or XML, TextWrangler looks for an "encoding=" or <meta charset=> directive.

2 **If the file contains a TextWrangler state resource, TextWrangler uses the encoding stored in the state resource.**

3 **If the file contains a UTF-8 or UTF-16 (Unicode) byte-order mark, TextWrangler opens it as that type of Unicode file.**

4 **If the file has a resource that contains font information (such as a 'styl' resource) and that resource specifies a multi-byte font, TextWrangler opens the file as a Unicode file.**

5 **If you are opening the file with the Open command, TextWrangler uses the encoding specified Read As pop-up menu on the Open dialog.**

6 **Finally, it uses the encoding specified by the "If the file's encoding can't be guessed, use" pop-up menu on the Text Files: Opening panel of the Preferences window.**

To change the encoding for a file after opening it, open the File pop-up menu on the window's status bar and use the Encoding submenu.

Note   If an encoding change results in the conversion of a document's contents from a single-byte script to a multi-byte script, TextWrangler will mark the document as being "dirty" or changed.

Using the Open Command

To open a file with the Open command:

1 Choose Open from the File menu.

TextWrangler displays the Open dialog box:
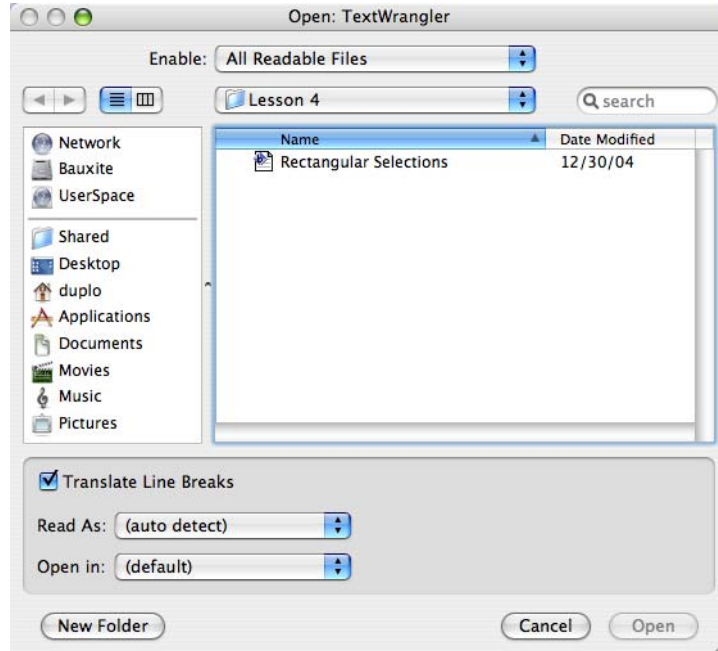
2 Select the file you want to open.

You can select (or deselect) multiple files by holding down the Command key or the Shift key as you click the files.

3 Change any desired options (see below).

4 Click Open to open the file.

You can use the options described below when you open a file.

Show Pop-Up Menu

This pop-up menu lets you choose what kinds of files can be selected from the Open dialog. If you know a file contains text, but it does not appear in the Open dialog, this means that the Macintosh type of the file is not set to 'TEXT'. This is sometimes the case with files received from other computers or downloaded from the Internet. Choose "All Files" from the Show pop-up menu to open the data fork of any file as a text file.

IMPORTANT   Given that Mac OS X does not set the file type of any text file created by a Unix program to a useful value, you may find it helpful to turn on the Map the File Name option in the Text Files: Opening panel of the Preferences window. With this preference turned on, TextWrangler will inspect the file's name to see if it can determine whether the file is text or not. TextWrangler will first attempt to map the file name to the list of

suffix-to-language mappings specified in the Languages preferences. If a file name matches up with a language (even if the language is "None"), the file is assumed to be a text file. Thus, you can use TextWrangler's own suffix mappings to convince it to recognize as text any files whose suffixes are not in Mac OS X's built-in list of file-suffix-to-file-type mappings. If no match is found in the Languages preferences, TextWrangler will next apply the Internet Config file name mappings. (See Chapter 10, "Preferences," for more details.

Translate Line Breaks

When this option is selected, TextWrangler translates DOS or Unix line breaks when opening a file. Otherwise, TextWrangler leaves the original line breaks untranslated.

The setting of this option is not preserved between uses of the Open command, unlike the other options in the Open dialog. Generally, you will want to change this option only temporarily, e.g. to read in a particular file. To change the default setting of this option, use the Translate Line Breaks option from the Text Files: Opening panel of the Preferences window.

Read As

When opening a file, you can tell TextWrangler what encoding it uses. Usually, TextWrangler will correctly auto-detect the encoding, but if it does not, you can close the file and try again, manually specifying the desired format. Chapter 5 includes more information on encodings.

Open In

When opening one or more files, you can use the options on this pop-up menu to override your default document opening preferences. These options have the following effect:

- (default): TextWrangler will open the selected documents according to your preference settings.

- Front Window: TextWrangler will open all of the selected documents into the frontmost text window. If there are no text windows open, or the frontmost text window contains an active sheet, this option will be disabled.

- New Window: TextWrangler will open all of the selected documents into a new text window.

- Separate Windows: TextWrangler will open each of the selected documents into its own text window.

Using the Open Hidden Command

The Open Hidden command in the File menu presents a dialog with the same appearance and behaviors as the standard Open dialog, except that it shows invisible files (including both files whose invisible attribute has been set, and those whose names begin with a period).

Using the Open Recent Command

The Open Recent submenu contains a list of files you have opened recently. To open one of these files, choose it from the Open Recent submenu.

To set the number of items in the Open Recent list, use the "Remember the most recently used items" option on the Application panel of the preferences window.

Using the Reopen using Encoding Command

The Reopen using Encoding submenu contains a list of all available text encodings. To reopen the current text document and have its contents interpreted using a different encoding, choose the desired encoding from the Reopen using Encoding submenu. This command will only be available if the current document is unmodified.

Using the Open Selection Command

The Open Selection command lets you open a file that is referenced in the text of a document. It is particularly useful for opening include files or any document referenced by another file.

To open a file whose name is referenced in the text of a document:

1 Select the file name within the body of the document.

2 Choose Open Selection from the File menu.

If a suffix of the form ".x" follows the name, TextWrangler will automatically expand the selection to include the suffix.

TextWrangler also understands the Unix-style line number specifications that can be appended to a file name. This type of specification is generated frequently by Unix command line tools. For example, selecting the text "main.cp:210" and choosing Open Selection will open the file "main.cp" and automatically select line 210. If the file is already open, this command will simply select the designated line.

In searching for the requested file, TextWrangler will look in the following locations, in order:

- If the selected file name is surrounded by angle brackets, TextWrangler will start its search in the folder that you have specified in the Search Folders list of the File Search preferences panel.

- Otherwise, TextWrangler will look first in the same folder as the file containing the selected file name, and then in any subfolders within that folder.

- If TextWrangler cannot find the file in any of these places, it will display a Choose Folder dialog to allow you to locate the file manually.

In some cases, there may be more than one file with the same name in the various folders TextWrangler looks in. Normally, TextWrangler opens the first one it encounters, and then stops. If you want TextWrangler to find all files that match the selected name, be sure to select the Find All Matching Files option in the File Search preferences panel.

Open File by Name

If there is no selection, or there is no text display view in the front window, Open Selection becomes Open File by Name. Choosing this command brings up a dialog in which you can specify a file name, an exact path, or a file URL. You can also chose a previously-used name by clicking on the pop-up control at the right of the name field, and selecting it from the menu. (The number of names kept on the menu is controlled by the "Remember () most recent used items" option in the Application panel of the Preferences.)

TextWrangler will attempt to open the specified file as described above for Open Selection. As with Open Selection, you can specify an optional line number and can enclose the file name in angle brackets to limit the search to the default directory specified in the File Search preferences panel.

Note   When specifying a file in the Open File by Name dialog on Mac OS X, you can use the shorthand "~user/" notation to refer to an arbitrary user's home directory; for example, "~admin/Documents/bigfile.c".

If you select the Match Wildcards checkbox in the Open File by Name dialog, you can use the following wildcards in the file name:

Wildcard	Meaning
?	Any single character
*	Any number of characters
#	Any numeric character
\	Escapes one of the above; for example, \? enters a question mark. To enter a literal backslash, use \\.

An International Text Primer

Mac OS X includes extensive support for working with international text, including Unicode. If you have enabled additional text input methods in the International section of the System Preferences, you will see the Input menu on the right-hand side of the menu bar. This menu allows you to change keyboard layouts or script systems as you work.

Note   Actually, even if you have never used a non-Roman script system before, you may still have used this menu, if you have ever chosen an alternate keyboard layout such as Dvorak, or a keyboard layout for a Roman language such as French. However, since the Roman script is suitable for several languages, choosing one of these keyboard layouts still leaves you in the Roman script.

International Text in TextWrangler

As a text editor, TextWrangler supports only one font per document window, though it can display all available characters in the active font, including Unicode characters.

TextWrangler supports editing in all languages which use left-to-right text input methods. To start entering text in any supported language, choose a suitable input method from the Input menu. The icon for that method will appear in the menu bar in place of either the American flag (for the U.S. English layout) or the icon for your usual Roman keyboard layout.

If you have turned off the "Try to match keyboard with text" option in the Options dialog of the International section of the System Preferences), you may also need to select a suitable display font via the Font panel. (We recommend leaving this option on, so that TextWrangler can automatically switch to the correct input method when you change document windows.)

You can use international text throughout TextWrangler—for example, in the Find & Replace dialog. Likewise, TextWrangler will provide the necessary style information so that if you copy and paste, or drag and drop, international text into another application, that application will have enough information to handle the text correctly (assuming it is capable of doing so).

TextWrangler remembers the encoding used in a document when you save it, so the next time you open it, you will not need to choose the font. However, files that do not have this stored information, such as those you download from the Internet, may not be readable until you choose an appropriate font and script for them.

Note   When saving a multi-byte document, TextWrangler will write out a 'styl' resource in the document's resource fork. This ensures compatibility with applications which otherwise cannot properly select the document font on their own.

When performing a search, TextWrangler respects any available information about each file's encoding. If a file does not contain any information about its encoding, TextWrangler will use the default encoding set in the Text Encodings preferences panel.

Unicode

Unicode is an international standard for character encoding, which includes an extensive selection of characters from Roman, Cyrillic, Asian, Middle Eastern, and various other scripts. For more background information or complete details on Unicode, the Unicode Consortium web site is the best place to look.

```
http://www.unicode.org/
```

TextWrangler 2 includes full support for and makes extensive use of Unicode, in addition to all other OS-supported text encodings. In particular, TextWrangler internally represents all documents as Unicode, regardless of their on-disk encoding.

Saving Unicode Files

TextWrangler lets you save documents that use character set encodings other than Mac Roman, even multi-byte character sets. When saving a file, you can choose to save text composed in any script with any encoding. In addition to the standard character set encodings, TextWrangler also lets you save the files in a variety of plain Unicode files:

- UTF-8

- UTF-8, no BOM

- UTF-16

- UTF-16, no BOM

- UTF-16, byte-swapped

- UTF-16, byte-swapped, no BOM

Here are details about what each of the above options means:

- UTF-8: UTF-8 encoding is a more compact variant of Unicode that uses 8-bit tokens where possible to encode frequently used sequences from the file. (This format makes it easier to view and edit content in non-Unicode-aware editors.)

- UTF-16: UTF-16 encoding always uses 16-bit tokens.

- no BOM: When saving Unicode files, you should always include a byte-order mark (BOM) so that the reading application knows what byte order the file's data is in. For maximum compatibility, the BOM should be used whenever possible. Use one of the "no BOM" options only if there is a specific reason to do so, such as providing compatibility with software that malfunctions when a BOM is present. (For purposes of recognition when you use this option, the UTF-16 BOM is FEFF, and the UTF-8 BOM is EFBBBF.)

- byte-swapped: Since UTF-16 uses two bytes to represent each character, this leaves the question of which of the two bytes comes first—whether it is "little-endian" or "big-endian." By default, TextWrangler writes UTF-16 big-endian (the standard). By choosing one of the "byte-swapped" options, you can write little-endian files instead, which some Windows software requires.

Files saved as Unicode from TextWrangler are given a type of 'utxt'—the standard for Unicode text files on the Mac. UTF-8 files are given a type of 'TEXT' for compatibility with other applications; however, TextWrangler will also recognize such files with type 'UTF8'.

Opening Unicode Files

When opening files, TextWrangler will ordinarily determine the format of a file based on its file type and content, and automatically process Macintosh text, Unicode, and UTF-8. However, some files are structured such that TextWrangler is unable to correctly determine their format based on their type or contents. The cases that we know of are:

- UTF-8 files whose type is 'TEXT' and which lack a byte-order mark. (If a UTF-8 file is of type 'TEXT' but has a byte-order mark, it will be correctly interpreted as UTF-8.)

- Byte-swapped Unicode files which were written without a byte-order mark (usually by broken Windows software);

- Unicode files whose type is 'TEXT' (instead of the Macintosh standard 'utxt') *and* which lack a byte-order mark. (If a UTF-16 file lacks a BOM but is of type 'utxt', TextWrangler will treat it as big-endian Unicode.)

If you know that a file you are trying to open is in Unicode but it displays as gibberish on your screen, close its window without saving. Then try reopening the file, using the Open As pop-up menu in the Open dialog to specify whether to treat the file as Unicode, byte-swapped (little-endian) Unicode, or UTF-8.

When opening a malformed UTF-8 document, TextWrangler will present an alert to warn you. When such a file is encountered during multi-file searching, a warning will be logged.

Accessing FTP/SFTP Servers

TextWrangler can open files directly from, and save them to, any available FTP server. It can also open and save files directly via SFTP (**S**SH **F**ile **T**ransfer **P**rotocol). In order to access a server via SFTP, that server must be running a compatible version of sshd. (A great many machines, including Mac OS X systems for which "Remote Login" is turned on in the Sharing preferences, satisfy these criteria.)

Aside from choosing the SFTP checkbox in the Open from…/Save to… dialogs, or the FTP/SFTP Browser, opening and saving files via SFTP works just like it does when using ordinary FTP. A file opened via SFTP will appear in the Open Recent submenu with an "sftp:" URL, and you can send a "get URL" event to TextWrangler with an "sftp" URL as well.

Opening Files from FTP/SFTP Servers

To directly open a file from an FTP or SFTP server, choose Open from FTP/SFTP Server from the File menu. The following dialog appears:

Enter the name of the server, or choose a server from the Rendezvous pop-up menu, specify your user name and password in the appropriate fields and choose the "SFTP" or "Passive FTP" options if appropriate; then click the Connect button to connect to the server. You can also click the Go To button to bring up a dialog in which you can type the exact pathname of a directory to view.

Note   The Rendezvous pop-up in the dialog will show FTP servers if the "SFTP" option is turned off, or SFTP servers if this option is turned on. (Note that SSH services are only advertised by machines running Mac OS X 10.3 when "Remote Login" is turned on in the "Sharing" section of the System Preferences.)

Alternatively, you can choose an item from the Bookmarks pop-up menu to fill in a predefined server, user name, and password. Bookmarks can be set by entering the appropriate information in the Open from… or Save to… dialogs and choosing Add from the Bookmarks pop-up menu, or by using the bookmark list in the FTP Settings preferences panel.

The Passive FTP checkbox tells TextWrangler to open a connection to the FTP server in passive mode, in which the server will send TextWrangler the number of the port to use for the session. If this option is turned off, TextWrangler will specify the port number. We recommend using passive mode when possible; however, not all servers or networks support it, so if you encounter difficulties, try turning this checkbox off.

The checkbox labeled Show Files Starting with "." tells TextWrangler whether to display hidden or admin files in the chosen directory, such as .login, .forward, and .signature. Starting a file name with a period is a Unix convention, to make it invisible in most directory listings.

Once the connection is made, you can use the Open button and the directory pop-up menu to navigate through the directories, just as you would navigate through the folders on your hard drive. Click Delete to remove the file from the server.

You can use the Get Info button to reveal the size, modification date, and if applicable, file system permissions of the selected file. You can edit the file's name and click the Rename button to rename the file on the server; you can also make changes to the permissions and click the Set button to change them. (Take care not to set the permissions such that the file becomes inaccessible to you!)

TextWrangler also supports FTP and SFTP URL clippings. Dragging the window proxy icon from an editing window corresponding to a file opened from an FTP or SFTP server will create a clipping containing that file's URL. You can double-click an FTP or SFTP URL clipping in a disk browser, or drop it on TextWrangler's icon in the Finder, to have TextWrangler open the specified document or folder.

Once you have selected a file and opened it, TextWrangler displays the file in a text editing window. The status bar displays the URL of the file on the server, not the pathname of the file on your hard drive as it does for local files.

The window path pop-up menu in editing windows for files opened via this command displays the file's path on the server, rather than the local path to the temporary file. Choosing one of the items from the pop-up menu will open a new FTP/SFTP Browser window to the chosen directory.

Note When you open a file from an FTP or SFTP server, TextWrangler downloads it "raw" (in binary mode) and then performs a standard line ending conversion on the downloaded file.

Specifying Alternate Ports

TextWrangler allows you to open an FTP connection on ports other than the default (port 21). To specify an alternate port, place it on the end of the server name, separated by a colon—for example, ftp.example.com:1111.

Storing Passwords

To store FTP passwords, TextWrangler takes advantage of the system Keychain. As long as the Keychain is unlocked, whenever you enter a server and user name pair for which there is a record, the corresponding password will be filled in automatically. This will happen regardless of whether the Remember Password option in the dialog is on or off.

If the Keychain is locked, the Remember Password option will also be disabled, and you will need to retype your password every time you use the FTP dialog. You can click the key icon in the FTP dialog (above the Password field) to unlock the Keychain, or to query it for a password if the Keychain is already unlocked.

Note In the Open from FTP/SFTP Server dialog, the Remember Password option applies *only* to the last FTP session that was initiated before dismissing the FTP dialog. Also, the Auto-Connect setting in the FTP dialogs does not require that Remember Password be turned on. This is useful if you have set up public-key authentication for SSH (in which case no password is required).

WARNING As a security measure, we do not recommend saving passwords without using Keychain support. If you prefer to store such information without relying on the Keychain, you should consider using the bookmark facility of a stand-alone FTP client, or a secure information storage application such as PGP or Web Confidential.

Saving Files to FTP/SFTP Servers

After you have edited a file opened from an FTP or SFTP server, pressing Command-S or choosing Save from the File menu saves the new version back to the server. If you want to save the file in a different directory or under another name, choose Save to FTP/SFTP Server to open a dialog (shown below) that works much like the Open from FTP/SFTP Server dialog. Like a standard Save dialog for saving a local file, it includes a field for the name of the file so that you can name it before saving.
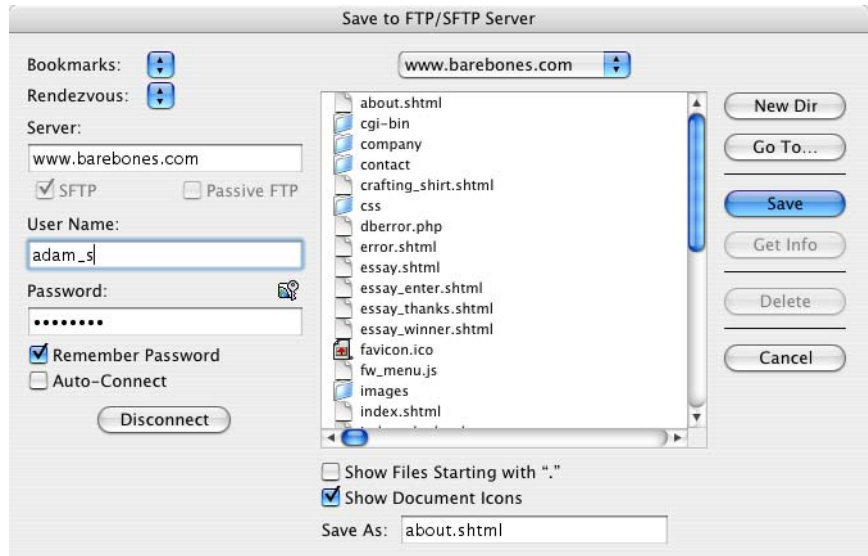
Note When you save a file to an FTP or SFTP server using either Save or Save to FTP/SFTP Server, and the file has Unix (LF) or Windows (CR+LF) line endings, TextWrangler uploads the file in binary mode, preserving its line endings exactly as they are on your local machine. However, if the file has Macintosh (CR) line endings, it is uploaded in text mode so that the server can convert the line endings as appropriate.

Finally, you can use Save a Copy to FTP/SFTP Server to upload a copy of your current file to an FTP server while keeping your local file open. This is especially useful when you maintain web site content on your local hard drive and only need to upload changes made in one or two files to the server.

Using FTP/SFTP Browsers

In addition to the Open from FTP/SFTP Server command for opening individual files, TextWrangler offers FTP/SFTP Browser windows. FTP/SFTP browsers keep a connection open to the server for as long as the browser window is open, and also allow you to select and open multiple files or directories from the server.

To open a browser, choose New FTP/SFTP Browser from the New submenu of the File menu. Enter your server address, user name, and password, or choose a server from the Rendezvous pop-up menu, and if appropriate, select the "SFTP" or "Passive FTP" options. Then, connect to the specified server by pressing the Connect button, or, while the Password field is active, hit the Return key. Alternatively, you can choose a bookmark from the Bookmarks menu.

To open a file from the browser listing, double-click it, or select it and click the Open button. For editing purposes, files opened from a browser behave exactly like those opened with the Open from FTP/SFTP Server command. To refresh the contents of the listing, click the button in the lower right with the circular arrow icon.

Opening Files from the Command Line

You can use the "edit" command line tool to open files in TextWrangler via the Unix command line. The first time you run TextWrangler after installation, it offers to install the command line tools for you. If you choose not to do so, you can use the "Install Command Line Tools" button in the Tools preferences panel to install the tool at a later time.

To open a file in TextWrangler from the command line, type

```
edit filename
```

where *filename* is the name of the file to be opened. To launch TextWrangler without opening a file (or activate it, if it is already running), type

```
edit –l
```

In addition to files, you can also specify FTP or SFTP URLs to files or directories, to have TextWrangler open the specified files, or an FTP/SFTP Browser for each directory. You will be prompted to enter passwords if necessary.

You can also pipe stdin to the "edit" tool, and it will open in a new untitled window in TextWrangler: for example,

```
ls -la | edit
```

If you just type

```
edit
```

with no parameters, the tool will accept stdin from the terminal; type Control-D (end-of-file) to terminate and send it to TextWrangler.

The complete command line syntax for the "edit" tool is

```
edit [ –bchlpuvVw --clean --resume --view-top ]
 [ -e <encoding_name> ] [ -t <string> ]  [ +<n> ]
 [ file (or) <S/FTP URL> ... ]
```

See the man page for the "edit" tool ('man edit') for a complete description of the available switches and options.

Using Stationery

Like most Macintosh applications, TextWrangler supports stationery pads. A stationery pad is a template document that, when opened, results in a new, untitled document with the content from the stationery file. In other words, you do not edit the stationery document itself; you use it as a starting point for a new document.

To create a stationery pad, click the Save As Stationery checkbox when saving the file from TextWrangler. Alternatively, you can change any document into a stationery pad in the Finder by clicking the Stationery Pad checkbox in the document's Get Info window.

You can create new documents from a stationery pad in any of these ways:

- Open the pad the same way you would open any other document.

- Choose New With Stationery from the File menu, and select the desired stationery pad from the contents of the Stationery folder (inside TextWrangler's application support folder).

- Use TextWrangler's Stationery List, which is available from the Window menu. The Stationery List is a palette that displays all the stationery pads you have placed inside the Stationery folder of TextWrangler's application support folder. You can create a new document from any of these pads by double-clicking them in this list.

To assign a keyboard shortcut to a stationery pad, select the pad in the Stationery List window; then, click the Set Key button, type the desired key in the Set Key dialog and click OK.

Manually Sorting the Stationery

By default, items in the Stationery List are displayed in alphabetical order. However, you can force them to appear in any desired order by including any two characters followed by a right parenthesis at the beginning of their names. For example, "00)Perl scripts" would sort before "01)CGI scripts". For such files, the first three characters are not displayed in TextWrangler. You can also insert a divider by including an empty folder whose name ends with the string "-***". (The folder can be named anything, so it sorts where you want it.) These naming conventions are the same as those used by the utilities FinderPop and OtherMenu.

Hex Dump for Files and Documents

Choose the Hex Dump File command to generate a hex dump representation from a file that you choose. You may also choose Hex Dump Front Window to generate a hex dump representation of the frontmost document as it exists in memory.

You should bear in mind that the result of performing the Hex Dump command against a disk file may differ from the result obtained by using it against an open document, since when a document is open in memory, even without any explicit edits being made, line-break translation and possibly character set encoding conversions have taken place.

Making Backups

TextWrangler can automatically make a backup copy of a document before saving it. You can also manually take a snapshot of a document at any time to make it easier to revert to previous versions.

TextWrangler names backup files with the same name as the original file and appends the date and a sequence number. For example, the first backup made on June 10, 2004 of "My Text File" would be named "My Text File (6-10-04)-1". The name of the original file may be shortened if adding the date and sequence number would make the name of the backup file longer than 255 characters.

The Text Files: Saving panel of the Preferences window includes an option that allows you to specify the default backup options for new files and for files that do not have state information. (See Chapter 10 for more details.) You can override this default on a file-by-file basis, or manually back up a file at any time.

A document must be saved and named before you can set its backup options or make a backup manually.

If an error occurs during the backup process, TextWrangler will now report a descriptive error, rather than just an OS result code. (The usual cause of such an error is that the defined backup directory has been deleted.)

Automatic Backups

To make automatic backups of a document, follow these steps:

1 Choose the Backup Options command from the File menu.

TextWrangler opens the Backup Options dialog box.
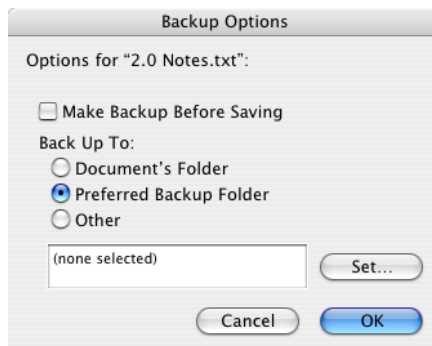
2 Specify the folder in which you want TextWrangler to save the backups.

You can choose the document folder, the preferred backup folder (set in TextWrangler's preferences), or some other folder. In the latter case, click Set or drag a folder icon from the Finder to the path box.

3 Select the Make Backup Before Saving option.

TextWrangler will make a backup before you save the file. The backup will contain the previously saved version of the file.

4 Click OK.

Manual Backups

To make backups manually, follow the same steps as above, but do not select the Make Backup Before Saving option. To make a backup at any point, choose the Make Backup Now command from the File menu.

TextWrangler opens a standard directory dialog box so that you can change the location and the name of the backup file. The default folder is the folder you specified in the Backup Options dialog box, and the default name is the standard backup name as described at the beginning of this section.

Printing

To print a document, choose the Print command from the File menu. TextWrangler opens the standard print sheet for your currently selected printer.
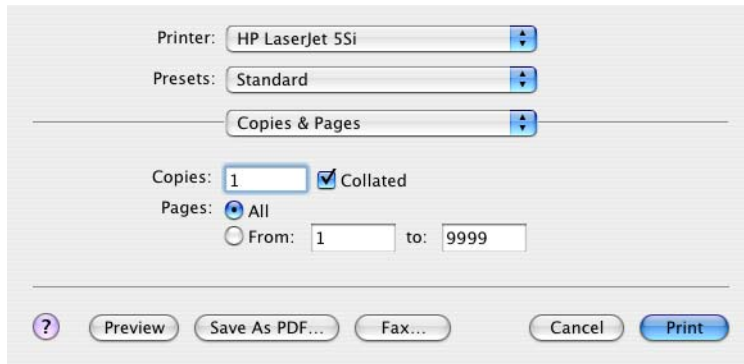
The TextWrangler-specific printing options are on a "page" of their own, where they can be selected by a pop-up menu at the top of the dialog.

To print one copy of the active document without displaying the print sheet, choose the Print One Copy command from the File menu. (Hold down the Shift key while the menu is visible to access this command.)

Text Printing Options

You can access all of TextWrangler's application-specific printing options by choosing the Text Printing "page" on the pop-up menu in the center of the print sheet.
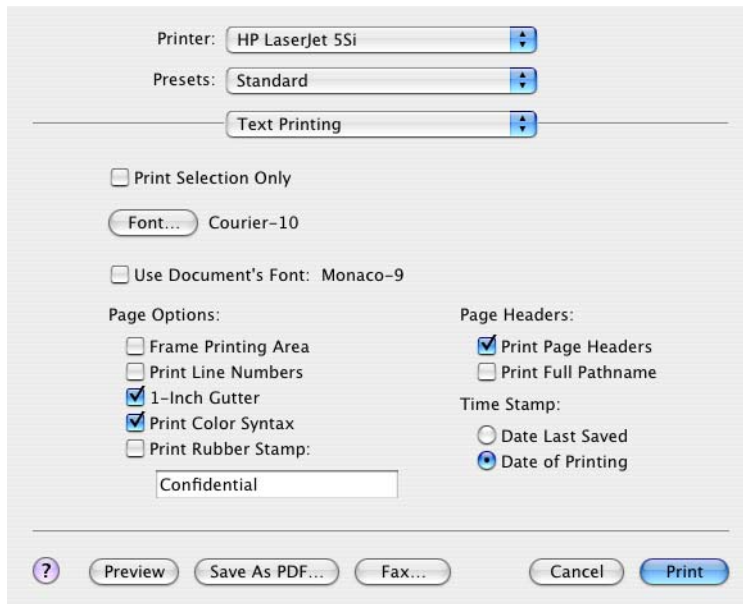
Note   You can set defaults for most printing options in the Text Printing preferences panel.

Print Selection Only
When this option is selected, TextWrangler prints only the selected text.

Font Button
Click this button to open a dialog box that lets you set the font, size, style, and tab settings to use while printing.

Use Document's Font
When this option is selected, TextWrangler uses the document's font when printing.

Page Options:
These options control how the printed pages will be laid out. You can also set these options for the current document by choosing the Printing Options command in the Edit menu. You can set default printing options in the Text Printing preferences panel.

Frame Printing Area
When this option is selected, TextWrangler draws a frame around the printed text.

Print Line Numbers
When this option is selected, TextWrangler prints line numbers along the left edge of the paper.

1-Inch Gutter
When this option is selected, TextWrangler leaves a one-inch margin along the left edge of the paper. Use this option if you usually put your pages in three-ring binders.

Print Color Syntax
When this option is selected, TextWrangler will print the document in color.

Print Rubber Stamp
When this option is selected, TextWrangler prints a message in gray diagonally across the page. Use the pop-up menu to choose a font, and type the message in the text field. TextWrangler chooses the right-size font to print the message.

If your printer supports grayscale printing, TextWrangler prints the rubber stamp in gray, otherwise it is printed in outline style.

Note   This feature is not supported by all printer drivers.

Print Page Headers
When this option is selected, TextWrangler prints the page number, the name of the file, and the time and date printed in a header at the top of each page.

Print Full Pathname
When this option is selected, TextWrangler prints the full pathname of the file in the header.

Time Stamp
The Time Stamp options let you choose whether the date that appears in the header is the date that the file was last modified or the date that the file was printed.

4

Editing Text with TextWrangler

This chapter describes the basics of editing text with TextWrangler, wrapping text, text manipulations, and file comparison.

In this chapter

Basic Editing

TextWrangler behaves like most Macintosh word processors and text editors. Characters that you type in an active window appear at the insertion point, a vertical blinking bar. You can click and drag the mouse to select several characters or words, and the selected text is highlighted in the system highlight color.
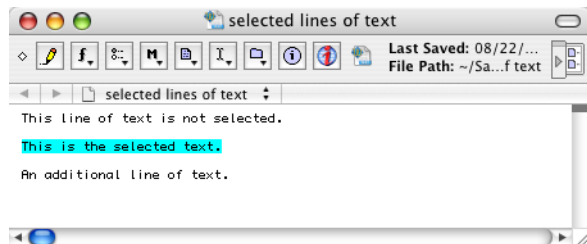
If you select some text and then type, whatever you type replaces the selected text.

To delete selected text, press the Delete key or choose Clear from the Edit menu. If you have a keyboard with a numeric keypad on it, you can press the Clear key on the keypad to delete the selected text.

In addition to clicking and dragging, you can use three selection commands in the Edit menu to select text.

To select…	Choose this from the Edit menu…
Line containing insertion point	Select Line
Paragraph containing insertion point	Select Paragraph
All text	Select All

You can then cut, copy, or perform any other action that affects the selected text.

Note TextWrangler defines a paragraph as a block of text surrounded by blank lines (lines containing no characters other than tabs or spaces). The beginning and end of the document also mark the beginning and end of paragraphs.

Moving Text

To move text from one place to another, follow these steps:

1 **Select the text you want to move.**

2 **Choose Cut from the Edit menu.**

   TextWrangler removes the text from the window and stores it on the clipboard.

3 **Use the scroll bars to move to the new place for the text if necessary; then click to set the insertion point where the text is to be inserted.**

4 Choose Paste from the Edit menu.

You can paste the contents of the clipboard as many times as you want in any TextWrangler window or in any other application.

Pasting inserts the text stored on the clipboard at the insertion point. If there is a selection, pasting replaces the selection with the contents of the clipboard.

To place text on the clipboard without deleting it, choose Copy from the Edit menu.

Tip   To add selected text to the existing contents of the clipboard, hold down the Shift key as you choose the Cut or Copy command. When you hold down the Shift key, TextWrangler changes these commands to Cut & Append and Copy & Append.

Multiple Clipboards

TextWrangler supports six separate clipboards. Each time you use the Cut or Copy command, TextWrangler automatically switches to the next clipboard (wrapping back around to the first clipboard after the sixth). This way, the last six things you copied or cut are always available for pasting—sort of a "clipboard history."

By default, the Paste command pastes text from the most recently used clipboard, so if you do nothing special, TextWrangler works just like any other Macintosh program. However, by using the Previous Clipboard command in the Edit menu you can access the previous clipboard contents. Next Clipboard moves forward through the clipboard history. There are also buttons in the Clipboard window (below) that let you move back and forth through the clipboards.
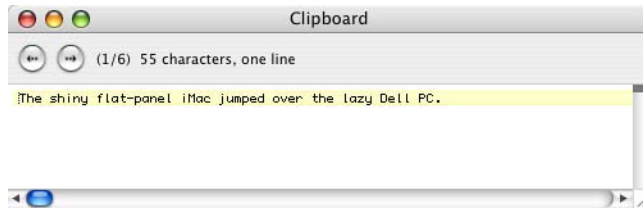
Once you have selected a clipboard using one of these methods, the next Cut, Copy, or Paste command will use the clipboard you chose. (Subsequent Cut or Copy commands will advance to the next clipboard; Paste never advances automatically.)

Holding down the Shift key changes the Paste command to Paste Previous Clipboard, or you can use the key equivalent Command-Shift-V. This command, enabled whenever the last operation was a paste and the previous clipboard is non-empty, replaces the pasted text with the contents of the previous clipboard. The previous clipboard becomes current and will be used for any further paste operations; repeated applications of the command cycle backward through the available clipboards.

Note   For compatibility with international text content, the Clipboard window displays text in the font (and font size) that it was put on the clipboard with. Changing the display font in the Clipboard window *does not* change the font of the underlying data.

Drag and Drop

Another way to move text from one place to another is by "drag and drop." If you drag and drop text from one window to another, TextWrangler copies the text to the target window without removing it from the original window.

In addition, you can drag and drop an item from the Finder onto an editing window in TextWrangler. If the item is a text file, the file's contents are inserted. If the item is a folder, a listing of the item's contents is inserted. If you hold down the Command key while dragging a folder, the path of the item is inserted instead.

Multiple Undo

TextWrangler provides the ability to undo multiple edits, one action at a time. The number of edits that may be undone is limited only by available memory. The practical limitation is determined by the extent of the edits and the amount of free memory.

TextWrangler also supports multiple Redos. If you have not made any changes after performing an Undo, you can redo each action, in order, by choosing that Redo command from the Edit menu or typing Command-Shift-Z. However, once you perform a new action, you cannot redo any actions that you undid before you made that change.

You can select the Clear Undo History from the Edit menu, or by typing Command-Control-Z, to clear the undo history for the current editing window. This command can be useful if you have performed many operations on a file and wish to recover memory stored by Undo state information (in the rare event that should become necessary). You can also script this operation via the "clear undo history" scripting command (see the scripting dictionary for details).

Window Anatomy

TextWrangler text windows have the same controls you are familiar with from other Macintosh applications (for example, text windows are resizable and zoomable, and have both vertical and horizontal scroll bars). Some additional elements which may be less familiar are the status bar, the split bar, the navigation bar, and the documents drawer.
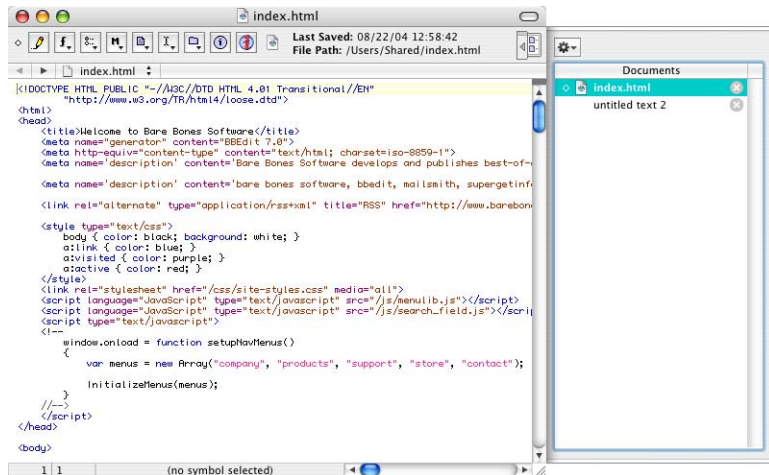
IMPORTANT You can choose whether TextWrangler should display all new and opened documents in the frontmost text window, or open each document into a new text window, by setting the New & Opened Documents option in the Documents preferences panel (see page 170). Similarly, you can control whether documents opened from other applications (such as the Finder) should open in the front text window or as separate text windows.

The Status Bar

The status bar is a panel at the top of editing windows containing buttons and pop-up menus that let you work with the text in the window. You can toggle display of the status bar by using the control in the top-right corner of the window, or by choosing Hide Status Bar/Show Status Bar in the View menu.
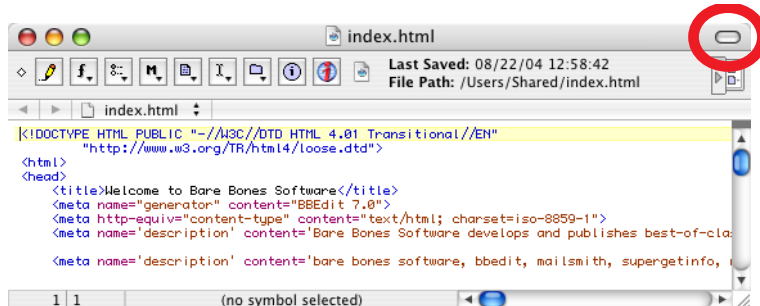
If the window contains a document that has been saved to disk, the status bar contains the full path to the file and the last time the file was changed. If the file has not been saved to disk, the status bar displays "(New Document)" instead of the file name.

Note Windows in which the text view status bar is not adjacent to the window title bar (for example, disk browsers and search results) do not have a status bar control, but do honor the global status bar preference, and you can use the Text Options sheet to show and hide the status bar on a per-window basis.

Tip You can Option-click the status bar control to show or hide the status bar for all text windows.

The icons on the status bar are indicators, buttons, and pop-up menus that give you quick access to commonly used functions. The following table explains each icon.

Icon	Meaning
◆ ◇	A solid diamond indicates that the document has been modified. A hollow diamond means only the *state* of the document (window position, selection range, scrolling position, and so on) has changed.
✎	The pencil icon indicates that the document can be modified. If the pencil has a slash across it, the document cannot be modified because the file is read-only or the disk is locked. If the file is not on a locked disk, you can click the pencil icon to toggle the document's editability.
f.	The Function pop-up menu gives quick access to routines and functions in languages that TextWrangler can parse. In HTML documents this menu lists the contents of the TITLE tag (if any), all named anchors in your document (that is, those defined with <A NAME="…">), all level 1–6 headings, any tags that have ID attributes, and any "BBmark" indicators.
⸬	The Text Options pop-up menu contains commands such as Soft Wrap Text and Show Invisibles that let you control how the text appears in the window.
M.	The Mark pop-up menu contains commands such as Set Marker and Find & Mark that let you set or mark specific locations in the file; when present, these marked locations will be listed on the pop-up menu below the commands.
🖹	The File pop-up menu contains commands that let you set line-break options, specify what state information is saved, and set Unicode file options if applicable.
I.	The Insert pop-up menu contains commands that let you insert the contents of files, folder listings, and page breaks.
📁	The Path pop-up menu displays the list of folders that contain the document. You can use this menu to open any of the folders along the path in the Finder.
①	The Info button displays a dialog box that lists the number of characters, words, lines, and pages in the document. Clicking this button is the same as choosing the Get Info command from the Window menu.

Icon	Meaning
	The Super Get Info button asks Super Get Info to display information regarding the current document. This button is available only if you have Super Get Info installed on your computer. (Super Get Info is a Mac OS X file info utility from Bare Bones Software; see our web site for more details.)
	The document proxy icon represents the current document. Clicking this icon is the same as choosing Reveal in Finder from the View menu: it opens a Finder window that contains the document. You can also drag the document proxy icon to any other application, or you can drag it to the Trash (which is the same as choosing Close & Delete from the File menu).

Key Equivalents for Status Bar Menu Items

You can assign keyboard shortcuts to items on the Text Options, Markers, and Line Breaks pop-up menus by choosing Set Menu Keys on the TextWrangler menu.

The Function Pop-Up Menu

The Function pop-up menu lists the functions defined in this file for a programming language source file or, for an HTML document, the tags described in the description of the menu in the table above.

The following indicators appear in the pop-up menu to show the type of function.

Indicator	Meaning
•	The function containing the insertion point
†	C/C++ typedef
◊	C/C++ "#pragma mark" directive
italic name	C/C++ function prototype
1-6	Heading level (in HTML files)
tag name	Tag name for the indicated name or ID attribute value (in HTML files)

The Split Bar

Every text window and every browser text pane has a split bar, a small black bar above the scroll bar, that lets you split it into two active view regions. Splitting a text pane lets you view and edit a document's content in two places at the same time. Each region is independently scrollable.

Note Scrolling the non-active split region does not automatically change view focus.

To split the text pane, simply drag the split bar down and let go.
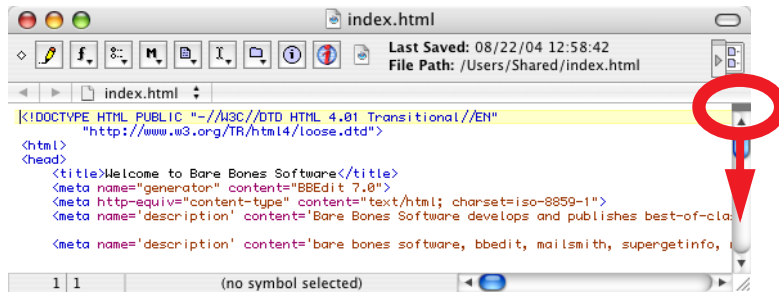
To collapse the text pane back down to a single region, drag the split bar (starting from anywhere along its length, not just at its right end) back up to its original position.

Tip Double-clicking the split bar unsplits a split text pane or restores the last-used split position. If the text pane has never been split, it will be split 50-50. To force a 50-50 split for a previously split text pane, Option-double-click the split bar when it is in its original position.

The Navigation Bar

The navigation bar is a panel at the top of a text window which provides controls for selecting the active document. Click the Previous or Next buttons to move to the previous or next document in the window, or choose Previous Document/Next Document from the View menu. You can also choose a specific document from the adjacent pop-up menu to make it frontmost.
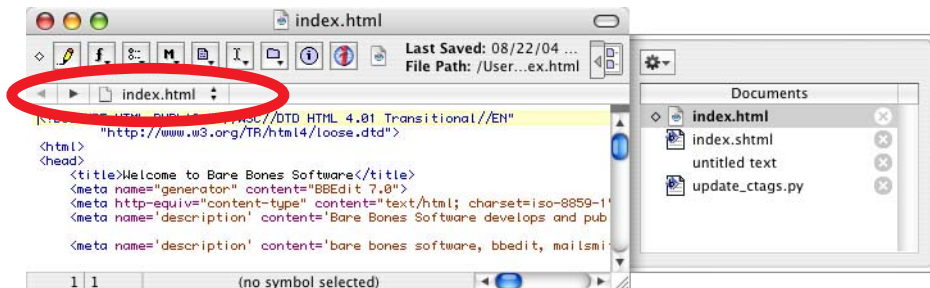
To hide the navigation bar, choose Hide Navigation Bar in the View menu, or turn off the Show Navigation Bar option in the Text Status Display preferences panel.

The Documents Drawer

The documents drawer is a panel which TextWrangler optionally displays at the side of a text window that lists all the documents current open in the window. Click the drawer toggle control at the right-hand edge of the status bar to show or hide the drawer, or choose Show Documents Drawer/Hide Documents Drawer in the View menu. Click any document's name in the drawer to make that document frontmost in the text window.
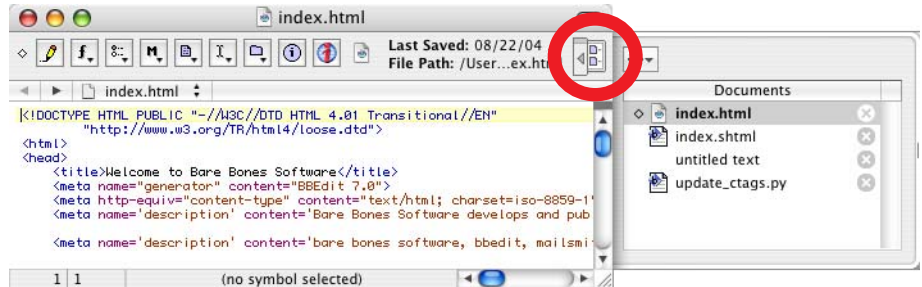
You can control whether TextWrangler should display documents in the drawer by name, or the order in which they were opened, via the "Sort Windows by:" option in the Windows preferences panel. This preference option also controls the order in which documents appear in the navigation bar's pop-up menu and the Documents submenus of the Window menu. (See "Windows Preferences" on page 191.)

The document drawer also contains an action menu, much like the Finder's, which you can use to move documents out into additional windows.
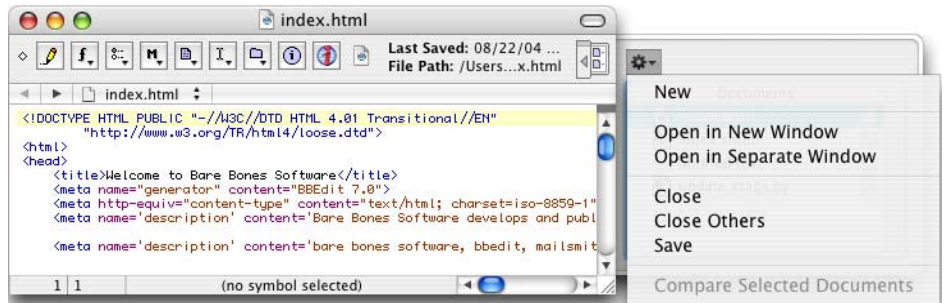
To close a document, you can choose Close Document from the File menu, click on the close box next to its name in the drawer, select it in the drawer and apply the Close command from the drawer's action menu, or Control-click on it in the drawer and select Close from the contextual menu. You can also choose the Close Others command from the action menu, or the contextual menu, to close all documents except the selected document.

To open an existing file into a text window, you can choose Open from the File menu, or drag and drop the file from the Finder into the window's document drawer.

To create a new document, you can choose New Text Document from the New submenu of the File menu, or use the New command from the drawer's action menu.

To move a document into its own text window, select it in the drawer and choose Open in Separate Window from the Action menu, or Control-click and choose this command from the contextual menu. You can also select multiple documents, and use Open in Separate Window to open each document into a separate text window, or Open in New Window to create a new text window containing all the selected documents.

To move a document from one text window to another, drag its name out of the first text window's documents drawer into the second text window's document drawer. You can select and move multiple documents at once.

Dragging a document's name from the documents drawer has the same effect as dragging its proxy icon in the status bar.

Tip   Under Mac OS X 10.4 and later, you can Option-click on the drawer toggle control to show or hide the documents drawer for all text windows

The View Menu

This menu contains commands which you can use to toggle the display of navigational elements in text windows, to select documents, and to get info on a document or file.

Hide/Show Status Bar

Choose this command to hide or show the status bar in the frontmost text window.

Hide/Show Navigation Bar

Choose this command to hide or show the navigation bar in the frontmost text window.

Hide/Show Documents Drawer

Choose this command to hide or show the documents drawer for the frontmost text window.

Previous Document/Next Document

You may use these commands to select documents within the frontmost text window in either their display order or most-recently viewed order. If the frontmost text window contains only one document, these commands will be disabled.

Open in Separate Window

Choose this command to open the active document of the frontmost text window into its own text window. If the frontmost text window contains only one document, this command will be disabled.

Get Info

Choose this command to display a dialog box that lists the number of characters, words, line, and pages in the selected text and in the document. Using this command is the same as clicking the info button in the status bar.
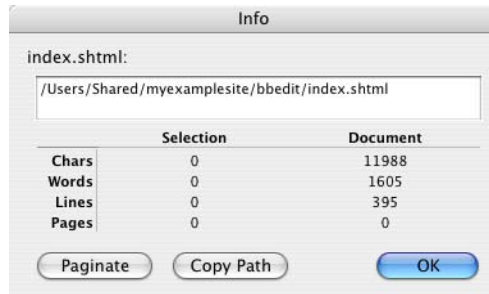
To find out how many pages the document will take to print, click the Paginate button. To put the full path to the file on the clipboard, click the Copy Path button.

Reveal in Finder

Choose this command to open a Finder window which will display the active document's file. If the active document is not associated with a file, this command will be disabled. Using this command is the same as clicking (without dragging) the document proxy icon in the status bar.

If the selected text in a document is the name of a file, hold down the Option key as you open the File menu and choose the Reveal Selection command to have BBEdit open a Finder window which will display that file.

Open in Super Get Info

This command provides integration with Super Get Info, Bare Bones Software's Mac OS X file info utility. If you choose this command, BBEdit will ask Super Get Info to open an info window for the active document's file.

If there is no file associated with the active document, or if you have not installed Super Get Info, this command will be disabled.

Super Get Info is a utility designed to serve as a supplement for the Finder's Show Info command. Super Get Info allows you to open more than one info window at a time; view and edit the Macintosh type and creator codes associated with a file; view and edit the Unix owner, group, and permission settings associated with a file or folder; and much more. For more information, or to download a free demo version, visit our web site.

```
http://www.barebones.com/products/supergetinfo.html
```

Cursor Movement and Text Selection

TextWrangler gives you several ways to move the insertion point and change the selection. You can click and drag using normal Macintosh text selection techniques or you can use various keys on the keyboard.

Clicking and Dragging

You can select text in an editing window in the normal Macintosh fashion, by clicking and dragging. Holding down the Shift key while clicking or dragging extends the selection.

	No Modifier	Shift
Click	Move insertion point	Extend selection
Double-click	Select word	Extend selection to word
Triple-click	Select line	

Triple-clicking is the same as clicking in a line and then choosing the Select Line command from the Edit menu.

Holding the Command and Option keys as you click or double-click triggers special actions:

	Option	Command	Command + Option
Click		Open URL	
Double-click	Look up selected word in programming reference		Select word and search for next occurrence of the word

TextWrangler optionally allows you to select entire lines by clicking in the left margin of an editing window. (If you have line numbers displayed, via the Show Line Numbers option in the Status Bar preferences panel, you can click in the line number as well.) You can click and drag to select multiple lines, double-click to select an entire paragraph, or double-click and drag to select a range of paragraphs. A checkbox in the Editing: General preferences panel, labeled Allow Single-Click Line Selection, controls this behavior. If the checkbox is turned off, clicking in the left margin simply moves the insertion point to the beginning of the clicked line.

Arrow Keys

You can use the arrow keys to move the insertion point right, left, up, and down, and augment these movements with the Command, Option, and Control keys:

	No Modifier	Option	Command	Control
Up	Up one line	Up one screen	Start of document	(scroll view up)
Down	Down one line	Down one screen	End of document	(scroll view down)
Left	Left one character	Left one word	Start of line	(scroll view left)
Right	Right one character	Right one word	End of line	(scroll view right)

Holding down the Shift key extends the selection. For example, pressing Shift-Option-Right Arrow selects the word to the right of the insertion point.

If you are used to a word processor or text editor that lets you use Command-key combinations to page through your document, you may want to swap the meaning of the Option and Command keys:

1 **Open the Preferences window (by choosing Preferences from the TextWrangler menu).**

2 **Select the Editing: Keyboard preferences panel in the list on the left in the Preferences window.**

3 **Under the heading "Exchange Command and Option Key Behavior," select Horizontally, Vertically, or both, as you prefer.**

When active, these settings change the sense of the up- and down-arrow keys as follows:

	No Modifier	Option	Command
Up	Up one line	Start of document	Up one screen
Down	Down one line	End of document	Down one screen
Left	Left one character	Start of line	Left one word
Right	Right one character	End of line	Right one word

When the Shift key is held down, the arrow keys behave as described in the table above, except that the selection range is extended to include the new placement of the insertion point. (This is the same effect as typing the arrow-key combination and then holding down the Shift key and clicking at the original placement of the insertion point, or at the end of the original selection range.)

Rectangular Selections

By holding down the Option key as you drag, or holding down the Shift and Option keys while clicking, you can select all text lying within a specified rectangular area. You can then perform all of the normal editing operations on this "rectangular selection," such as Cut, Copy, Paste, or drag and drop, as well as text transformations such as Change Case, Shift Left, Shift Right, Entab, Detab, Increase Quote Level, Decrease Quote Level, Strip Quotes, and Zap Gremlins.

IMPORTANT   Rectangular selection and soft wrapping are mutually incompatible. When soft wrapping is enabled, dragging the mouse performs normal (non-rectangular) selection even if the Option key is held down; when there is a rectangular selection, the Soft Wrap Text option is unavailable in the Text Options pop-up menu and sheet.

Working with Rectangular Selections

Commonly, while working with text, you will be performing actions on a line-by-line basis; for example, when making a selection, you will start by selecting the contents of one line before moving on to the next. However, if you need to deal with tabular data, it can be useful to think in terms of rectangles or blocks of text that include parts of several lines. This is where you can make use of TextWrangler's ability to manipulate rectangular selections.

For example, consider you have the document shown below, and you want to move only the bottom left column (the one that says "This text goes in the middle") and move it in between the top left and top right columns. To do this using standard selection methods, you would have to perform five separate cut-and-paste operations. However, by using rectangular selections, you can move the whole column in one operation.
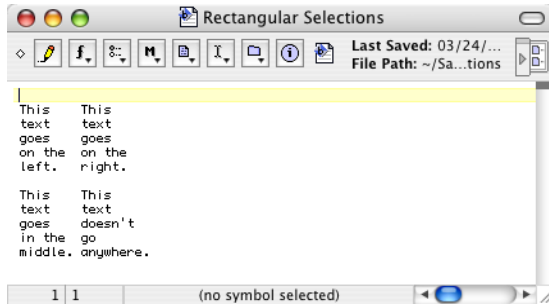
To start, hold down the Option key while dragging over the bottom left column, until you get a selection that looks like this:
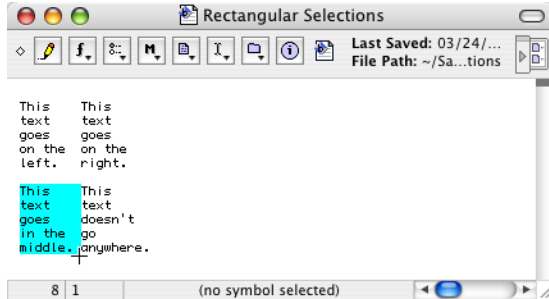
Choose Cut from the Edit menu (or press Command-X) to cut the selected text out of the document and place it on the Clipboard.
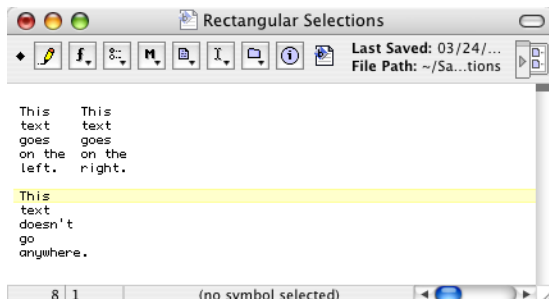
Next, you must make a selection to tell TextWrangler where you want to paste the text you just cut. To do this, position the arrow pointer just to the left of the top right column, press and hold the Option key, press the mouse button, and drag straight down until you have a very thin vertical selection just to the left of the whole column, as shown below.
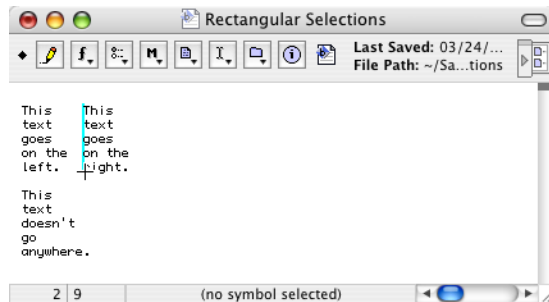
Now, paste the text you previously cut back in.
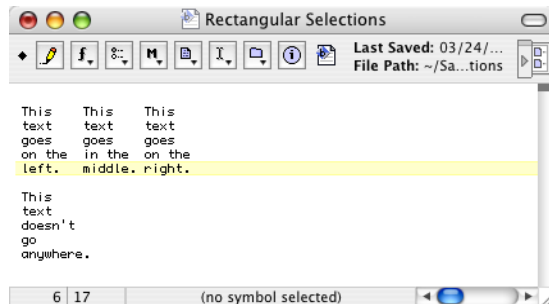
Alternatively, you can use the Paste Column command from the Edit menu to force the current contents of the Clipboard to be pasted in columnar fashion, starting at the current insertion point.

Note   Some word processors also provide support for rectangular selections which works a little differently than TextWrangler's, so you may wish to keep this difference in mind. Typically, when you copy a rectangular selection of text to the clipboard in these programs, they handle that piece of text different than text copied from a line-by-line selection. Then, when you paste, the text will be entered in a block, even when you have not made a rectangular selection to paste into. TextWrangler does not do this. Instead, when you copy a rectangular selection to the clipboard, TextWrangler turns the selection into a series of individual lines, which is why you must make a rectangular selection before pasting, so TextWrangler will know it should paste the text in block fashion. Though this method does require an extra step, it is more flexible, because you can select a set of lines and then paste it as a block, or vice versa.

Scrolling the View

When holding down the Control key, the arrow keys will scroll document windows without moving the insertion point.

Accelerated Scrolling

When clicking the arrows in a scroll bar, you can use the Command and Option keys to accelerate the scrolling. These shortcuts also apply if you use a mouse with a built-in scroll wheel.

Modifier	Scroll Speed
none	Normal
Command	2x accelerated
Option	3x accelerated
Command+Option	6x accelerated

The Delete Key

The Delete key deletes the character to the left of the insertion point. If you have selected text, the Delete key deletes all the text in the selection. You can use the Command and Option keys to modify the way the Delete key works:

Modifier	Action
none	Deletes character to the left of the insertion point
Option	Deletes to the beginning of the word to the left of the insertion point
Command	Deletes to the beginning of the line
Command+Option	Deletes to the beginning of the document

Holding down the Shift key with the Delete key makes the Delete key work the same way as the Forward Delete key on extended keyboards. This feature is particularly useful on PowerBooks.

To enable this feature:

1 **Open the Preferences window (by choosing Preferences from the TextWrangler menu).**

2 **Select Text Editing from the list on the left in the Preferences window.**

3 **Select Enable Shift-Delete for Forward Delete.**

Note   If you have activated Horizontally for Exchange Command and Option Key Behavior as described in the previous section, the effects of Command and Option shown in the table above will be reversed accordingly.

The Numeric Keypad

Most Macintosh keyboards have a numeric keypad on the right side. Normally, you use the keys on the keypad to enter numbers. If you prefer, you can use the numeric keypad to move the insertion point:

1 **Open the Preferences window (by choosing Preferences from the TextWrangler menu).**

2 **Select Text Editing from the list on the left in the Preferences window.**

3 **Mark the Use Numeric Keypad for Cursor Movement checkbox.**

start of line **7**	up **8**	Scroll up **9**
left **4**	show selection **5**	Right **6**
end of line **1**	down **2**	Scroll down **3**

You can use the Shift key with the keys on the numeric keypad to extend a selection. You can use the Command and Option keys with the 2, 4, 6, and 8 keys as you would the arrow keys.

To toggle the behavior of the keypad between moving the cursor and entering numbers, hold down the Option key and press the Clear key in the upper-left corner of the keypad. (This key is also labeled Num Lock on some keyboards.)

Go To Line Command

To move the insertion point to a specific line, use the Go To Line command in the Search menu. When you choose this command, TextWrangler opens a Go To Line sheet in the frontmost document.

Type the number of the line you want to move to and click Go To.

Function Keys

If your keyboard has function keys, you can use the following key equivalents for cutting and pasting, to scroll, and to move the insertion point.

	No Modifier	**Option**	**Command**	**Shift**
F1	Undo			Redo
F2	Cut			Cut & Append
F3	Copy			Copy & Append
F4	Paste			
del	forward delete	delete to end of word	delete to end of line	
Home	scroll to top of document		move insertion point to start of document	
End	scroll to end of document		move insertion point to end of document	
Pg Up	scroll page up			
Pg Dn	scroll page down			

Note Holding down the Command and Option keys as you press the forward delete key deletes to the end of the document.

Text Options

You can use the Text Options command to change the way TextWrangler edits text and the way it displays text and additional elements in its windows.

The controls on the Text Options sheet are divided into two parts: the Editing options on the left control the way TextWrangler behaves while you type, and the Display options on the right control the appearance of the TextWrangler window.
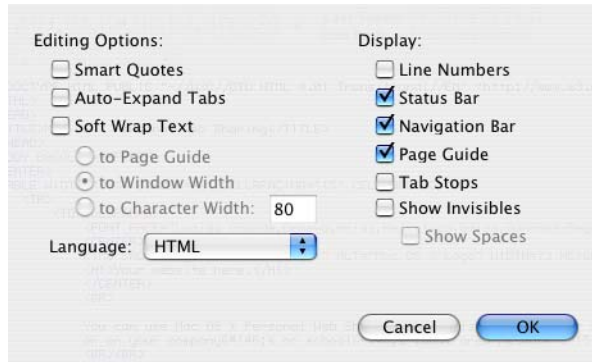
You can also change many of the text options through the Text Options pop-up menu (left) in the status bar.

Changes you make in the Text Options dialog box affect only the active document or window. To set options which will apply to all text windows you open, use the Editor Defaults and Text Status Display preferences panels.

Editing Options

These options control the way TextWrangler behaves as you type text in the active document window. Changes you make here affect only that document. To change the default editing options for documents that you will open in the future, use the Editor Defaults preferences panel.

Smart Quotes

When this option is on, TextWrangler automatically replaces straight quotes (" ') with typographer's quotes (" " ' '). If you need to type a straight quote when this option is selected (or to type a typographer's quote when the option is not selected), hold down the Control key as you type the " or ' key.

Note   Smart quote marks should not be used in HTML documents, since they will not display correctly; you must use straight quotes, or entity codes, instead. We recommend leaving this option turned off if you are editing HTML content or program code.

Auto-Expand Tabs

When this option is selected, TextWrangler inserts an appropriate number of spaces when you press Tab, rather than inserting a tab character.

Soft Wrap Text

When this option is selected, TextWrangler soft-wraps the text in the file to the right margin that you choose: the Page Guide, the window width, or a specific number of characters. The Page Guide is an arbitrary visual boundary whose width you can set in the Window Status Display panel of the Preferences.

See "How TextWrangler Wraps Text" later in this chapter to learn how wrapping works in TextWrangler.

Language

The Language menu lets you specify which source code language the file uses. The file's language setting affects how TextWrangler performs syntax coloring and parses function names for the function pop-up menu. TextWrangler generally determines the file's language from its filename extension, using the mapping table in the Languages panel of the Preferences window. For example, .cp files are C++, and .m files are Objective-C. You can use this menu to override those settings for a specific file. To quickly check the language for a file, click the Text Options pop-up menu in the status bar and look at the Languages item.

Display Options

These options determine which controls appear in the frontmost text window, regardless of whether that window contains one or more documents. Changes you make here affect only that window. To change the display characteristics for text windows that you will open in the future, use the Text Status Display preferences panel.

Line Numbers

This option displays line numbers along the left edge of the window.

Status Bar

This option shows or hides the status bar in the window.

Navigation Bar

This option shows or hides the navigation bar in the window.

Page Guide

This option shows or hides the page guide in the window.

Tab Stops

This option shows or hides tab stop indicators in the window.

Show Invisibles

This option shows or hides non-printing characters in the window. Select this option when you want to see line breaks, tabs, and "gremlins" (other invisible characters). TextWrangler uses these symbols:

Symbol	Meaning
Δ	tab
◊	space
•	non-breaking space
¬	line break
¶	page break
¿	other non-printing or special characters

When the Show Invisibles option is selected, the Show Spaces suboption is available, allowing you to turn off display of the visually "noisy" space characters if desired.

Syntax Coloring

When this option is selected and the editing window contains a document in a programming language TextWrangler recognizes, TextWrangler displays keywords and other language elements in color.

TextWrangler uses several methods to determine what language (if any) to use for a particular file. The primary way to activate syntax coloring in a document is simply to save it with a file name extension that indicates what programming or markup language the file contains. For example, if you save your file with ".html" at the end of the file name, TextWrangler will color your HTML tags and anchors. Other common suffixes are ".tex" for TeX files and ".c" for C files.

For any file whose name does not have an extension, or whose name has an extension that does not match any of the mappings in TextWrangler's Languages preferences, TextWrangler will attempt to guess what language the file contains and apply the appropriate syntax coloring. If TextWrangler guesses wrong (or is unable to guess), you can resort to the Language submenu of the Text Options pop-up menu in the status bar or the Language pop-up menu in the Text Options dialog, which gives you the ability to manually select *any* installed language to be applied to the document, regardless of its name. If the file is saved with "TextWrangler" state, the manual language selection will persist and override any suffix mapping.

By default, TextWrangler recognizes over 20 different languages and several dozen suffix mappings. You can add new suffixes to map to existing languages or (by installing third-party language plug-ins) add syntax coloring support for new languages as well. All the specific languages that TextWrangler recognizes, and the suffixes or extensions it expects for them, are listed in the Languages preferences panel, and suffix mappings can also be changed there. You can choose the colors that TextWrangler uses for syntax coloring in the Text Colors preferences panel.

Note   TextWrangler will recognize and syntax-color VBScript embedded within HTML via the <%…%> and <SCRIPT>…</SCRIPT> tags.

Show Fonts

You can choose the font, font size, text style, and tab spacing for a window by choosing Show Fonts from the Text menu, and using the standard system Font panel.

You can choose the desired font, font size, and style using controls in this window. You can also tell TextWrangler how many spaces should occur between tab stops.

IMPORTANT   The chosen display style will be used for *all* text in the window; TextWrangler does not support the use of selective text styles.

Note   The changes you make here affect only the active document. To set the default font, size, style, and tab information for all documents, use the "Default Font" option in the Editor Defaults preferences panel.

How TextWrangler Wraps Text

TextWrangler wraps text in one of two ways: soft wrapping or hard wrapping.

Soft wrapping is like the word wrapping found in most word processors. When the insertion point reaches a right margin as you type, the word processor automatically moves the insertion point to the beginning of the next line. You never need to type a carriage return (that is, press the Return key) at the end of a line, but only to start a new paragraph. If you place the insertion point in the middle of a paragraph and start typing, the text reflows so that words that are pushed out beyond the right margin end up on the next line. Usually, you use soft wrapping when you are editing memos, mail messages, and other prose. It is also useful for HTML documents. With soft wrapping, you generally do not have to scroll the window horizontally to see all the text in the file.

Unlike soft wrapping, hard wrapping requires a carriage return at the end of every line. When soft wrapping is turned off, TextWrangler lets you type as far as you like on a line, and never automatically moves the insertion point to the beginning of the next line. You have to manually type a carriage return to start a new line. You usually use hard wrapping to write programs, tabular data, resource descriptions, and so on. With hard wrapping, each line of source code or data appears on its own line in the window, although you may have to scroll the window horizontally to see the entire line if it is long.

Note   When you use the Hard Wrap command on a rectangular selection, lines will be padded with spaces as necessary.

Tip   If you open a file in TextWrangler that appears to consist of a few very long lines, you should select the soft wrapping option for that file.

This table summarizes the commands to soft-wrap and hard-wrap text. The sections that follow give details about using the wrapping commands.

To do this…	Do this…
Soft-wrap text as you type	Choose Soft Wrap Text from the Text Options pop-up menu (left) or select the Soft Wrap Text option from the Text Options dialog
Convert hard-wrapped text to soft-wrapped text	Use the Remove Line Breaks command in the Text menu, and activate soft wrapping
Convert soft-wrapped text to hard-wrapped text	Use the Add Line Breaks command in the Text menu
Hard-wrap text to a specific margin, reflowing paragraphs as needed	Use the Hard Wrap command in the Text menu

Soft Wrapping

To turn on soft wrapping for the active window do one of the following:

- Choose Soft Wrap Text from the Text Options pop-up menu (left) in the status bar.

- Select the Soft Wrap Text option from the Text Options dialog by choosing Text Options from the Edit menu.

To specify the wrapping margin, use the Text Options command. You can have text wrap at the Page Guide, the edge of the window, or a specific character position.

IMPORTANT Soft wrapping and rectangular selection are mutually incompatible. When soft wrapping is enabled, dragging the mouse performs normal (non-rectangular) selection even if the Option key is held down; when there is a rectangular selection, the Soft Wrap Text option is unavailable in the Text Options pop-up menu and dialog box.

To make soft wrapping the default for new windows, select the Soft Wrap Text option in the Editor Defaults panel of the Preferences window. You can also use the settings in that panel to specify the default wrapping margin.

To "freeze" the current line endings and hard-wrap the text at the current soft wrapping settings, use the Add Line Breaks command to insert a carriage return at the end of each line.

While TextWrangler prefers to break lines at white space when soft-wrapping, lines will be broken as close as possible to the designated wrap width if they do not contain any white space. This way, long URLs and other extended strings of characters are visible without requiring horizontal scrolling.

Exporting Soft-Wrapped Text
TextWrangler will not insert hard line breaks into Unix or DOS-format files upon saving.

Soft Wrapping in Browsers
Use the Text Options command from the Edit menu to control soft wrapping (and other display options) for files viewed in a browser window.

Soft Wrapping and Line Numbers
The preference Use "Hard" Line Numbering in Soft-Wrapped Views controls the way line numbers are displayed when you use soft wrapping. If this option is turned on, the line number bar, cursor position display, and Go To Line commands in editing views will use line numbers that correspond to "hard" carriage returns in the document, rather than to soft-wrapped line breaks. To restore the behavior of previous versions of TextWrangler, turn this preference off.

Hard Wrapping
The easiest way to hard-wrap text is to type a carriage return (by pressing the Return key) whenever you want to start a new line. If the file you are editing is a program, it is best to turn off soft wrapping altogether.

To turn off soft wrapping for the active window, do one of the following:

- Choose Soft Wrap Text from the Text Options pop-up menu in the status bar.

- Deselect the Soft Wrap Text option from the Text Options dialog box by choosing Text Options from the Edit menu.

To turn off soft wrapping for new windows, deselect the Soft Wrap Text option in the Editor Defaults section of the Preferences window.

TextWrangler provides two ways to convert soft-wrapped text into hard-wrapped text. The first is a simple technique that uses a single command; the second is a bit more complicated but gives you much more control over wrapping.

Hard-Wrapping Soft-Wrapped Text

To convert soft-wrapped text to hard-wrapped text, use the Add Line Breaks command in the Text menu. This command inserts a carriage return at the end of every line of the text as it appears in the window. If your wrapping margin is the edge of the window, you will get different results depending on the width of the window.

Note   The Add Line Breaks command does not turn off soft wrapping.

Hard Wrapping and Filling Text

The Hard Wrap command in the Text menu gives you more flexibility over hard-wrapping text than the Add Line Breaks command. While Add Line Breaks merely freezes the line breaks as they appear in the editing window, whereas the Hard Wrap command lets you specify a wrapping margin, apply the command to the selection only, and reflow paragraphs.

When you choose the Hard Wrap command, TextWrangler opens a sheet in the frontmost document:
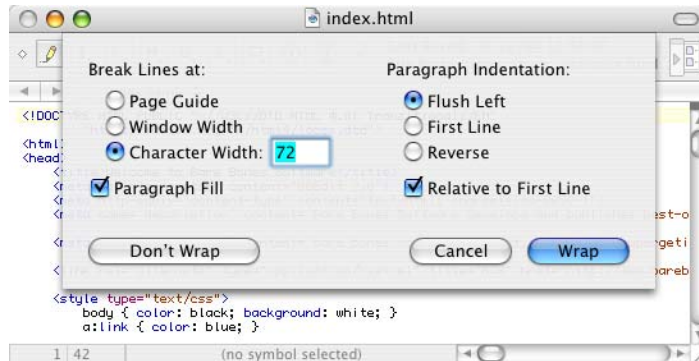
The controls in the left half of the sheet control how much of the text is hard-wrapped and the maximum length of lines after hard wrapping. The controls in the right half determine how this command deals with paragraphs.

The Selection Only option limits the effect of the command to the selected text. If this option is not selected, the command affects the entire contents of the window.

The "Break Lines at" buttons let you specify the wrapping margin.

If the Paragraph Fill option is selected, TextWrangler forms the lines into paragraphs before wrapping the lines. An example is the best way to illustrate this option.

Suppose you start with this text:
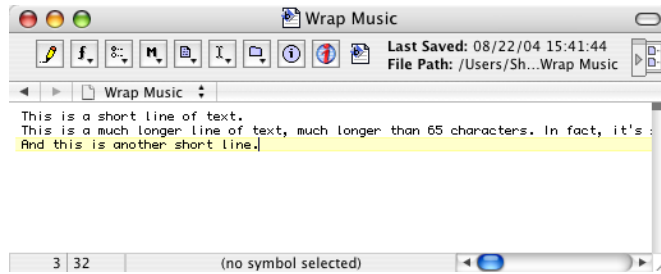
This is what happens when you wrap to 65 characters with Paragraph Fill off:
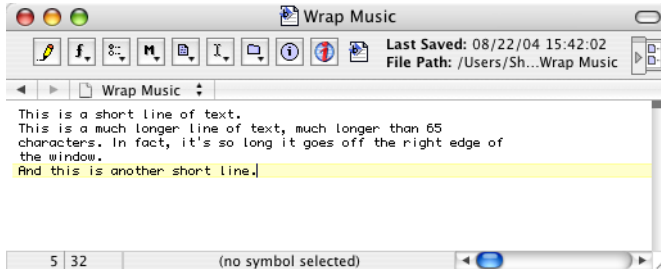
TextWrangler breaks the long line at a width of 65 characters (twice, because the line was so long) and leaves the short lines alone.

This is what happens to the same text when you wrap with Paragraph Fill on:
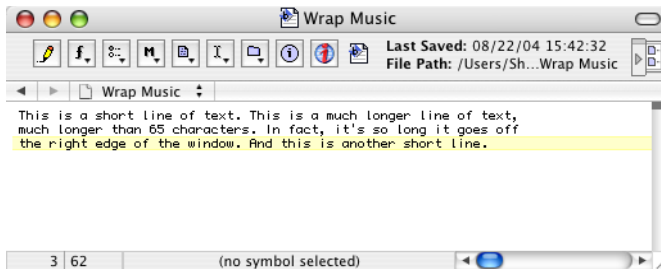
TextWrangler joins all the lines together to form a paragraph and then wraps the text to a width of 65 character.

The Paragraph Indentation buttons let you indent paragraphs after they have been wrapped.

- Flush Left does not indent paragraphs at all.

- First Line indents all lines in the paragraph by one tab stop.

- Reverse places the first line in the paragraph flush against the left edge of the window and indents all other lines in the paragraph by one tab stop.

Mark the Relative to First Line checkbox to make any paragraph indents relative to the original indent of the first line of the selection or document. If you want paragraph indents to be relative to the left margin of the document, make sure this checkbox is not marked.

Click the Wrap button to perform the Hard Wrap command. Click the Don't Wrap button to save the settings without changing the text.

Tip   If you hold down the Option key as you choose the Hard Wrap command, TextWrangler uses the last Hard Wrap settings to perform the operation, without displaying a sheet.

The Mark Submenu

A marker is a selection range that you can name. If a document contains any markers, you can select them from the Mark pop-up menu to move quickly to the specified section of the file.

Note   If you are programming, you may be tempted to use markers to mark functions in your source code. However, if TextWrangler supports the language you are using, this is usually unnecessary; your functions will automatically appear in the Function pop-up menu in the document window.

Setting Markers

To set a marker:

1 Select the text you want to mark.

2 Choose the Set Marker command from the Mark pop-up menu (identified by the icon shown at left), or Control-click the selected text and choose Set Marker from the contextual menu.

TextWrangler opens a sheet so that you can name the marker. If you have selected a range of text, the sheet will contain the first characters of the selection.
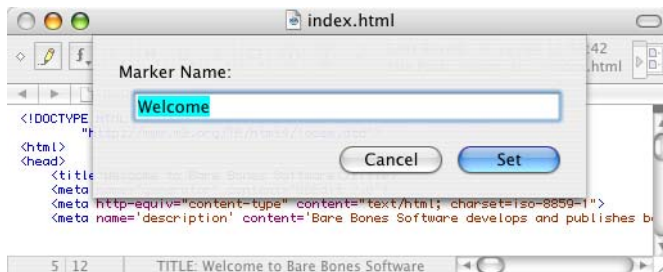
3 Click Set to set the marker.

Tip   If you hold down the Option key as you choose Set Marker, TextWrangler sets the marker using the leading characters of the selected text as the name of the marker, without displaying a dialog box.

Clearing Markers

To clear a marker:

1 Choose the Clear Markers command from the Mark pop-up menu.

TextWrangler displays the list of markers.
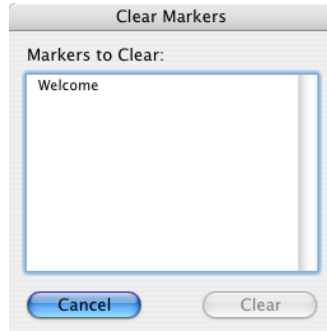
2 Select the marker you want to delete.

3 Click Clear to clear the marker.

TextWrangler also offers a Clear All Markers command, which clears all the markers in the document in one fell swoop. You can access this command by holding down the Option key and using the Mark pop-up menu.

Using Grep to Set Markers

You can use the Find & Mark All command in the Mark pop-up menu to mark text that matches a grep pattern. To learn more about using grep patterns, see Chapter 8, "Searching with Grep."

To use a grep pattern to mark text:

1 Choose the Find & Mark All command from the Mark submenu.

TextWrangler opens the Find & Mark All sheet.
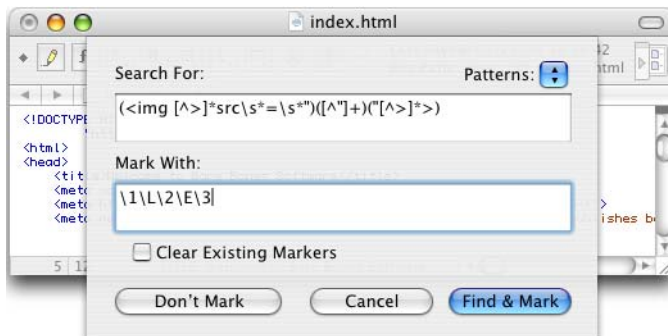
2 Type the pattern in the Search For field and the marker names in the Mark With field.

You can also choose stored patterns from the Patterns pop-up menu.

3 Click Find & Mark to mark the matching text.

TextWrangler searches the current document for text that matches the pattern and marks it the way you specified.

The Insert Submenu

In addition to typing, you can use the commands in the Insert submenu of the Edit menu to insert text into the active window. These commands, which are also available in the Insert pop-up menu (left) in the document status bar, let you insert the contents of other files, folder listings, Macintosh Toolbox templates, and page break characters.

Inserting File Contents

The Insert File Contents command inserts the contents of one or more files into the document you are editing. When you use this command, TextWrangler displays the Open dialog box and lets you select the files to insert. To insert more than one file hold down the Shift key or Control key as you click the files. TextWrangler then inserts the files at the insertion point or replaces the selected text. If you select more than one file, the files are inserted in alphabetical order, according to file name.

When you use the Insert File Contents command, you can now select more than one file from the dialog box.

Tip   You can also use the File Contents command in the Insert pop-up menu in the status bar, or you can drag a file's icon from the Finder into a TextWrangler editing window to insert the contents of that file.

Inserting a Folder Listing

The Insert Folder Listing command inserts a textual listing of a folder hierarchy. When you use this command, TextWrangler displays a directory dialog box that lets you select a folder to insert. TextWrangler inserts the folder listing at the insertion point or replaces the selected text.

Tip   You can also use the Folder Listing command in the Insert pop-up menu in the status bar to insert a folder listing, or you can drag a folder's icon from the Finder into a document to insert a folder listing.

Inserting File & Folder Paths

The Insert File Path command inserts the full path information for a selected file into the document you are editing, and the Insert Folder Path command inserts the full path information for the contents of the selected folder hierarchy. When you use these commands, TextWrangler opens a directory dialog box that lets you select the file or folder. TextWrangler inserts the path information at the insertion point or replaces the selected text.

Tip   You can also use the File Path or Folder Path command in the Insert pop-up menu in the status bar to insert path information.

Inserting a Page Break

To insert a page break, choose the Page Break command from the Insert submenu of the Edit menu. This will place a form feed character (ASCII 12) at the location of the insertion point. TextWrangler uses this character to indicate the start of a new page when printing.

Tip   You can also use the Page Break command in the Insert pop-up menu in the status bar to insert a page break.

Comparing Text Files

If you have ever had to reconcile changes between two different versions of a file, or even larger numbers of documents, you know how laborious this task can be. TextWrangler's Find Differences command is a powerful tool for doing such comparisons faster and more effectively. Using Find Differences, you can compare any two files, or the contents of two folders. You can also specify options to eliminate minor variations in document content, such as different amounts of white space, from being considered.

If you have two or more text documents open, choose the Compare Two Front Documents command on the Search menu to quickly compare the topmost two documents. (TextWrangler will automatically determine which document is newer and which older based on their modification dates.)

To compare two arbitrary files or folders:

1 Choose the Find Differences command from the Search menu.

TextWrangler opens the Find Differences dialog box.
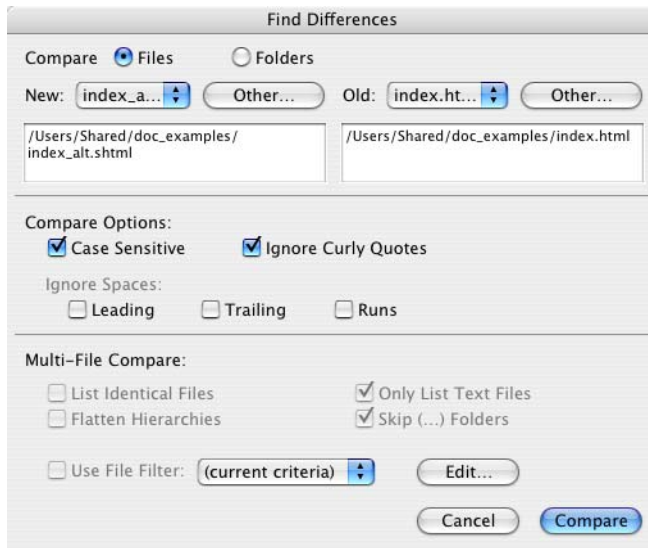
2 Click the Files radio button.

3 Use the New and Old pop-up menus to select the documents you want to compare.

If the files you want to compare are already open, they will appear in the pop-up menus; otherwise, you can select them by clicking the Other button next to one of the pop-up menus, or by dragging the files' or folders' icons from the Finder into the New and Old boxes in the Find Differences dialog.

You can also select recently opened files from the Recent Files item on the pop-up menu.

The terms "new" and "old" are used for convenience since most often you will want to find changes in the same file across time. However, the Find Differences command can be used to compare any two files or folders.

4 Select the Compare options that apply.

When the Case Sensitive option is selected, TextWrangler distinguishes uppercase from lowercase letters; deselect this option if you want TextWrangler to consider uppercase and lowercase letters the same.

When Ignore Curly Quotes is selected, TextWrangler treats typographers' quotes the same as straight quotes.

When one of the Ignore Spaces options is selected, TextWrangler ignores any tabs and spaces at the beginning or end of lines (Leading or Trailing, respectively) or runs of more than one space.

5 Click Compare to perform the comparison.

Alternatively, you can use the 'twdiff' command line tool to specify two files or folders, and have TextWrangler perform a Find Differences on them.

If the two files are different, TextWrangler tiles the documents according to your Differences preference settings and opens a Differences window below them.
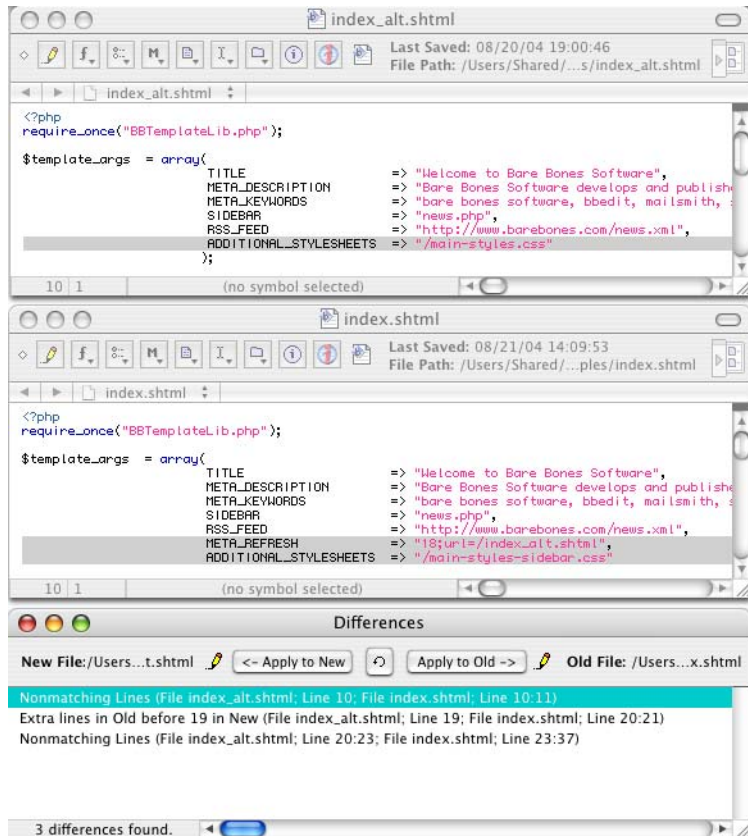
Tip   You can change this arrangement with the Arrange command in the Window menu. To change the default arrangement, use the Differences preferences panel.

The Differences window lists all the differences between the new file and the old file. To see the differences in context, click a line in the Differences window; TextWrangler scrolls and selects that spot in both files.

Use the Apply to New and Apply to Old buttons in the Differences window to make the new file look like the old file or vice versa. After you use one of these buttons, TextWrangler italicizes the line in the Differences window to indicate that you have already applied that change.

If a Differences window is open and is the frontmost window, the Compare Again command in the Search menu will recompare the two files being compared and refresh the list of differences accordingly. The small button (with the circular icon) between the Apply to New and Apply to Old buttons performs the same function.

Note   You can also compare the current clipboard contents against any file. Just choose Clipboard from either the New or the Old pop-up menu in the Find Differences dialog when the Clipboard window is open.

Compare Against Disk File

You can use the Compare Against Disk File command to compare the contents of the front document against the disk file for that same document. This capability makes it easy to locate in-progress changes to a document.

Multi-File Compare Options

You can compare multiple files at once by selecting the Folders button in the Find Differences dialog; TextWrangler lists all the files and marks those that are different with a bullet. You have the additional options described below.

List Identical Files

Normally, when you compare folders using the Find Differences command TextWrangler presents you with three lists: one list of the items that are in the first folder but not in the second folder, another list of the items that are in the second folder but not in the first one, and another list of the items that appear in both folders.

The list of items that appear in both folders generally displays a bullet next to items that are not identical. For example, if you have an archived mail folder that you are comparing against a current mail folder, mailbox files that appear in both the old and new file will all be listed together; however, if there have been any changes to the contents of particular mailbox files, the changed mailbox files will be listed with bullets next to them.

If you are comparing very large folders, however, the list of common items can be extremely long, making the flagged items hard to find. When you deselect the List Identical Files checkbox, TextWrangler will list only the flagged items (the ones that have been changed) in the list of items that appear in both folders.

Flatten Hierarchies

Normally, TextWrangler retains the hierarchy of the files being compared in a folder. In other words, when comparing folders, it looks in each subfolder of the first folder you select and tries to match it with a file of the same name in the same subfolder of the second folder, and so on down for all subfolders. If you choose Flatten Hierarchies, TextWrangler considers the files in the folders as a single flat list, allowing a file in one folder to match a file of the same name in the other folder, regardless of whether they are in the same subfolder in both hierarchies.

Only List Text Files

If this option is set, TextWrangler does not list non-text files when comparing folders.

Skip (...) Folders

If this option is set, TextWrangler skips subfolders whose names are enclosed in parentheses when comparing folders.

Use File Filter

File Filters allow you to select files for comparison with great precision. If either file in a compared pair matches the filter, the files are eligible for comparison; if *neither* file matches the filter, the files will not be compared. See Chapter 7, "Searching," for more information on creating, editing, and using file filters.

Note   When comparing folders with the Find Differences command, TextWrangler applies any specified file filter to the contents of the resulting "Only in new" and "Only in old" lists, so that only those files that match the filter criteria will appear in the lists.

Spell Checking Documents

The Find … Misspelled Words commands in the Text menu lets you check the spelling of the text in your documents. You can use either the Mac OS X system spelling checker (the default) or an external spelling checker that supports Apple's Word Services Suite (see "Using an External Spelling Checker" later in this chapter).

Using the Built-in Spelling Checker

Choose the Find Next Misspelled Word command from the Text menu, or type its key equivalent (Command-;) to start checking a document's spelling. TextWrangler will check every word in the document, starting from the top, using the system spelling checker. When TextWrangler encounters a word which is either misspelled or not in the checker's dictionary, it will draw a heavy red underline beneath the word. You can either type a correction, or Control-click on the word and select a suggested correction from the contextual menu.

To skip the identified word and continue checking, use the Check Spelling command again. To ignore all further instances of the word, Control-click on it and choose Ignore Spelling from the contextual menu. To add the word to the dictionary, Control-click on it and choose Learn Spelling from the contextual menu.

To check the spelling of all words in the document at once, choose the Find All Misspelled Words command, or type its key equivalent (Command-Option-;). TextWrangler will draw an underline under every questioned word in the document. You can then correct the spelling of any questioned word by typing, or by using the contextual menu to select a suggested correction or to skip, ignore, or add the word to the dictionary.

The Spelling Panel

In addition to allowing you to correct, ignore, or learn identified words, the Spelling panel allows you to choose which spelling dictionary TextWrangler will use, and to forget learned spellings. To use the Spelling panel:

1 Choose the Show Spelling Panel command from the Text menu.

TextWrangler opens the Spelling panel.

2 Set spelling options.

Choose a dictionary to use by selecting it from the Dictionary pop-up menu. Select Skip All Caps to avoid checking words consisting of only capital letters. (Note that these settings persist across runs of the application.)

3 Click Find Next to begin checking.

TextWrangler scans the document, and stops at the first misspelled or unrecognized word. This word is displayed in the text field to the left of the Correct button. Possible corrections for the questioned word are listed in the Guess box above.

4 If the questioned word is misspelled, choose the correct spelling from the Guess list or type it yourself in the Correct field.

5 Click one of the Spelling panel's action buttons to handle the questioned word.

Click Ignore to ignore further instances of the questioned word, without adding it to the active dictionary.

Click Guess to display a list of possible corrections.

Click Find Next to ignore this instance of the questioned word and continue checking.

Click Correct to replace this instance of the questioned word with the text in the adjacent text field.

Click Learn to add the questioned word to the active dictionary.

Click Forget to remove the questioned word from the active dictionary.

Using an External Spelling Checker

You can use any external spelling checker that supports Apple's Word Services Suite. To set up an external spelling checker:

1 Open the Preferences window (by choosing Preferences from the TextWrangler menu) and select the Spelling panel.

2 Select the Word Services option.

3 Click the Choose button and select an external spell checker.

You can also drag the icon of the spelling checker from the Finder to the path box to the left of the Choose button.

To start the external spelling checker, use the Check Spelling command just as you would for the built-in spelling checker. See the documentation for your spelling checker for details on how to use it once it has been invoked.

Text Transformations

This chapter describes the range of powerful text transformation commands offered by TextWrangler. In addition to providing individual commands which you can apply to the current document, TextWrangler allows you to run Text Factories which have been created in BBEdit. (Text Factories are sequences of commands that can be applied to one or more documents.)

In this chapter

Text Menu Commands

TextWrangler provides a variety of commands which you can use to transform text in different and useful ways. Most of these commands are situated in the Text menu, and described in this section. You can also use TextWrangler's search and replace capabilities, or additional plug-in tools, to transform text; each of these topics is covered in a separate chapter.

Unless otherwise specified, each of these commands will be applied to the active text selection in the frontmost document range, or if there is no active selection, to the entire contents of the document.

Hold down the Option key when selecting any command from the menu in order to quickly re-invoke it with its last-used option settings. (These "short form" commands are also available in the Set Menu Keys dialog, so that you can set key equivalents for them.)

Balance

This command locates the pair of parentheses, braces, brackets, or smart (curly) quotes that surround the insertion point or the current selection. If there are unmatched delimiters within this area, TextWrangler beeps. You can also double-click a delimiter character to invoke this command.

When syntax coloring is active for a document, Balance (including auto-balance) will ignore balance characters that appear inside strings or comments.

Exchange Characters

This command (once named Twiddle) swaps two characters according to the following rules:

- If there is no selection and the insertion point is not at the beginning or end of a line or of the document, this command transposes the two characters on either side of the insertion point.

- If the insertion point is at the beginning of a line or document, this command transposes the two characters following the insertion point.

- If the insertion point is at the end of a line or document, this command transposes the two characters before the insertion point.

- If there is a selection, this command transposes the characters at *either end* of the selection.

If you hold down the Option key as you choose this command, Exchange Characters becomes Exchange Words. Exchange Words behaves like Exchange Characters except that it acts on entire words rather than individual characters.

Change Case

This command lets you change between uppercase and lowercase characters, or capitalize word, line, or sentence starts. You can choose to change the text in the current selection or in the whole document. When you choose the Change Case command, the following sheet appears:

The radio buttons let you choose how to change the case of the text. The following table explains the function of each option in this dialog.

This button…	Changes the text like this…
ALL UPPER CASE	Every character changes to uppercase.

This button…	Changes the text like this…
all lower case	Every character changes to lowercase.
Capitalize Words	The first character of every word changes to uppercase; other characters are unaffected.
Capitalize sentences	The first character of every sentence changes to uppercase; other characters are unaffected.
Capitalize lines	The first character of every line changes to uppercase; other characters are unaffected.

Shift Left / Shift Right

These commands indent or outdent the selected text by one tab stop. If there is no selection, this command works on the current line. Hold down the Shift key while choosing these commands, to have TextWrangler indent or outdent the text by one space instead of one tab stop.

TextWrangler also entabs and detabs on the fly as you shift text. For example, if the selected text is indented one tab stop and you apply Shift Left One Space, the tab will be converted to spaces and the text will be outdented one space. If you then apply Shift Right One Space, the spaces will be converted back to a single tab.

Un/Comment Selection

This command automates the task of commenting and uncommenting sections of code in various programming languages. Choose a range of text and apply this command to add or remove comments to it, depending on its initial comment state. If there is no selection, this command place a comment at the insertion point.

Note   You can use the Options button of the Installed Languages list in the Languages preferences panel to modify or set comment strings for any available languages.

Hard Wrap

This command wraps long lines by inserting hard line breaks and can reflow (fill) paragraphs if desired. See "How TextWrangler Wraps Text" on page 76 for more information.

Add Line Breaks

This command inserts a hard line break at the end of each line of text as displayed. See "How TextWrangler Wraps Text" on page 76 for more information.

Remove Line Breaks

This command removes carriage returns and spaces from sections of text. Use this command to turn text that has hard line breaks into text that can be soft-wrapped. See "How TextWrangler Wraps Text" on page 76 for more information.

Apply Text Factory

This command presents a submenu which lists all the text factories present in the Text Factories folder of TextWrangler's application support folder. When you choose a text factory from the menu, TextWrangler will run that factory against the current document. (See "Text Factories in TextWrangler" on page 101 for more information on installing and using text factories.)

Apply Text Factory <most recently used>

This command allows you to reapply the most recently used text factory against the current document.

Convert to ASCII

This command will convert certain eight-bit Mac Roman characters (characters whose decimal values are greater than 128 and less than 255) to 7-bit (printable ASCII range) equivalents. Converted characters include umlauted and accented vowels, ligatures, typographer's quotes, and various specialized punctuation forms. This conversion may entail expansion to multiple characters; for example, in the case of ligatures.

Educate Quotes

This command converts straight quotes (" and ') to typographer's quotes (" " and ' ').

Note   You should not use this plug-in to prepare text for posting on a web page or use in an email, as typographer's quotes in the Mac character set will generally not be properly displayed by applications on other platforms.

Straighten Quotes

This command performs the reverse of Educate Quotes; it converts typographer's quotes (" " and ' ') to straight quotes (" and ').

Add/Remove Line Numbers

This command displays a sheet which allows you to add or remove line numbers for each line of the selected text or of the document. You can set the starting number and numbering increment, as well as whether to right-justify the inserted numbers, by choosing the appropriate options.
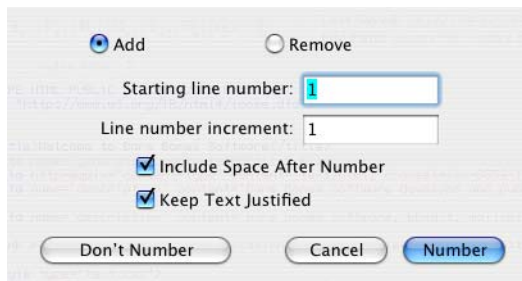
Prefix/Suffix Lines

This command displays a sheet which allows you to add or remove the specified prefix and/or suffix strings to each line of the selected text or of the document.
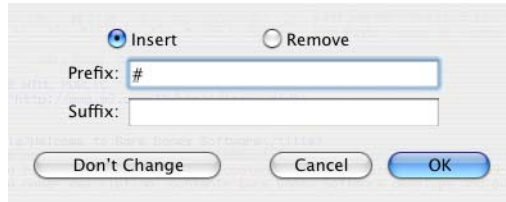
If you define both a prefix and a suffix string, TextWrangler will apply them to the text at the same time.

Note When using the "add prefix", "add suffix", "remove prefix", or "remove suffix" scripting commands, the string direct parameter is required.

Sort Lines

This command displays a sheet which allows you to sort lines of text by collating them in alphanumeric order. The sorted lines can be copied to the clipboard, be displayed in a new untitled window, replace the selection within the original document, or any combination of the three.
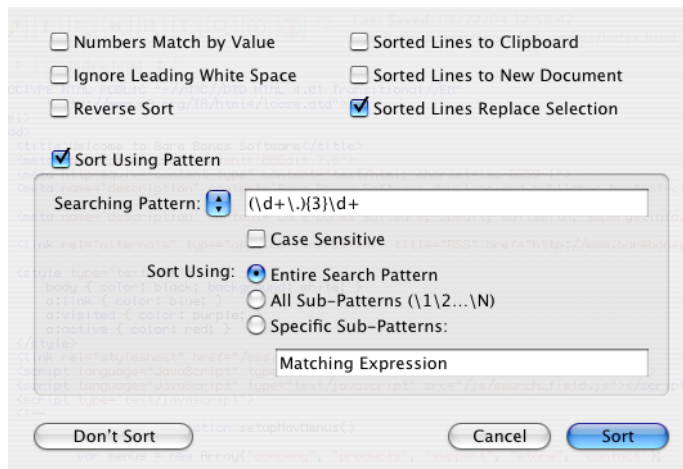
There are also options for ignoring white space at the beginning of lines, taking case distinctions into account, sorting strings of digits by numerical value instead of lexically, and sorting in descending rather than ascending order.

By checking the Sort Using Pattern option, you can specify a grep pattern to further filter the lines to be sorted. If the pattern contains subpatterns, you can use them to control the sort order based on the contents of the strings they match. The Case Sensitive option controls the case sensitivity of the search pattern in the same way as the equivalent option in the Find dialog.

For example, suppose you are sorting a list of cities together with their two-letter state abbreviations, separated by a tab character. The pattern and subpatterns shown in the figure will sort the results first by city name and second by state abbreviation. Changing the contents of the Specific Sub-Patterns field from "\1\2" to "\2\1" will instead sort the results by state first and by city second.

IMPORTANT   When you use a grep pattern with this command, matches are not automatically anchored to line boundaries, so ambiguous patterns may produce unpredictable results. To avoid this problem, you should use the line start ^ and line end operators as necessary. Also, keep mind that the pattern will only be tested against a single line at a time. So, if the pattern would matches a string which spans multiple lines, but not a single line, the line will be skipped

Process Duplicate Lines

This command displays a sheet which allows you to locate duplicate lines within a body of text and operates on them in various ways.

The Matching All option processes all duplicate lines; Leaving One ignores the first of each set of duplicate lines and processes only the additional ones.

The Numbers Match by Value and Ignore Leading White Space options allow you to choose whether strings of digits should be evaluated numerically or compared as strings, and whether white space at the beginnings of lines should be considered.

The Match Using Pattern option allows you to use a grep pattern to further filter the lines to be processed. You can enter a pattern in the Searching Pattern field, or choose a stored pattern from the pop-up menu. The Match Using: radio buttons control what part of the specified pattern should be used to determine duplication.

IMPORTANT   When you use a grep pattern with this command, matches are not automatically anchored to line boundaries, so ambiguous patterns may produce unpredictable results. To avoid this problem, you should use the line start ^ and line end operators as necessary.

The options on the right-hand side of the sheet allow you to specify how duplicate lines should be handled once they have been identified. You can copy duplicate lines to the clipboard (Duplicates to Clipboard), copy them to a new document (Duplicates to New Document Window), and/or delete them from the current document (Delete Duplicate Lines). You can likewise specify how to handle the lines that are *not* duplicated by choosing Unique Lines to Clipboard and/or Unique Lines to New Document).

Since each of these options is an independent checkbox, you can select any combination of them that you wish. For example, selecting both Delete Duplicate Lines and Unique Lines ton Clipboard would delete the duplicate lines from the document and copy them to the clipboard for pasting elsewhere.

Process Lines Containing

This command displays a sheet which allows you to search the active window for lines containing a specified search string and then removes those lines or copies them to the clipboard. The options on the left side of the dialog box control how the search is performed and the options on the right side control what happens to the lines that are found.
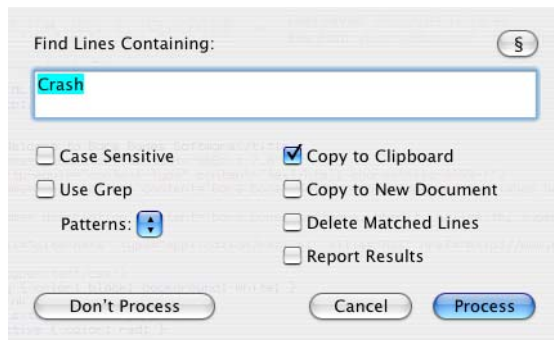
To specify a search pattern, enter it in the Find Lines Containing field. If you do not want TextWrangler to match text when the letters in the text differ from the letters in the search string only by case (upper-case versus lower-case), select Case Sensitive.

To search using a grep pattern, select Use Grep and enter the pattern in the text field. You can also select a predefined search pattern from the Patterns pop-up menu or click the "grab selection" (§) button to use the current selection as the search pattern.

Note   If the selection ends in a trailing carriage return, the carriage return will be omitted from the search string copied into the text field.

The checkboxes on the right of the sheet control the way lines containing the specified search pattern will be processed. By selecting the appropriate combinations of these options, you can achieve the effect of applying various editing commands to each line:

- Setting both Copy to Clipboard and Delete Matched Lines on is equivalent to applying the Cut command.

- Setting Copy to Clipboard on and Delete Matched Lines off is equivalent to applying the Copy command.

- Setting Copy to Clipboard off and Delete Matched Lines on is equivalent to applying the Clear command.

The Copy to New Document option opens a new, untitled document containing copies of all lines matching the search pattern, whether or not they are deleted from the original window. By using this option and turning Copy to Clipboard off, you can collect all matching lines without affecting the previous contents of the clipboard.

The Report Results option causes TextWrangler to display a dialog reporting the total number of lines matched, regardless of their final disposition. With all of the other options turned off, this can be useful for pretesting the extent of a search operation without affecting the clipboard or the contents of the original window.

Rewrap Quoted Text

This command rewraps hard-wrapped text having Internet-style quoting, while retaining the quoting characters and quote level.
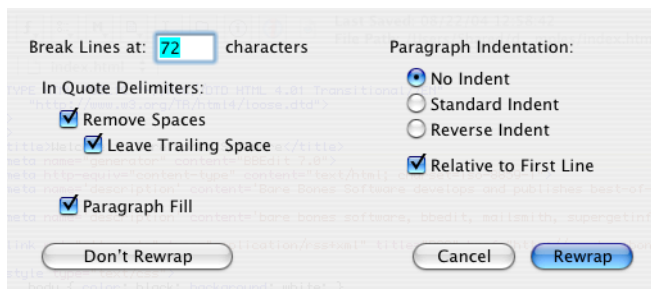
In Internet messages, it is common to use the ">" symbol to indicate that part of a message is quoted from a message that is being replied to. As a message gets batted back and forth in a discussion, the oldest bits of text will end up having several ">" symbols in front of them. Each line of text in the message has a carriage return at the end, making rewrapping the text to a different width somewhat problematic.

When you apply this command, TextWrangler first extracts each chunk of quoted text (a successive set of lines with the same number of markers), and temporarily removes the markers and any hard line breaks from the chunk of text, forming it into a soft-wrapped paragraph. TextWrangler then hard-wraps that paragraph according to your chosen settings, which are the same as for the Hard Wrap command (see "Hard Wrap" on page 93), and reinserts the quote markers.

Note   When you use this command on a rectangular selection, TextWrangler will pad lines with spaces as necessary.

Increase and Decrease Quote Level

This command inserts or deletes a standard Internet quote character (">") from the selected hard-wrapped text, or for the current line if there is no selection.

Strip Quotes

This command removes all Internet-style quoting from the selected hard-wrapped text, or from the current line if there is no selection.

Zap Gremlins

This command displays a sheet which allows you to remove or replace various non-printing characters, often known as "gremlins". Use this command when you have a file that may contain extraneous control characters, or any non-ASCII characters, which you wish to identify or remove.
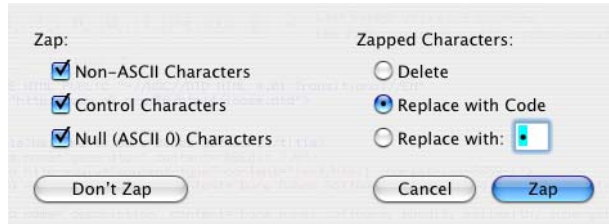
The checkboxes on the left-hand side of the sheet determine which types of characters the Zap Gremlins command affects, while the radio buttons on the right-hand side determine what to do with gremlins that are found.

Zap Non-ASCII Characters

When this option is selected, Zap Gremlins zaps *all* characters in the file that do not fall in the 7-bit (or ASCII) range. Examples of such characters include special Macintosh characters such as bullets (•) and typographer's quotes (" and ", ' and '), as well as all multi-byte characters. In general, such special characters are those that you type by holding down the Option key.

Zap Control Characters

When this option is selected, Zap Gremlins zaps a specific range of invisible low-ASCII characters, also known as control characters. Control characters can cause compilers and other text-processing utilities to malfunction, and are therefore undesirable in many files.

Zap Null (ASCII 0) Characters

When this option is selected, Zap Gremlins zaps all instances of the null character (ASCII 0). Like other control characters, nulls can cause many programming tools and text-processing utilities to malfunction. This specific option is included in case you want to remove only nulls without affecting other control characters that may be present in a file.

Delete

This option removes the zapped character completely from the text. It is useful if you are only interested in destroying gremlins and you do not care where they were in the text.

Replace with Code

This option replaces the gremlin character with any other character specified in escaped hexadecimal format. The escape code is formed via the same convention used by the C programming language: \0x followed by the character code in hexadecimal (base 16). This option is useful for identifying both the value and the location of gremlin characters. Later, you can search for occurrences of \0x to locate the converted characters. (Searching for the grep pattern of "\\0x.." will select the entire character code for easy modification or deletion.)

Replace with <character>

This option replaces the gremlin with the character you type in the text field next to the radio button. It is useful for identifying the location of gremlins, but not their value. The replacement character can be specified not only as any typeable character, but also by using any of the special characters defined for text searches, including hex escapes. (See "Special Characters" on page 115.)

Note   In some cases, this option could be counterproductive, since hex escapes (\xNN) can themselves be used to insert unprintable characters.

Entab

This command displays a sheet which allows you to set the number of consecutive space characters which should be converted into tabs. This transformation is useful when you are copying content from many online sources, which use spaces to line up columns of text. If you do not use a monospaced font, columns usually will not line up unless you entab the text first.

Detab

This command displays a sheet which allows you to set the number of consecutive spaces which should replace each tab. This command is useful when you are preparing text for use in a program which has no concept of tabs as column separators, for email transmission, and similar purposes.

Normalize Line Endings

This command converts a document containing mixed line endings to have a uniform set of line endings.

If you open a file which contains a mixture of Mac, Unix, and DOS/Windows line endings, the "Translate Line Breaks" option may not suffice to properly convert the document for viewing and editing. After conversion, the document may appear to not have any line breaks at all (this usually happens if the first line break in the file is a Mac line break, and all the rest are Unix), or to have an invisible character at the beginning of each line.

Should this happen, use Normalize Line Breaks to convert the remaining line endings, and save the document. Once you have done this, the document's line endings will be consistent, and TextWrangler's line-break translation will suffice when you next open it.

Text Factories in TextWrangler

A text factory is a special type of document that enables you to apply either BBEdit's or TextWrangler's powerful text transformation commands in the order and fashion that you decide, to whatever collection of files you choose. So, for example, if you routinely need to process a folder full of server logs by reducing them to lines containing "www.apple.com", prefixing each line with a line number, and converting the file's line endings to Macintosh, you can assemble and save a text factory to do that work for you, and apply it at any time.
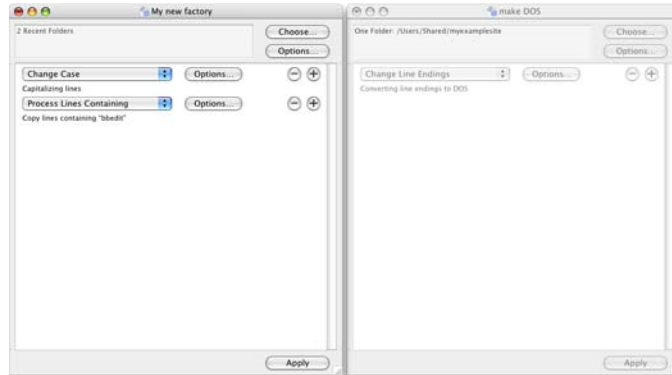
Although you can run any existing text factory in TextWrangler, you must use BBEdit to create or modify a text factory.

For complete details about the capabilities of text factories and how to create them, please refer to the "Text Factories" section in Chapter 5 of the BBEdit user manual.

Note   Text factory documents will have either a file type of "TxEN" and a creator type of "R*ch", or a filename extension of ".textfactory". If you store documents in any manner which does not preserve file type info, you must name your text factory documents accordingly.

Installing Text Factories

To make a text factory available, place it in the Text Factories folder of the TextWrangler application support folder. Once you have done so, that text factory will be available in the Text Factories menu, in the Apply Text Factory command (see page 94), and on the Text Factories palette (see page 107).

The Text Factories Menu

This menu lists all the text factories installed in the Text Factories folder. To run a text factory, choose it from the menu. TextWrangler will then display the Run Text Factory dialog, which will display information about the target files and folders (if any) that the text factory is configured to process.

Choosing Targets

If the text factory does not have any defined targets, or you would like to add additional targets, click Choose to present a sheet containing a list of selected and available target items. (This list is similar to the Sources drawer used for multi-file searches in the Find dialog.)

To choose or deselect an item as a target, click the checkbox next to its name. To add a file or folder to the list, click Other and select it in the resulting Choose Object dialog.

Click Options to select additional options for controlling which target items will be processed. To process all the files in subfolders of each target folder, mark the Search Nested Folders checkbox. The Skip (…) Folders checkbox skips the contents of folders whose names are enclosed in parentheses. The "Search Invisible Folders" checkbox allows you to process the contents of invisible folders.
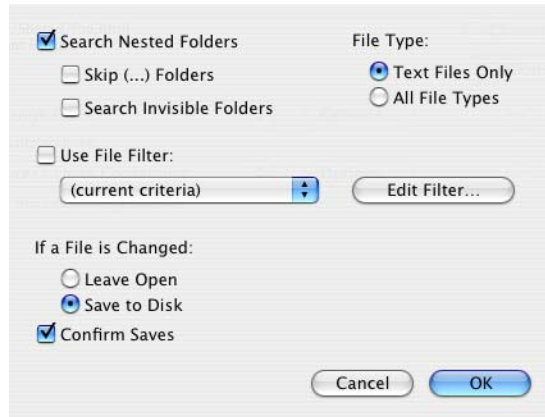
You can also choose to process only text files or to process all file types. If you have graphics or other types of files in the target folders, you should restrict processing to only text files. This setting works in addition to any file filter (see "File Filters" on page 120) and is in fact applied *before* the filter.

The last group of options controls how TextWrangler treats processed documents. Choose Leave Open to have TextWrangler leave all the documents open so that you can inspect the results of the operation. Choose Save to Disk to have TextWrangler automatically save changes to each file after processing it. When the Confirm Saves setting is active, you will have an opportunity to approve the changes before TextWrangler saves them to disk. You should not turn this off unless you are sure that the processing being applied is what you want.

Applying a Text Factory
Once you have selected the files and folders to process, click Run to have TextWrangler apply the actions specified by the text factory to each file in the target set. This processing happens in the background, so you can keep using TextWrangler while it's going on (similar to a multi-file search operation).

CHAPTER

6

Arranging Windows & Palettes

This chapter describes the commands in the Window menu. These commands allow you to arrange and access editing and browser windows quickly, and also to access TextWrangler's extensive set of tool palettes and floating windows.

In this chapter

Window Menu

The Window menu provides easy, centralized access to all of TextWrangler's tool and utility palettes, in addition to offering commands that you can use to access and organize editing and results windows on screen.

TextWrangler also offers several preference options (in the Applications panel of the Preferences window) so that you have greater control over the listing of open documents. You can choose whether items are grouped by window kind, or are all listed together without dividers. You can also elect to sort windows by name or in order of creation. Please refer to Chapter 10 for additional details.

Minimize Window

This command puts the frontmost window into the Dock. Click the window icon in the Dock to restore the window. Hold down the Option key and this command will become Minimize All Windows.

Bring All to Front

This command will bring all un-minimized TextWrangler windows to the front.

Palettes

The Palettes submenu provides quick access to all of TextWrangler's numerous tool palettes and utility windows. Choosing an item from this submenu toggles display of the corresponding palette.

When moved or resized, palettes now automatically "snap" to the edges of the screen and the edges of other palettes. You can override this behavior by holding down the Shift key while dragging or resizing.

ASCII Table

The ASCII Table command opens a palette that contains the 127 entries of the ASCII character set plus all of the standard extended (8-bit) Macintosh character set (MacRoman). The decimal value for each character is displayed in the center column, while in the right-hand column, the character value is displayed in either hexadecimal "escape" format, or in URL-encoded format, based on the language mapping of the frontmost text window.

Depending on the modifier keys you hold down, the Insert button inserts the selected character in different formats:

Clicking Insert while holding...	Inserts in this format...
None	Escape code appropriate to the front window—for example, (\x69) or (%69)
Option	Decimal value—for example, (105)
Command	Literal character—for example, (i)

Note You can also double-click on a line in the ASCII table to insert the corresponding character or character code into the editing window.

Clicking the Show button in the ASCII Table window displays the ASCII value of the character to the right of the insertion point or the first character of the selection.

Plug-In Tools

The Tools palette displays a list of all the plug-ins in your TextWrangler Plug-Ins folder. Any plug-ins you have installed will appear both in this Tool List window and in the Tools menu itself. See Chapter 13, "Using Plug-Ins," for more information on installing plug-ins.

Scripts

The Scripts palette displays all the currently installed AppleScripts in your TextWrangler Scripts folder. See Chapter 11, "Scripting TextWrangler," for information about using AppleScript with TextWrangler.

Stationery

The Stationery List is a palette that displays all the stationery pads you have placed inside the Stationery folder of TextWrangler's application support folder. You can create a new document from any of these pads by double-clicking it in this list. Although the document created will have the content and all the state information from the stationery pad, it is a new untitled document separate from the stationery pad.

To create a stationery pad, click the Save As Stationery checkbox when saving the file from TextWrangler. Alternately, any document can be changed into a stationery pad in the Finder by clicking the Stationery Pad checkbox in the document's Get Info window.

By default, items in the Stationery List are displayed in alphabetical order. However, you can force them to appear in any desired order by including any two characters followed by a right parenthesis at the beginning of their name. (For example "00)Perl scripts" would sort before "01)CGI scripts".) For such files, the first three characters are not displayed in TextWrangler. You can also insert a divider by including an empty folder ending with the string "–***". (The folder can be named anything, so it sorts where you want it.) These conventions are the same as those used by the utilities FinderPop and OtherMenu.

Note   In the Tools, Stationery, or any of the Scripts palettes, the Set Key button allows you to assign key equivalents to any item contained in that window. You can use combinations of the Command, Shift, Option, and Control keys, plus any single other key, to create such equivalents, except that any equivalent must contain either the Command or Control keys (or both). You can also map Function keys directly to items, with or without the use of a modifier.

Text Factories

The Text Factories palette displays all the text factory documents that you have placed inside the Text Factories folder of TextWrangler's application support folder. You can run any available text factory against the current document by double-clicking on its name. (See "Text Factories in TextWrangler" on page 101 for more information on installing and using Text Factories.)

Windows

The Windows palette displays the names of all open windows, ordered either by name, by creation order, or by window kind, as determined by the settings your Application preferences panel (see Chapter 10). You can open a file by dragging its icon from the Finder into the Windows palette.

Document windows, which correspond to text files, have a document icon next to them; display windows, such as browsers and search results windows, do not. A solid diamond to the left of a window's name means that the window's contents have been modified and have not yet been saved, while a hollow diamond indicates that the window's state has been modified but not yet saved.

To bring any window to the front, click its name in the Windows palette. You can select one or more windows in the list, and choose the Save, Close, or Print commands from the action menu at the top of the palette. Holding down the Option key changes these commands to Save All, Close All, and Print All, which apply to all listed windows for which the given command is possible. You can also Control-click on any selected windows and apply the Save, Close, or Print commands from the resulting contextual menu.

"Hovering" the mouse over a window name displays a tool tip showing the full window title; this is useful for names that have been truncated with ellipses (…) because they are too long to fit within the width of the window. If you hold down the Option key, the tool tip will appear instantly, with no hovering delay. Holding down the Command key displays the full pathname for document windows (or other relevant windows such as disk browsers and FTP browsers).

Unix Scripting Tools, Unix Filters, and Unix Scripts

TextWrangler integrates directly with any Unix scripting language, including Perl, shell scripts, and any other scripting languages you install (such as Python or Ruby).

The Unix Scripting Tools palette contains a subset of the commands available in the Shebang menu. The Unix Filters palette displays shell scripts that read the selection of the current document window as STDIN and replace the selection with STDOUT. For more information on these tools, see Chapter 12, "Unix Scripting and the Command Line."

Arrange

The Arrange command gives you several ways to organize TextWrangler windows. When you choose the Arrange command, TextWrangler opens the Arrange Windows dialog box.
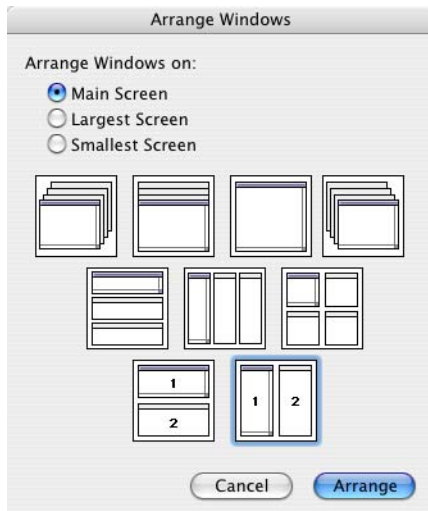
The radio buttons at the top of the dialog specify which screen the windows will be arranged on. You can choose the main screen, the largest screen, or the smallest screen.

Icon	Description
	Stacks every window so that some part is visible no matter which window is frontmost. If you select the Leave Room for Finder options in the Windows section of the Preferences window, TextWrangler leaves room along the right side or bottom of the screen for Finder icons. TextWrangler offers four different ways to stack windows: down and to the left, straight down, atop, and down and to the right. These are the top four choices in the dialog.
	TextWrangler tiles the windows in two or three rows (or columns). If you have more than three windows open, TextWrangler stacks additional windows behind the three front windows.
	TextWrangler figures out how many rows and columns it needs to tile windows. The larger your screen, the more rows and columns TextWrangler uses. The windows are never narrower than half of a classic Macintosh screen.
	TextWrangler tiles the front two window horizontally or vertically and stacks any additional windows behind the two front windows.

Note To arrange the windows using the same settings as the last time you used this command, hold down the Option key as you choose Arrange from the Window menu.

Zoom

There is no actual Zoom command in the Window menu, but you can use the key equivalent of Command-/ to produce the same effect as clicking a window's zoom box: it makes the active window larger if it is small, or returns it to its original size if it was previously enlarged by a Zoom command.

When zooming windows, TextWrangler will move the window as little as possible (consistent with maximizing the window's size). This behavior is similar to what the Finder does when zooming a window. The "Move as Little as Possible" switch in the Windows preferences panel controls this behavior; turning this switch off will revert the behavior to be the same as previous versions of TextWrangler.

Tip You can Option-click the zoom box to zoom all open windows, or Command-click to zoom the selected window to its largest possible size (as determined by the Window Zooming preference options in the Windows preferences panel).

Send to Back

This command sends the front window behind all the other windows.

Exchange with Next

This command makes the second window the active window. Choose this command repeatedly to alternate between the front two windows.

Synchro Scrolling

When you have two or more windows open, Synchro Scrolling makes both files scroll when you scroll one. This feature is useful to look over two versions of the same file.

Window Names

The last items in the Window menu are the names of all the open documents, browsers, and other editing windows. Choose a window's name from this menu (or use its numbered Command key equivalent, if applicable) to bring that window to the front.

Tip   You can also use the Windows floater to select windows quickly.

Searching

This chapter describes TextWrangler's powerful Find command, now enhanced with a flexible file filtering mechanism. It tells you how to search for text in the active window or within a set of files. TextWrangler can also do advanced pattern, or grep, searching. To learn about pattern searching, you should read this chapter first and then read Chapter 8, "Searching with Grep."

In this chapter

Basic Searching and Replacing

This section describes the basic steps for searching and replacing text in a document. Later sections in this chapter cover more advanced techniques. To search and replace text in the front document, follow these steps:

1 Choose Find from the Search menu. TextWrangler opens the Find & Replace dialog box.
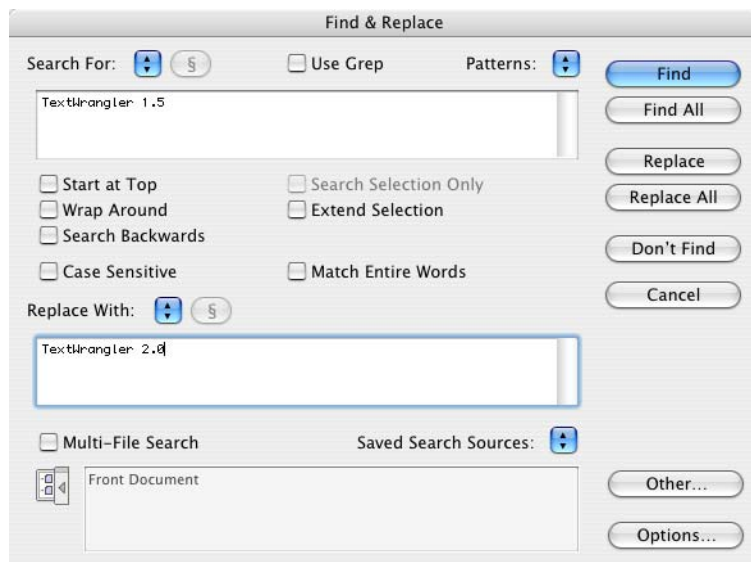
For now, disregard the bottom part of the dialog—the Multi-File Search checkbox and everything below it. The section "Multi-File Searching" later in this chapter discusses this part of the dialog box.

2 Type the string you are looking for in the Search For text field.

You can use special characters in the Search For text field to search for tabs, line breaks, or page breaks. See "Special Characters" later in this section.

TextWrangler remembers the last 12 search terms you used since launching the application. If you are searching for something you recently searched for, you can choose it from the pop-up menu above the Search For field.

Click the § button to enter the document selection in the Search For field.

3 Type the replace string (if any) in the Replace With text field.

TextWrangler also remembers the last 12 replace terms you used since launching the application. These appear in the pop-up menu above the Replace With field.

Click the § button to enter the document selection in the Replace With field.

4 Mark any checkboxes in the middle part of the dialog box that you want to apply to your search.

For more info about these options, see "Search Settings" later in this section.

5 Click one of the buttons along the right side of the dialog box.

Note The size of both the search and replace strings is limited only by available memory. However, the memory requirements for searching multi-byte text files are generally at least double those for single-byte text.

The following table explains what each of the buttons does.

This button…	Does this…
Find	Finds the first occurrence of the text in the active window. Shortcut: Cmd-F
Find All	Finds all the occurrences of the search string and displays the results in a search results window. Shortcut: Cmd-Opt-F
Replace	If there is a selection, replaces the current selection with the replace string. Otherwise, finds the first occurrence of the text in the active window after the current insertion point and replaces it with the replace string. Shortcut: Cmd-R
Replace All	Replaces every occurrence of the search string in the active window with the replace string. Shortcut: Cmd-Opt-R
Don't Find	Saves the settings of the Find & Replace dialog without doing a search. Shortcut: Cmd-D
Cancel	Does not do the search and restores the settings of the dialog box to their previous state. Shortcut: Cmd-.

TextWrangler closes the Find & Replace dialog and then selects the search string in the active window if the search was successful. Once TextWrangler finds your text, you can use the commands in the Search menu (see "Search Menu Reference" later in this chapter). The table below summarizes the most common commands you can use at this point.

This command…	Does this…
Find Again	Finds the next occurrence of the search string without displaying the Find & Replace dialog again. To reverse the search direction, hold down Shift.
Replace	Replaces the selection with the replace text
Replace All	Replaces all occurrences of the search string with the replace string.
Replace & Find Again	Replaces the selection with the replace string and looks for the search string again.

Search Settings

The checkboxes in the Find & Replace dialog let you control how TextWrangler searches your document for the indicated text.

Note   You can set the defaults for many of these settings in the Text Search section of the Preferences window.

Use Grep

When this checkbox is selected, TextWrangler treats the search and replace strings as grep patterns. Otherwise, TextWrangler searches the document for text that matches the search string as it appears literally, and will replace any matched text with the replace string. To learn more about pattern searching see "Searching with Grep" on page 129.

Start at Top

When this checkbox is selected, TextWrangler always starts searches from the beginning of the document. Choosing this option will disable the Wrap Around and Search Backwards settings.

Wrap Around

When this checkbox is selected, TextWrangler continues searching from the beginning of the document if a match is not found (or from the end of the document if searching backwards). Otherwise, TextWrangler stops searching when it reaches the end (or the beginning if searching backwards) of the file. Choosing this option will automatically disable Selection Only. (Unavailable if Start at Top is checked.)

When performing a Replace All with Wrap Around selected, the replace is transformed into "Start at Top" to allow for performance optimizations.

Search Backwards

When this checkbox is selected, TextWrangler searches from the insertion point to the beginning of the document. Otherwise, TextWrangler searches from the insertion point to the end of the file. (Unavailable if Start at Top is checked.)

Search Selection Only

When this checkbox is selected, TextWrangler searches only the selected text. Otherwise, TextWrangler searches the entire document. (Unavailable if Wrap Around or Start at Top is checked.)

Extend Selection

When this checkbox is selected, TextWrangler extends the selection from the current insertion point to the end of the matched search string.

Case Sensitive

When this checkbox is selected, TextWrangler treats upper- and lowercase letters as different letters. Otherwise, TextWrangler treats upper- and lowercase letters as if they were the same.

Match Entire Words

When this checkbox is selected, TextWrangler matches the search string only if it is surrounded in the document text by word-break characters (white space or punctuation). Otherwise, TextWrangler matches the search string anywhere in the text.

Multi-File Search

When this checkbox is selected, TextWrangler searches a set of files for the search string. See "Multi-File Searching" below to learn more about searching across multiple files.

Special Characters

You can use the following special characters to search for line breaks and other non-printing characters, as well as hexadecimal escapes to search for any desired 8-bit character.

Character	Matches...
\r	line break (carriage return)
\n	Unix line break (line feed)
\t	tab
\f	page break (form feed)
\xNN	hexadecimal character code NN (for example, \x0D for CR)
\\	backslash (\)

The form of a hex escape is "\xNN", where "N" is any single hex digit [0-9,A-F]. The "x" may be upper or lower case. (You can use the ASCII Table in the Window menu to find the hex value for any 8-bit Macintosh character.) You can perform a literal search for any character, including a null, using this option. Malformed escapes are treated as literal strings.

Note   In older versions of TextWrangler, you could not perform a grep search for a null character (ASCII 0), even if it was escaped. This limitation is no longer present; you can find nulls as part of a grep search.

Multi-File Searching

The only difference between single-file searching and multi-file searching is that to perform a multi-file search, you must specify the files to be searched. TextWrangler gives you a great deal of flexibility in how to do this. You can search all the files in a given folder, in open editing windows, or in an existing search results browser. For even great control, you can choose a diverse sets of search sources, or apply TextWrangler's advanced multi-criteria file filtering options.

IMPORTANT   Multi-file searching is no longer a modal operation. When you start a search, TextWrangler will display a search progress window and return control, so that you can continue working. You can perform more than one multi-file searches at a time; each search will have its own progress window. Closing a search's progress window or clicking Cancel in the progress window will stop the operation, and TextWrangler will display a search results browser containing any matches found up to that point.

Starting a Search

To search for a string in multiple files, do the following steps:

1 Choose Find from the Search menu to open the Find & Replace dialog box (if it is not already open).

2 Mark the Multi-File Search checkbox.
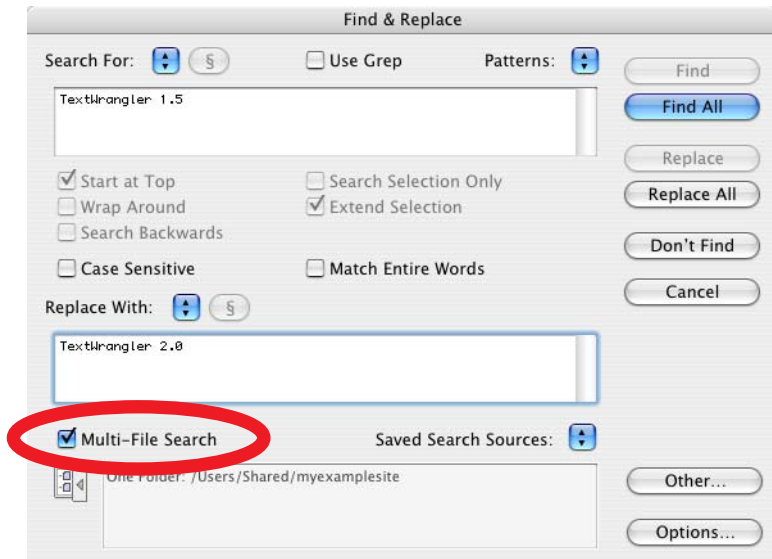
3 Type the string you are looking for in the Search For text field.

4 Type the replace string (if any) in the Replace With text field.

Be sure to read the section "Multi-File Replacing" later in this chapter if you use the replacement features.

5 Mark any checkboxes in the middle part of the dialog box that you want to apply to your search.

To learn more about these options, see "Search Settings" earlier in this chapter.

6 If you want TextWrangler to find only files that *do not* contain the search string, select the Exclude Matches checkbox.

7 Drag a folder to the search target area to search its contents, or open the Sources drawer specify the set of files to search.

See "Specifying the Search Set" later in this chapter for more information.

8 Click one of the buttons along the right side of the dialog box to begin the search.

The table below tells you what each of the buttons does.

This button…	Does this…
Find	Finds the first occurrence of the search string in the frontmost document. (Not available when the multi-file search option is selected.)
Find All	Finds all occurrences of the search string in all the files in the search sources set. TextWrangler displays the results in a search results browser.
Replace	Replaces the current selection in the frontmost document with the replace string. (Not available when the multi-file search option is selected.)
Replace All	Finds all occurrences of the search string in all the files in the search set and replaces them with the replace string.
Don't Find	Preserves the current search and replace strings and settings of the Find & Replace dialog without performing a search.
Cancel	Does not do the search and restores the search and replace strings and settings to their state prior to display of the Find & Replace dialog

Multi-File Search Results

When you perform a multi-file search, TextWrangler displays every occurrence of the search string in the set of files searched using a search results browser.
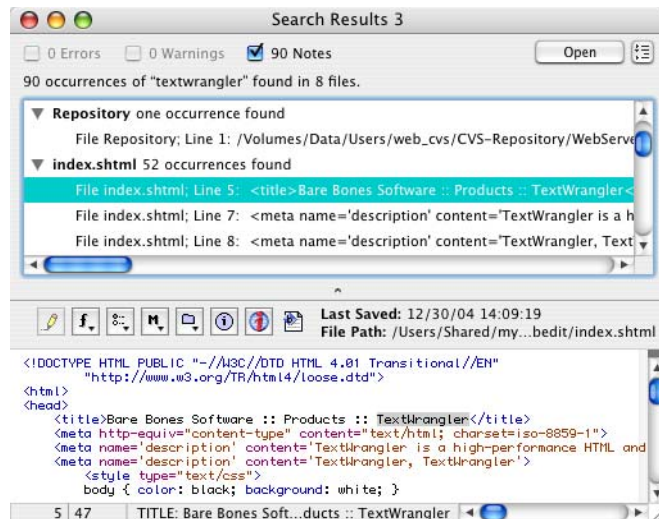
The items at the top of the window tells you how many matches TextWrangler found in the set of files you specified, as well as specifying whether there were any error conditions or warnings generated during the search. You can display or hide any combination of errors, warnings, and matches, by checking the appropriate options.

The middle panel lists each line that contains the matched text. (Depending on how you have configured TextWrangler, this list may be a Finder-style hierarchical list, where each match in a file is listed under the file's name, or a flat list where each occurrence is simply displayed in order.) Every match is identified by file name and line number.

To see the contents of a file, click a line in the list of matched searches. The part of the file that contains the search string appears in the lower pane. You cannot edit text in a search results browser. To edit a file, double-click the line that contains the match you want to edit, and the specified file will open at the point of the associated match.

After you have opened a file, you can use the Find Again, Replace, Replace All, and Replace & Find Again commands in the Search menu to continue searching it, as if you had chosen a File by File search. See the next section for information on File by File searching.

Note   You can use a search results window as the basis of another multi-file search. See "Specifying the Search Set" below.

Specifying the Search Set

When you turn on the Multi-File Search option, the Sources drawer of the Find & Replace dialog will open. You can also open or close the Sources drawer at any time by clicking the drawer toggle control at the lower left of the dialog. Use this drawer to specify which files and folders TextWrangler should examine when performing a multi-file search.
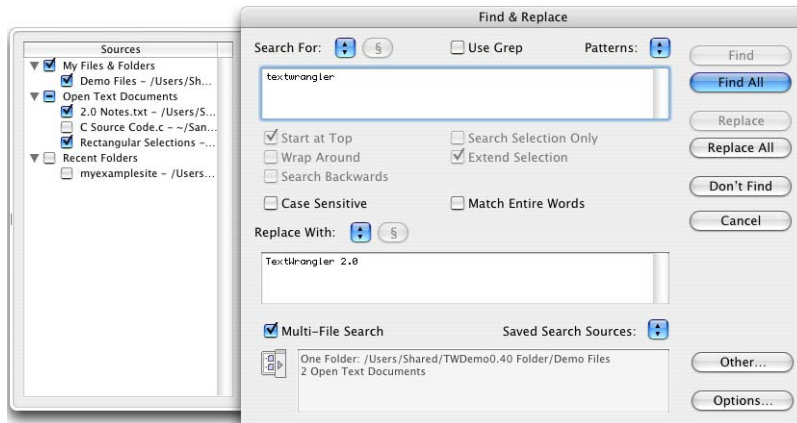
 You can choose multiple sources for a multi-file search, and you can mix different types of sources. Available sources include:

- open text documents

- the files in any selected or recently-searched folder

- the files listed in a search results browser

- any saved search targets (listed under "My Files & Folders")

To add a file, folder, or other suitable item to the Sources drawer, click Other in the Find & Replace dialog, and choose the item using the resulting selection sheet. (You can select multiple items to be added.)

To designate any item in the drawer as a search source, click on the box next to its name, or double-click on the name, to add a checkmark. To deselect a search source, click the box next to its name, or double-click its name, to turn off the checkmark. TextWrangler will display a summary of the selected sources in the search items box of the dialog. Here are some common scenarios.

Searching the files in a folder

To search the files in a folder, click on the box next to the folder's name, or double-click its name, in the Sources drawer. If the folder you want to search is not in the Sources drawer, click the Other button at the right of the dialog and pick the folder using the resulting selection sheet.

You can also drag a folder from the Finder directly into the search items box of the Find & Replace dialog to choose it as the source.

Searching all open documents

You can choose any or all open text documents as search sources. This option allows you to search documents that have not yet been saved to a file, or which contain unsaved changes. To choose all open documents, click the box next to the Open Documents item, or double-click on the item, in the Sources drawer.
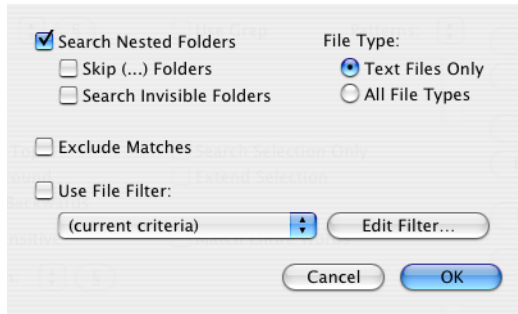
Searching the files contained in a results browser

If a previous multi-file search found many files that contain your search string, you may want to narrow the search. To search the files listed in any results browser window, click the box next to that browser's name, or double-click on its name, in the Sources drawer. You can also click the box next to the Results Browsers item, or double-click on this item, to search the files listed in all results browsers.

Note   The Choose a Folder dialog will display any packages it encounters as folders (rather than just as single files, the way they appear in the Finder). This allows you to navigate their internal structure just as you would any other folder. Similarly, you can drag a package from the Finder into the path box in the Find & Replace dialog and it will be treated as a true folder rather than as a single file.

Multi-File Search Options

Click the Options button to display the search options sheet.

To search the contents of all subfolders within the folders you choose, select the Search Nested Folders option in the resulting sheet. You can also choose to skip any folders whose names are enclosed in parentheses here by selecting the Skip (…) Folders option, or whether to search the contents of invisible folders by selecting the Search Invisible Folders option.

You can also choose to search only text files or to search all file types. If you have image files or other non-text files in search source folders, it may be a good idea to restrict the search to only text files. This setting is applied in addition to any file filter (see next section) and in fact takes effect *before* the filter.

To find only files whose contents do **not** contain the search string, select the Exclude Matches option.

You can further restrict which files from the chosen sources will be searched by applying a file filter. See "File Filters" (below) for more details.

File Filters

If you do not want to search every file in the set you selected, but want to include only those that meet certain criteria (such as those created on a certain date, or only those created by TextWrangler and not some other program, or those that are Perl documents), you can use a file filter. Mark the Use File Filter checkbox in the search options sheet and then choose the desired file filter from the Filters pop-up menu. If none of the filters you have already defined meets your needs, you can define a new one, or create a temporary filter.

New Filter

To define a new saved file filter, select New Filter from the pop-up menu. TextWrangler will ask you for a filter name; then the Edit File Filter dialog, below, appears.
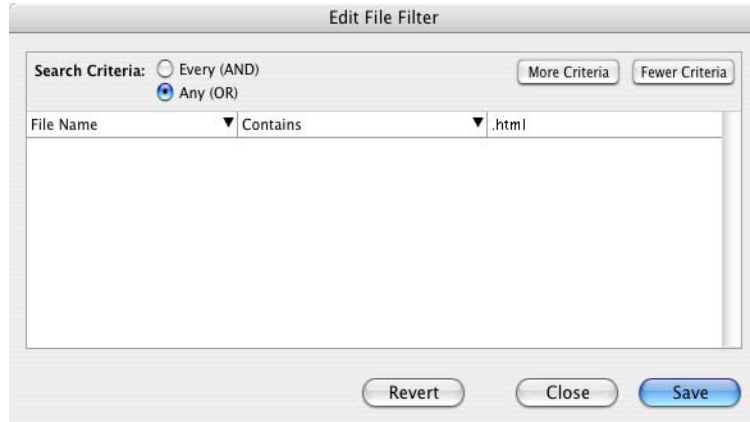
Note If you have the Preferences window open, you will not be able to add filters in the Find & Replace dialog. To work around this, close the Preferences window before using the Find & Replace dialog to add new filters.

The Edit File Filter dialog lets you specify multiple criteria that determine whether a given file is selected by the filter. You can choose whether these criteria are exclusive (that is, whether a file must meet *every* listed test to be selected) or inclusive (that is, whether a file that meets *any* of the tests is selected) using the Every (AND) and Any (OR) radio buttons at the top of the dialog.

To add a test, click More Criteria. A new row appears in the dialog, as shown below.
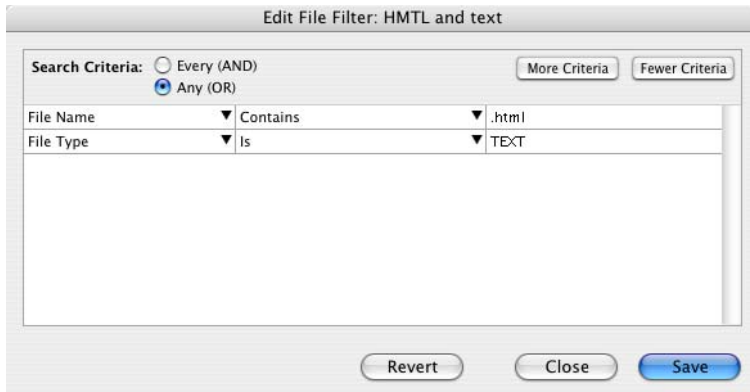
The leftmost column lets you specify which attribute of a file you wish to test. TextWrangler lets you test a file's name, the name of its enclosing folder, its creator or type, its creation and modification date (or both date and time), or its Finder label, visibility, or the programming or markup language it is written in. You can also test the content of a file, using the "Contents" criterion.

The middle column lets you choose the test to be applied to the selected attribute. The available options here change depending on what attribute you selected. If you choose Visibility in the first column, for instance, your only choices are whether the file is or is not visible, However, if you choose File Name in the first column, the middle column lets you test to see if the name does or does not exactly match, contain, begin with, or end with a particular string. You can also test file names to see if they match wildcard or Grep patterns.

Note   In wildcard patterns, the asterisk (*) and question mark (?) characters have special meanings. The asterisk matches any number of characters, such that "*.c" matches any file whose name ends with ".c". The question mark matches a single character, so that "foo?" matches "food", fool", "foot", and many other words. Both the asterisk and the question mark can be used anywhere in a wildcard pattern, and any number of either can be used in a single pattern.

Grep patterns, also known as regular expressions, are a powerful method of selecting file names based on classes of text or repeating text. They are covered in great detail in the next chapter.

The right column specifies the match criterion. For example, when filtering by File Name, you type the text you want the name to match, contain, begin with, or end with (or not). When filtering by Language, you choose a supported language from a pop-up menu.

Tip   To quickly set a match criterion, drag a file from the Finder to a field in the Define File Filter dialog. TextWrangler inserts the appropriate attribute from the dragged file, if possible.

Specifying Time and Date Criteria

When using a time or date criterion, you can use the special words below to specify dates and times relative to the current date and time.

Word	Means…
now	current date and time
today	midnight on the current date
yesterday	current date and time minus 24 hours
tomorrow	current date and time plus 24 hours

You can add any number of criteria using the More Criteria button. To delete the last criterion, click the Fewer Criteria button. To select any single criterion for deletion, press the Option key and click on the desired item. To select multiple continuous criteria, press Option-Shift and drag across the items, or to select discontinuous criteria, press Command-Option and click on the desired items.

Click Save to save the file filter and use it for this search. TextWrangler will ask you to name the filter, and it will then appear in the Filters pop-up menu in the Find & Replace dialog (and in the Define File Filter dialog). Click Revert to undo any changes you have made to the filter. (Hold the Option key when you click Revert to skip the confirmation alert.)

Filtering by Name

In order to provide the greatest possible flexibility, TextWrangler offers several different criteria for filtering based on file names

File Name: Tests the complete string corresponding to the file name.

File Name Root: Tests only the "root" portion of the file name. Given a name of the form "foo.txt", the root is the string which occurs before the period, in this case "foo".

File Name Suffix: Tests only the file name suffix. In the above example, the suffix is "txt". Note that the suffix does not include the period.

Temporary Filters

Choose "(current criteria)" from the pop-up menu in the Find & Replace dialog to reuse the last set of criteria applied (either from using a saved filter, or from using the Edit button to define criteria). Thus, you can use filter criteria on the fly, without the need to create and store a throwaway filter.

Editing and Deleting Filters

To edit a file filter you have already defined, choose it from the Filters pop-up menu, change it as desired, and click Save. Since each filter must have a unique name, saving it will replace the old version of the filter. To delete a filter entirely, visit the File Filters panel in the Preferences window. (You can also create or modify filters there.)

Multi-File Replacing

If you want to replace only some occurrences of text in multiple files, you can simply search those files, select the instances you want to change in the search results browser to open the files to those points, and perform the replacements individually. However, TextWrangler can also change *all* occurrences of a string in a group of files with one command.

Globally replacing text in more than one file works the same as replacing it in a single file. The only possible complication is that, if you make a mistake, it can have much wider consequences. If you are not sure what effect a replace operation will have, test it out on a few sample files, or a copy of your data, first!

To do a multi-file search and replace, replacing all occurrences:

1 **Set up the find and replace strings in the Find & Replace dialog as described in the section "Multi-File Search."**

2 **Choose the files to be searched as described in "Specifying the Search Set".**

3 Click Replace All in the Find & Replace dialog, or use its key equivalent of Command-Option-R.

TextWrangler displays the Find & Replace All Matches dialog box:
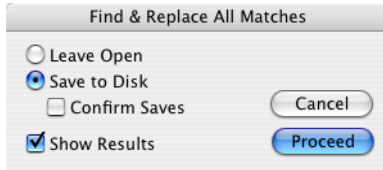
This is what each of the options does:

This option…	Replaces all occurrences of the search string with the replace string and…
Leave Open	Leaves all the files open so that you can inspect the replacements. If there are many files that contain the search string, TextWrangler may run out of memory.
Save to Disk	Saves each file with the changes. When the Confirm Saves setting is active, you will have an opportunity to approve the changes before TextWrangler saves them to disk. You should *not* turn this off unless you are sure that the replace operation being done is what you want.
Show Results	Opens a results browser listing each of the files which was changed, and the number of changes in each file.

Quick Search

The Quick Search command performs an incremental search. In other words, it shows the matching text as you type the search string, so you only have to type until you find the text you want.

To use Quick Search:

1 Choose Quick Search from the Search menu.

2 Type the string you are looking for in the Quick Search window.
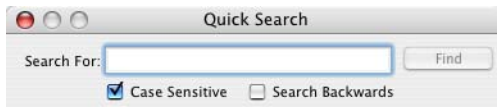
As you type, TextWrangler selects the first occurrence of what you have typed so far.

3 To find the next occurrence of the matching text, click Find Again, or press the Return or Enter keys.

You can use the Case Sensitive and Find Backwards options to change the way Quick Search looks for text. To clear the most recent word of the search string, you can press Option-Delete, or to clear the entire search string, you may use any of these shortcuts:

- the Clear command on the Edit menu
- the Clear key on the numeric keypad
- the Escape key

You can keep the Quick Search window open all the time and bring it to the front whenever you want to search. Quick Search always searches in the text view of the window immediately behind the Quick Search window.

Search Menu Reference

This section describes all of the commands in the Search menu.

Find

Opens the Find & Replace dialog. You can set the search and replace strings, choose searching options, and, for a multi-file search, specify the set of files to search.

Quick Search

Opens the Quick Search window. You can use this window to interactively search for text strings, as described in the previous section.

Find Again

Uses the previous settings of the Find & Replace dialog to search for the next occurrence of the search string.

Tip   Hold down the Shift key to search in the opposite direction from the current search direction. For example, if you have set the Backwards option in the Find & Replace dialog, holding down Shift while performing a Find Again will search *forward* in the document.

Find Selection

Uses the selected text as the search string and finds the next occurrence of the selected text. This command is the same as using the Enter Search String command followed by the Find Again command.

Tip   Hold down the Shift key to search in the opposite direction from the current search direction, just as for Find Again. Hold down the Option and Command keys as you double-click on a word to search for the next occurrence of that word.

Enter Search String

Choose the Enter Search String command to enter the currently selected text into the Find & Replace dialog as the search string (without opening the dialog). This command does not perform a search, but only sets the contents of the search string.

Enter Replace String

Choose the Enter Search String command to enter the currently selected text into the Find & Replace dialog as the replace string (without opening the dialog). This command does not perform a search, but only sets the contents of the replace string.

Replace

Replaces the selected text (usually an occurrence of the search string) with the replace string.

Replace All

Replaces every occurrence of the search string in a file with the replace string.

Replace & Find Again

Replaces the selected text with the replace string and searches for the next occurrence of the search string.

Go to Line

When you choose this command, TextWrangler opens the Go To Line dialog box. Type in a line number and the frontmost text window will jump to display that line.

Note   The Go To Line command honors the "Use 'Hard' Line Numbering in Soft-Wrapped Text Views" option in the Editing: General preferences panel.

Go to Center Line

Will move the insertion point to the beginning of the middle or center line of the displayed text.

Go to Function Start/End

When you choose one of these commands, BBEdit will move the insertion point to a position immediately before the start or immediately after the end of the current function, where a function is any item which appears on the function pop-up menu. If you anticipate using these commands often, you may wish to assign them key equivalents by using the Set Menu Keys command from the BBEdit menu.

Go to Previous/Next Function

When you choose one of these commands, BBEdit will select the name of the previous or next function in the document, where a function is any item which appears on the function pop-up menu. If you anticipate using these commands often, you may wish to assign them key equivalents by using the Set Menu Keys command from the BBEdit menu.

Find Differences

Finds the differences between two files, or all of the files contained in two folders. See "Comparing Text Files" in Chapter 4 for more details.

Compare Two Front Documents

Performs a Find Differences on the two frontmost text documents.

Compare Against Disk File

Performs a Find Differences between the contents of the front document and the disk file for that same document. This capability makes it easy to locate in-progress changes to a document.

Apply to New

Applies the currently selected difference to the "New" version of two files which are being compared. See "Comparing Text Files" for more details.

Apply to Old

Applies the currently selected difference to the "Old" version of two files which are being compared. See "Comparing Text Files" for more details.

Compare Again

Find the differences between two files, using the same settings that were used in the last time you used the Find Differences command. See "Comparing Text Files" for more details.

Find in Reference

Performs a search using the selected symbol on the Apple Developer Connection web site. If there is no selection, TextWrangler will attempt to determine the symbol name by inspecting the text around the insertion point.

If you wish to modify the URL template which TextWrangler uses to perform the lookup, you may do so by issuing a 'defaults write' command in the Terminal as follows:

```
defaults write com.barebones.textwrangler
Services:ADCReferenceSearchTemplate <template>
```

In the template, use "%@" to indicate where the selected symbol name should be inserted in the lookup. For example, to make Find in Reference use Google, you could specify:

```
defaults write com.barebones.textwrangler
Services:ADCReferenceSearchTemplate "http://www.google.com/
search?q=%@&ie=UTF-8"
```

8

Searching with Grep

This chapter describes the Grep option in TextWrangler's Find command, which allows you to find and change text that matches a set of conditions you specify. Combined with the multi-file search and replace features described in Chapter 7, TextWrangler's grep capabilities can make many editing tasks quicker and easier, whether you are extracting data from a file, rearranging source code, or just updating a phone list.

In this chapter

What Is Grep or Pattern Searching?

Grep patterns offer a powerful way to make changes to your data that "plain text" searches simply cannot. For example, suppose you have a list of people's names that you want to alphabetize. If the names appear last name first, you can easily put these names in a TextWrangler window and use the Sort tool. But if the list is arranged first name first, a simple grep pattern can be used to put the names in the proper order for sorting.

A grep pattern, also known as a regular expression, describes the text that you are looking for. For instance, a pattern can describe words that begin with C and end in l. A pattern like this would match "Call", "Cornwall", and "Criminal" as well as hundreds of other words.

In fact, you have probably already used pattern searching without realizing it. The Find & Replace dialog's "Match Case" and "Entire Word" options turn on special searching patterns. Suppose that you are looking for "corn". With the "Match Case" option turned off, you are actually looking for a pattern that says: look for a C or c, O or o, R or r, and N or n. With the "Entire Word" option on, you are looking for the string "corn" only if it is surrounded by white space or punctuation characters; special search characters, called metacharacters, are added to the search string you specified to indicate this.

What makes pattern searching counterintuitive at first is how you describe the pattern. Consider the first example above, where we want to search for text that begins with the letter "C" and ends with the letter "l" with any number of letters in between. What exactly do you put between them that means "any number of letters"? That is what this chapter is all about.

Note   Grep is the name of a frequently used Unix command that searches using regular expressions, the same type of search pattern used by TextWrangler. For this reason, you will often see regular expressions called "grep patterns," as TextWrangler does. They're the same thing.

Recommended Books and Resources

Mastering Regular Expressions, 2nd Edition
by Jeffrey E.F. Friedl. O'Reilly & Associates, 2002. ISBN 0-596-00289-0

Although it does not cover TextWrangler's grep features specifically, *Mastering Regular Expressions* is an outstanding resource for learning the "how-to" of writing useful grep patterns, and the new second edition is even better than the original.

TextWrangler-Talk
The TextWrangler-Talk online mailing list covers a wide range of topics and questions about using TextWrangler, which frequently include searching and the use of grep patterns. To subscribe to this list, please visit the support section of our web site, which offers the option to sign up.

```
http://www.barebones.com/support/lists.html
```

TextWrangler's grep engine is based on the PCRE library package, which is open source software, written by Philip Hazel, and copyright 1997-2000 by the University of Cambridge, England. For details, see <ftp://ftp.csx.cam.ac.uk/pub/software/programming/pcre/>.

Writing Search Patterns

This section explains how to create search patterns using TextWrangler's grep syntax. For readers with prior experience, this is essentially like the syntax used for regular expressions in the Perl programming language. (However, you *do not* need to understand anything about Perl in order to make use of TextWrangler's grep searching.)

Most Characters Match Themselves

Most characters that you type into the Find & Replace dialog match themselves. For instance, if you are looking for the letter "t", Grep stops and reports a match when it encounters a "t" in the text. This idea is so obvious that it seems not worth mentioning, but the important thing to remember is that these characters *are* search patterns. Very simple patterns, to be sure, but patterns nonetheless.

Escaping Special Characters

In addition to the simple character matching discussed above, there are various special characters that have different meanings when used in a grep pattern than in a normal search. (The use of these characters is covered in the following sections.)

However, sometimes you will need to include an exact, or literal, instance of these characters in your grep pattern. In this case, you must use the backslash character \ before that special character to have it be treated literally; this is known as "escaping" the special character. To search for a backslash character itself, double it \\ so that its first appearance will escape the second.

For example, perhaps the most common "special character" in grep is the dot: ".". In grep, a dot character will match any character except a return. But what if you only want to match a literal dot? If you escape the dot: "\.", it will only match another literal dot character in your text.

So, most characters match themselves, and even the special characters will match themselves if they are preceded by a backslash. TextWrangler's grep syntax coloring helps make this clear.

Note When passing grep patterns to TextWrangler via AppleScript, be aware that both the backslash and double-quote characters have special meaning to AppleScript. In order to pass these through correctly, you must escape them in your script. Thus, to pass \r for a carriage return to TextWrangler, you must write \\r in your AppleScript string.

Wildcards Match Types of Characters

These special characters, or metacharacters, are used to match certain types of other characters:

Wildcard	Matches…
.	any character except a line break (that is, a carriage return)
^	beginning of a line (unless used in a character class)
$	end of line (unless used in a character class)

Being able to specifically match text starting at the beginning or end of a line is an especially handy feature of grep. For example, if you wanted to find every instance of a message sent by Patrick, from a log file which contains various other information like so:

```
From: Rich, server: barebones.com
To:   TextWrangler-Talk, server: lists.barebones.com
From: Patrick, server: example.barebones.com
```

you could search for the pattern:

```
^From: Patrick
```

and you will find every occurrence of these lines in your file (or set of files if you do a multi-file search instead).

It is important to note that ^ and $ do not actually match return characters. They match zero-width *positions* after and before returns, respectively. So, if you are looking for "foo" at the end of a line, the pattern "foo$" will match the three characters "f", "o", and "o". If you search for "foo\r", you will match the same text, but the match will contain four characters: "f", "o", "o", and a return.

In soft-wrapped files, ^ and $ will also match after and before soft line breaks, respectively.

You can combine ^ and $ within a pattern to force a match to constitute an entire line. For example:

```
^foo$
```

will only match "foo" on a line by itself, with no other characters. Try it against these three lines to see for yourself:

```
foobar
foo
fighting foo
```

The pattern will only match the second line.

Other Positional Assertions

TextWrangler's grep engine supports additional positional assertions, very similar to ^ and $.

Escape	Matches
\A	only at the beginning of the document (as opposed to ^, which matches at the beginning of the document and also at the beginning of each line)
\b	any word boundary, defined as any position between a \w character and a \W character, in either order
\B	any position that is *not* a word boundary
\z	at the end of the document (as opposed to $, which matches at the end of the document and also at the end of each line)
\Z	at the end of the document, or before a trailing return at the end of the doc, if there is one

Examples (the text matched by the pattern is underlined)

Search for: \bfoo\b
Will match: bar <u>foo</u> bar
Will match: <u>foo</u> bar
Will not match: foobar

Search for: \bJane\b
Will match: <u>Jane</u>'s
Will match: Tell <u>Jane</u> about the monkey.

Search for: \Afoo
Will match: <u>foo</u>bar
Will not match: This is good foo.

Character Classes Match Sets or Ranges of Characters

The character class construct lets you specify a set or a range of characters to match, or to ignore. A character class is constructed by placing a pair of square brackets […] around the group or range of characters you wish to include. To exclude, or ignore, all characters specified by a character class, add a caret character ^ just after the opening bracket [^…]. For example:

Character Class	Matches
`[xyz]`	any one of the characters x, y, z
`[^xyz]`	any character except x, y, z
`[a-z]`	any character in the range a to z

You can use any number of characters or ranges between the brackets. Here are some examples:

Character Class	Matches
`[aeiou]`	any vowel
`[^aeiou]`	any character that is not a vowel
`[a-zA-Z0-9]`	any character from a-z, A-Z, or 0-9
`[^aeiou0-9]`	any character that is neither a vowel nor a digit

A character class matches when the search encounters any *one* of the characters in the pattern. However, the contents of a set are only treated as separate characters, not as words. For example, if your search pattern is `[beans]` and the text in the window is "lima beans", TextWrangler will report a match at the "a" of the word "lima".

To include the character] in a set or a range, place it immediately after the opening bracket. To use the ^ character, place it anywhere except immediately after the opening bracket. To match a dash character (hyphen) in a range, place it at the beginning of the range; to match it as part of a set, place it at the beginning or end of the set. Or, you can include any of these character at any point in the class by escaping them with a backslash.

Character Class	Matches
`[]0-9]`	any digit or]
`[aeiou^]`	a vowel or ^
`[-A-Z]`	a dash or A - Z
`[--A]`	any character in the range from - to A
`[aeiou-]`	any vowel or -
`[aei\-ou]`	any vowel or -

Character classes respect the setting of the Case Sensitive checkbox in the Find & Replace dialog. For example, if Case Sensitive is on, `[a]` will only match "a"; if Case Sensitive is off, `[a]` will match both "a" and "A".

Matching Non-Printing Characters

As described in Chapter 7 on searching, TextWrangler provides several special character pairs that you can use to match common non-printing characters, as well as the ability to specify any arbitrary character by means of its hexadecimal character code (escape code). You can use these special characters in grep patterns as well as for normal searching.

For example, to look for a tab or a space, you would use the character class `[\t ]` (consisting of a tab special character and a space character).

Character	Matches
\r	line break (carriage return)
\n	Unix line break (line feed)
\t	tab
\f	page break (form feed)
\a	alarm (hex 07)
\cX	a named control character, like \cC for Control-C
\b	backspace (hex 08) *(only in character classes)*
\e	Esc (hex 1B)
\x*NN*	hexadecimal character code *NN* (for example, \x0D for CR)
\x{*NNNN*}	any number of hexadecimal characters *NN…* (for example, \x{0} will match a null, \x{304F} will match a Japanese Unicode character)
\\	backslash

Use \r to match a line break in the middle of a pattern and the special characters ^ and $ (described above) to "anchor" a pattern to the beginning of a line or to the end of a line. In the case of ^ and $, the line break character is not included in the match.

Other Special Character Classes

TextWrangler uses several other sequences for matching different types or categories of characters.

Special Character	Matches
\s	any whitespace character (space, tab, carriage return, line feed, form feed)
\S	any non-whitespace character (any character not included by \s)
\w	any word character (a-z, A-Z, 0-9, _, and some 8-bit characters)
\W	any non-word character (all characters not included by \w, including carriage returns)
\d	any digit (0-9)
\D	any non-digit character (including carriage return)

A "word" is defined in TextWrangler as any run of non-word-break characters bounded by word breaks. Word characters are generally alphanumeric, and some characters whose value is greater than 127 are also considered word characters.

Note that any character matched by \s is by definition not a word character; thus, anything matched by \s will also be matched by \W (but not the reverse!).

Quantifiers Repeat Subpatterns

The special characters *, +, and ? specify *how many times* the pattern preceding them may repeat. {}-style quantifiers allow you to specify exactly how many times a subpattern can repeat. The preceding pattern can be a literal character, a wildcard character, a character class, or a special character.

Pattern	Matches
p*	zero or more p's
p+	one or more p's
p?	zero or one p's
p{COUNT}	match exactly COUNT p's, where COUNT is an integer
p{MIN,}	match at least MIN p's, where MIN is an integer
p{MIN, MAX}	match at least MIN p's, but no more than MAX

Note that the repetition characters * and ? match *zero or more* occurrences of the pattern. That means that they will *always* succeed, because there will always be at least zero occurrences of any pattern, but that they will not necessarily select any text (if no occurrences of the preceding pattern are present).

For this reason, when you are trying to match more than one occurrence, it is usually better to use a + than a *, because + *requires* a match, whereas * can match the empty string. Only use * when you are sure that you really mean "zero or more times," not just "more than once."

Try the following examples to see how their behavior matches what you expect:

Pattern	Text	Matches
`.*`	Fourscore and seven years	Fourscore and seven years
`[0-9]+`	I've been a loyal member since 1983 or so.	1983
`\d+`	I've got 12 years on him.	12
`A+`	BAAAAAAAB	AAAAAAA
`A{3}`	BAAAAB	AAA *(first three A's)*
`A{3,}`	BAAAAB	AAAA
`A{1,3}`	BAAAAB	AAA on the first match, the remaining A on the second match
`c?andy`	andy likes candy	"andy" on the first match, "candy" on the second
`A+`	Ted joined AAA yesterday	"AAA" on the first match; "a" from "yesterday" on the second

Combining Patterns to Make Complex Patterns

So far, the patterns you have seen match a single character or the repetition of a single character or class of characters. This is very useful when you are looking for runs of digits or single letters, but often that is not enough.

However, by combining these patterns, you can search for more complex items. As it happens, you are already familiar with combining patterns. Remember the section at beginning of this discussion that said that each individual character is a pattern that matches itself? When you search for a word, you are already combining basic patterns.

You can combine any of the preceding grep patterns in the same way. Here are some examples.

Pattern	Matches	Examples
`\d+\+\d+`	a string of digits, followed by a literal plus sign, followed by more digits	4+2 1234+5829
`\d{4}[\t ]B\.C\.`	four digits, followed by a tab or a space, followed by the string B.C.	2152 B.C.
`\$?[0-9,]+\.\d*`	an optional dollar sign, followed by one or more digits and commas, followed by a period, then zero or more digits	1,234.56 $4,296,459.19 $3,5,6,4.0000 0. *(oops!)*

Note again in these examples how the characters that have special meaning to grep are preceded by a backslash (\+, \., and \$) when we want them to match themselves.

Creating Subpatterns

Subpatterns provide a means of organizing or grouping complex grep patterns. This is primarily important for two reasons: for limiting the scope of the alternation operator (which otherwise creates an alternation of everything to its left and right), and for changing the matched text when performing replacements. A subpattern consists of any simple or complex pattern, enclosed in a pair of parentheses:

Pattern	Matches
`(p)`	the pattern *p* and remembers it

You can combine more than one subpattern into a grep pattern, or mix subpatterns and other pattern elements as you need.

Taking the last set of examples, you could modify these to use subpatterns wherever actual data appears:

Pattern	Matches	Examples
`(\d+)\+(\d+)`	a string of digits, followed by a plus sign, followed by more digits	4+2 1234+5829
`(\d{4})[\t ]B\.C\.`	four digits, followed by a tab or a space, followed by the string B.C.	2152 B.C.
`\$?([0-9,]+)\.(\d*)`	an optional dollar sign, followed by one or more digits and commas, followed by a period, then zero or more digits	1,234.56 $4,296,459.19 $3,5,6,4.0000 0.

What if we wanted to match a series of digits, followed by a plus sign, followed by the exact same series of digits as on the left side of the plus? In other words, we want to match "1234+1234" or "7+7", but *not* "5432+1984".

Using grouping parentheses, you can do this by referring to a backreference, also known as a captured subpattern. Each set of parentheses in the pattern is numbered from left to right, starting with the opening parenthesis. Later in the pattern, you can refer to the text matched within these backreferences by using a backslash followed by the number of the backreference.

Pattern	Matches	Examples
`(\d+)\+\1`	a string of digits, followed by a plus sign, followed the same digits	7+7 1234+1234
`(\w+)\s+\1`	double words	the the
`(\w)(\w)\2\1`	a word character, a second word character, followed by the second one again and the first one again	abba

We will revisit subpatterns in the section on replacement, where you will see how the choice of subpatterns affects the changes you can make.

Using Alternation

The alternation operator | allows you to match any of several patterns at a given point. To use this operator, place it between one or more patterns x | y to match either x or y.

As with all of the preceding options, you can combine alternation with other pattern elements to handle more complex searches.

Pattern	Text is…	Matches…
`a|t`	A cat	each "a" and "t"
`a|c|t`	A cat	each "a", "c", and "t"
`a (cat|dog) is`	A cat is here. A dog is here. A giraffe is here.	"A cat is", "A dog is"
`A|b+`	Abba	"A", "bb", and "a"
`Andy|Ted`	Andy and Ted joined AAA yesterday	"Andy" and "Ted"
`\d{4}|years`	I've been a loyal member since 1983, almost 16 years ago.	"1983", "years"
`[a-z]+|\d+`	That's almost 16 years.	"That", "s", "almost", "16", "years"

The "Longest Match" Issue

When creating complex patterns, you should bear in mind that the quantifiers +, *, ? and {} are "greedy." That is, they will always make the longest possible match possible to a given pattern, so if your pattern is E+ (one or more E's) and your text contains "EEEE", the pattern matches all the E's at once, not just the first one. This is usually what you want, but not always.

Suppose, for instance, that you want to match an HTML tag. At first, you may think that a good way to do this would be to search for the pattern:

```
<.+>
```

consisting of a less-than sign, followed by one or more occurrences of a single character, followed by a greater-than sign. To understand why this may not work the way you think it should, consider the following sample text to be searched:

```
<B>This text is in boldface.</B>
```

The intent was to write a pattern that would match both of the HTML tags separately. Let's see what actually happens. The < character at the beginning of this line matches the beginning of the pattern. The next character in the pattern is . which matches any character (except a line break), modified with the + quantifier, taken together, this combination means one or more repetitions of any character. That, of course, takes care of the B. The problem is that the next > is also "any character" and that it *also* qualifies as "one or more repetitions." In fact, all of the text up to the end of the line qualifies as "one or more repetitions of any character" (the line break does not qualify, so grep stops there). After grep has reached the line break, it has exhausted the + operator, so it backs up and sees if it can find a match for >. Lo and behold, it can: the last character is a greater-than symbol. Success!

In other words, the pattern matches our entire sample line at once, *not the two separate HTML tags in it as we intended.* More generally, the pattern matches all the text in a given line or paragraph from the first < to the *last* >. The pattern only does what we intended when there is only one HTML tag in a line or paragraph. This is what we meant when we say that the regular quantifiers try to make the longest possible match.

Non-Greedy Quantifiers

To work around this "longest match" behavior, you can modify your pattern to take advantage of *non-greedy* quantifiers.

Quantifier	Matches…
+?	one or more
*?	zero or more
??	zero or one
{COUNT}?	match exactly *COUNT* times
{MIN,}?	match at least *MIN* times
{MIN, MAX}?	match at least *MIN* times, but no more than *MAX*

Astute readers will note that these non-greedy quantifiers correspond exactly to their normal (greedy) counterparts, appended with a question mark.

Revisiting our problem of matching HTML tags, for example, we can search for:

```
<.+?>
```

This matches an opening bracket, followed by one or more occurrences of any character other than a return, followed by a closing bracket. The non-greedy quantifier achieves the results we want, preventing TextWrangler from "overrunning" the closing angle bracket and matching across several tags.

A slightly more complicated example: how could you write a pattern that matches all text between <B> and </B> HTML tags? Consider the sample text below:

```
<B>Welcome</B> to the home of <B>TextWrangler!</B>
```

As before, you might be tempted to write:

```
<B>.*</B>
```

but for the same reasons as before, this will match the entire line of text. The solution is similar; we will use the non-greedy `*?` quantifier:

```
<B>.*?</B>
```

Writing Replacement Patterns

Subpatterns Make Replacement Powerful

We covered subpatterns earlier when discussing search patterns and discussed how the parentheses can be used to limit the scope of the alternation operator. Another reason for employing subpatterns in your grep searches is to provide a powerful and flexible way to change or reuse found information as part of a search-and-replace operation. If you do not use subpatterns, you can still access the complete results of the search with the & metacharacter. However, this precludes reorganizing the matched data as it is replaced.

Pattern	Matches…
&	the entire matched pattern [replacement only]
(p)	the pattern p and remembers it [search only]
\1, \2, …, \99	the nth subpattern in the entire search pattern

Note   TextWrangler will now remember up to 99 backreferenced subpatterns. Versions prior to 6.5 were limited to 9 subpatterns.

Using the Entire Matched Pattern

The & character is useful when you want to use the entire matched string as the basis of a replacement. Suppose that in your text every instance of product names that begin with the company name "ACME" needs to end with a trademark symbol (™). The following search pattern finds two-word combinations that begin with "ACME":

```
ACME [A-Za-z]+
```

The following replacement string adds the trademark symbol to the matched text:

```
&™
```

For example, if you start with

```
ACME Magnets, ACME Anvils, and ACME TNT are all premium
products.
```

and perform a replace operation with the above patterns, you will get:

```
ACME Magnets™, ACME Anvils™, and ACME TNT™ are all premium
products.
```

Using Parts of the Matched Pattern

While using the entire matched pattern in a replacement string is useful, it is often more useful to use only a portion of the matched pattern and to rearrange the parts in the replacement string.

For example, suppose a source file contains C-style declarations of this type:

```
#define Util_Menu 284
#define Tool_Menu 295
```

and you want to convert them so they look like this, Pascal-style:

```
const int Util_Menu = 284;
const int Tool_Menu = 295;
```

The pattern to find the original text is straightforward:

```
#define[ \t]+.+[ \t]+\d+[^0-9]*$
```

This pattern matches the word "#define" followed by one or more tabs or spaces, followed by one or more characters of any type, followed by one or more tabs or spaces, followed by one or more digits, followed by zero or more characters that are *not* digits (to allow for comments), followed by the end of the line.

The problem with this pattern is that it matches the entire line. It does not provide a way to remember the individual parts of the found string.

If you use subpatterns to rewrite the above search pattern slightly, you get this:

```
#define[ \t]+(.+)[ \t]+(\d+)[^0-9]*$
```

The first set of parentheses defines a subpattern which remembers the name of the constant. The second set remembers the value of the constant.

The replacement string would look like this:

```
const int \1 = \2;
```

The sequence \1 is replaced by the name of the constant (the first subpattern from the search pattern), and the sequence \2 is replaced by the value of the constant (from the second subpattern).

Our example throws out any comment that may follow the C-style constant declaration. As an exercise, try rewriting the search and replace patterns so they preserve the comment, enclosing it in (*…*) style Pascal comment markers.

Here are some more examples:

Data	Search for	Replace	Result
4+2	(\d+)\+(\d+)	\2+\1	2+4
1234+5829	(\d+)\+(\d+)	\1+\1	1234+1234
2152 B.C.	(\d{4})[\t]B\.C\.	\1 A.D.	2152 A.D.
1,234.56	\$?([0-9,]+)\.(\d+)	\1 dollars and \2 cents	1,234 dollars and 56 cents
$4,296,459.19	\$?([0-9,]+)\.(\d+)	\1 dollars and \2 cents	4,296,459 dollars and 19 cents
$3,5,6,4.00000	\$?([0-9,]+)\.(\d+)	\1 dollars and \2 cents	3,5,6,4 dollars and 00000 cents

Case Transformations

Replace patterns can also change the case of the original text when using subpattern replacements. The syntax is similar to Perl's, specifically:

Modifier	Effect
\u	Make the next character uppercase
\U	Make all following characters uppercase until reaching another case specifier (\u, \L, \l) or \E
\l	Make the next character lowercase
\L	Make all following characters lowercase until reaching another case specifier (\u, \U, \l) or \E
\E	End case transformation opened by \U or \L

Here are some examples to illustrate how case transformations can be used.

Given some text:

```
mumbo-jumbo
```

and the search pattern:

```
(\w+)(\W)(\w+)
```

the following replace patterns will produce the following output:

```
\U\1\E\2\3    MUMBO-jumbo
\u\1\2\u\3                Mumbo-Jumbo
```

Note that case transformations also affect literal strings in the replace pattern:

```
\U\1\2fred             MUMBO-FRED
\lMUMBLE\2\3           mUMBLE-jumbo
```

Finally, note that \E is not necessary to close off a modifier; if another modifier appears before an \E is encountered, that modifier will take effect immediately:

```
\Ufred-\uwilma         FRED-Wilma
```

Examples

The example patterns in this section describe some common character classes and shortcuts used for constructing grep patterns, and addresses some common tasks that you might find useful in your work.

Matching Identifiers

One of the most common things you will use grep patterns for is to find and modify identifiers, such as variables in computer source code or object names in HTML source documents. To match an arbitrary identifier in most programming languages, you might use this search pattern:

```
[a-z][a-zA-Z0-9]*
```

This pattern matches any sequence that begins with a lowercase letter and is followed by zero or more alphanumeric characters. If other characters are allowed in the identifier, add them to the pattern. This pattern allows underscores in only the first character of the identifier:

```
[a-z_][a-zA-Z0-9]*
```

The following pattern allows underscores anywhere *but* the first character, but allows identifiers to begin with an uppercase or lowercase letter:

```
[a-zA-Z][a-zA-Z0-9_]*
```

Matching White Space

Often you will want to match two sequences of data that are separated by tabs or spaces, whether to simply identify them, or to rearrange them.

For example, suppose you have a list of formatted label-data pairs like this:

```
User name:    Bernard Rubble
Occupation:   Actor
Spouse:       Betty
```

You can see that there are tabs or spaces between the labels on the left and the data on the right, but you have no way of knowing how many spaces or tabs there will be on any given line. Here is a character class that means "match one or more white space characters."

```
[ \t]+
```

So, if you wanted to transform the list above to look like this:

```
User name("Bernard Rubble")
Occupation("Actor")
Spouse("Betty")
```

You would use this search pattern:

```
([a-z ]+):[ \t]+([a-z ]+)
```

and this replacement pattern:

```
\1\("\2"\)
```

Matching Delimited Strings

In some cases, you may want to match all the text that appears between a pair of delimiters. One way to do this is to bracket the search pattern with the delimiters, like this:

```
".*"
```

This works well if you have only one delimited string on the line. But suppose the line looked like this:

```
"apples", "oranges, kiwis, mangos", "penguins"
```

The search string above would match the entire line. (This is another instance of the "longest match" behavior of TextWrangler's grep engine, which was discussed previously.)

Once again, non-greedy quantifiers come to the rescue. The following pattern will match "-delimited strings:

```
".+?"
```

Marking Structured Text

Suppose you are reading a long text document that does not have a table of contents, but you notice that all the sections are numbered like this:

```
3.2.7    Prehistoric Cartoon Communities
5.19.001  Restaurants of the Mesozoic
```

You can use a grep pattern to create marks for these headings, which will appear in the Mark pop-up menu. Choose Find & Mark All from the Mark pop-up menu in the status bar. Then, decide how many levels you want to mark. In this example, the headings always have at least two digits and at most four.

Use this pattern to find the headings:

```
^(\d+\.\d+\.?\d*\.?\d*)[ \t]+([a-z ]+)
```

and this pattern to make the file marks:

```
\1 \2
```

The ^ before the first search group ensures that TextWrangler matches the numeric string at the beginning of a line. The pattern

```
\.?\d*
```

matches a (possible) decimal point and a digit sequence. The other groups use the white space idiom and the identifier idiom. You can use a similar technique to mark any section that has a section mark that can be described with grep.

Marking a Mail Digest

You can elaborate the structured text technique to create markers for mail digests. Assume that each digest is separated by the following lines:

```
From: Sadie Burke <sadie@burke.com>
Date: Sun, 16 Jul 1995 13:17:45 -0700
Subject: Fishing with the judge
```

Suppose you want the marker text to list the subject and the sender. You would use the following search string:

```
^From:[ \t]+(.*)\r.*\rSubject:[ \t]+(.*)
```

And mark the text with this replacement string:

```
\2 \1
```

Note that for the sequence \r.*\r in the middle of the search string, the \r before "Subject" is necessary because as previously discussed, the special character . does not match carriage returns. (At least, not by default. See "Advanced Topics," below, for details on how to make dot match *any* character, including carriage returns.)

Rearranging Name Lists

You can use grep patterns to transform a list of names in first name first form to last name first order (for a later sorting, for instance). Assume that the names are in the form:

```
Junior X. Potter
Jill Safai
Dylan Schuyler Goode
Walter Wang
```

If you use this search pattern:

```
^(.*) ([^ ]+)$
```

And this replacement string:

```
\2, \1
```

The transformed list becomes:

```
Potter, Junior X.
Safai, Jill
Goode, Dylan Schuyler
Wang, Walter
```

Advanced Grep Topics

TextWrangler's new PCRE-based grep engine offers unparalleled syntactical power. The topics below cover areas that show how grep can effectively match very complicated patterns of text—matches which were impossible to achieve with older versions of TextWrangler. However, with this power comes complexity.

If you are new to grep, it is possible that the topics covered in this section will not make much sense to you. That's OK. The best way to learn grep is to use it in real life, not by reading example patterns. In many cases, the basic grep syntax covered previously in this chapter will be all that you need.

If you are an experienced user of grep, however, many of the topics covered below will be of great interest.

Matching Nulls

TextWrangler's grep engine is capable of searching for any sequence of characters, including sequences which contain null characters (ASCII value zero). Here's one way to match a null:

```
\x{0}
```

Backreferences

The following charts explain the rules TextWrangler uses for determining backreferences.

In Search Patterns

Modifier	Effect
\0	A backslash followed by a zero is an octal character reference. Up to two further octal characters are read. Thus, "\040" will match a space character, and "\07" will match the ASCII BEL (\x07), but "\08" will match an ASCII null followed by the digit 8 (because octal characters only range from 0-7).
\1-9	A backslash followed by a single decimal digit from 1 to 9 is always a backreference to the *Nth* captured subpattern.

Modifier	Effect
\10-99	A backslash followed by two decimal digits, which taken together form the integer N (ranging from 10 to 99), is a backreference to the *Nth* captured subpattern, *if* there exist *N* capturing sets of parentheses in the pattern. If there are fewer than *N* captured subpatterns, the grep engine will instead look for up to three octal digits following the backslash. Any subsequent digits stand for themselves.
	So, in a search pattern, "\11" is a backreference if there are 11 or more sets of capturing parentheses in the pattern. If not, it matches a tab. "\011" always matches a tab. "\81" is a backreference if there are 81 or more captured subpatterns, but matches an ASCII null followed by the two characters "8" and "1" otherwise.

In Character Classes

Modifier	Effect
OCTAL	Inside a character class, a backslash followed by up to three octal digits generates a single byte character reference from the least significant eight bits of the value. Thus, the character class "[\7]" will match a single byte with octal value 7 (equivalent to "\x07"). "[\8]" will match a literal "8" character.

In Replacement Patterns

Modifier	Effect
NNN+	If more than two decimal digits follow the backslash, only the first two are considered part of the backreference. Thus, "\111" would be interpreted as the 11th backreference, followed by a literal "1". You may use a leading zero; for example, if in your replacement pattern you want the first backreference followed by a literal "1", you can use "\011". (If you use "\11", you will get the 11th backreference, even if it is empty.)
NN	If two decimal digits follow the backslash, which taken together represent the value N, and if there is an Nth captured substring, then all three characters are replaced with that substring. If there is not an Nth captured substring, all three characters are discarded—that is, the backreference is replaced with the empty string.
N	If there is only a single digit N following the backslash and there is an Nth captured substring, both characters are replaced with that substring. Otherwise, both characters are discarded—that is, the backreference is replaced with the empty string. In replacement patterns, \0 is a backreference to the entire match (exactly equivalent to "&").

POSIX-Style Character Classes

TextWrangler provides support for POSIX-style character classes. These classes are used in the form `[:CLASS:]`, and are *only* available inside regular character classes (in other words, inside another set of square brackets).

Class	Meaning
alnum	letters and digits
alpha	letters
ascii	character codes 0-127
cntrl	control characters
digit	decimal digits (same as \d)
graph	printing characters, excluding spaces
lower	lower case letters
print	printing characters, including spaces
punct	punctuation characters
space	white space (same as \s)
upper	upper case letters
word	"word" characters (same as \w)
xdigit	hexadecimal digits

For example: `[[:digit:]]+` is the same as: `[\d]+`

POSIX-style character class names are case-sensitive.

It is easy to forget that POSIX-style character classes are only available inside regular character classes. The pattern `[:space:]`, without enclosing square brackets, is just a character class consisting of the characters ":", "a", "c", "e", "p", and "s".

The names "ascii" and "word" are Perl extensions; the others are defined by the POSIX standard. Another Perl extension supported by TextWrangler is negated POSIX-style character classes, which are indicated by a ^ after the colon. For example, to match any run of non-digit characters:

        [[:^digit:]]+

Non-Capturing Parentheses

As described in the preceding section "Creating Subpatterns", bare parentheses cluster and capture the subpatterns they contain. The portion of the matching pattern contained within the first pair of parentheses is available in the backreference \1, the second in \2, and so on.

Opening parentheses are counted from left to right to determine the numbers of the captured subpatterns. For example, if the following grep pattern:

        ((red|white) (king|queen))

is matched against the text "red king", the backreferences will be set as follows:

```
\1   "red king"
\2   "red"
\3   "king"
```

Sometimes, however, parentheses are needed only for clustering, not capturing. TextWrangler supports non-capturing parentheses, using the syntax:

```
(?:PATTERN)
```

That is, if an open parenthesis is followed by "?:", the subpattern matched by that pair of parentheses is not counted when computing the backreferences. For example, if the text "red king" is matched against the pattern:

```
(?:(red|white) (king|queen))
```

the backreferences will be set as follows:

```
\1   "red"
\2   "king"
```

Perl-Style Pattern Extensions

TextWrangler's grep engine supports several extended sequences, which provide grep patterns with super-powers from another universe. Their syntax is in the form:

```
(?KEY…)
```

in other words, an open parenthesis followed by a question mark, followed by a KEY for the particular grep extension, followed by the rest of the subpattern and a closing parenthesis. This syntax—specifically, an open parenthesis followed by a question mark—was not valid in older versions of TextWrangler, thus, none of these extensions will conflict with old patterns.

We have already seen one such extension in the previous section of this document—non-capturing parentheses: (?:…). The remainder are listed in the chart below, and discussed in detail afterward.

Extension	Meaning
(?:…)	Cluster-only parentheses, no capturing
(?#…)	Comment, discard all text between the parens
(?imsx-imsx)	Enable/disable pattern modifiers
(?imsx-imsx:…)	Cluster-only parens with modifiers
(?=…)	Positive lookahead assertion
(?!…)	Negative lookahead assertion
(?<=…)	Positive lookbehind assertion
(?<!…)	Negative lookbehind assertion
(?()…|…)	Match with if-then-else
(?()…)	Match with if-then

Extension	Meaning
(?>…)	Match non-backtracking subpattern ("once-only")
(?R)	Recursive pattern

Comments

The sequence `(?#` marks the start of a comment which continues up to the next closing parenthesis. Nested parentheses are not permitted. The characters that make up a comment play no part in the pattern matching at all.

> **Search for:** `foo(?# Hello, this is a comment)bar`
> **Will match:** <u>foobar</u>

Pattern Modifiers

The settings for case sensitivity, multi-line matching, whether the dot character can match returns, and "extended syntax" can be turned on and off within a pattern by including sequences of letters between "(?" and ")".

Modifier	Meaning	Default
i	case insensitive	according to Case Sensitive checkbox in Find & Replace dialog
m	allow ^ and $ to match at \r	on
s	allow . to match \r	off
x	ignore most white space and allow inline comments in grep patterns	off

i — By default, TextWrangler obeys the "Case Sensitive" checkbox in the Find & Replace dialog (or the corresponding property of the search options when using the scripting interface). The `(?i)` option overrides this setting.

m — By default, TextWrangler's grep engine will match the ^ and $ metacharacters after and before returns, respectively. If you turn this option off with `(?-m)`, ^ will only match at the beginning of the document, and $ will only match at the end of the document. (If that is what you want, however, you should consider using the new \A, \Z, and \z metacharacters instead of ^ and $.)

s — By default, the magic dot metacharacter . matches any character except return ("\r"). If you turn this option on with `(?s)`, however, dot will match *any* character. Thus, the pattern `(?s).+` will match an entire document.

x — When turned on, this option changes the meaning of most whitespace characters (notably, tabs and spaces) and #. Literal whitespace characters are ignored, and the # character starts a comment that extends until a literal return or the "\r" escape sequence is encountered. Ostensibly, this option intends to let you write more "readable" patterns.

Perl programmers should already be familiar with these options, as they correspond directly to the -imsx options for Perl's m// and s/// operators. Unadorned, these options turn their corresponding behavior on; when preceded by a hyphen (-), they turn the behavior off. Setting and unsetting options can occur in the same set of parentheses.

Example	Effect
`(?imsx)`	Turn all four options on
`(?-imsx)`	Turn all four options off
`(?i-msx)`	Turn "i" on, turn "m", "s", and "x" off

The scope of these option changes depends on where in the pattern the setting occurs. For settings that are outside any subpattern, the effect is the same as if the options were set or unset at the start of matching. The following patterns all behave in exactly the same way:

```
(?i)abc
a(?i)bc
ab(?i)c
abc(?i)
```

In other words, all four of the above patterns will match without regard to case. Such "top level" settings apply to the whole pattern (unless there are other changes inside subpatterns). If there is more than one setting of the same option at the top level, the right-most setting is used.

If an option change occurs inside a subpattern, the effect is different. An option change inside a subpattern affects *only* that part of the subpattern that follows it, so, if the "Case Sensitive" checkbox is turned on:

Search for: `(a(?i)b)c`
Will match: <u>abc</u> *or* <u>aBc</u>

and will not match anything else. (But if "Case Sensitive" is turned off, the "`(?i)`" in the above pattern is superfluous and has no effect.) By this means, options can be made to have different settings in different parts of the pattern. Any changes made in one alternative do carry on into subsequent branches within the same subpattern. For example:

Search for: `(a(?i)b|c)`

matches "ab", "aB", "c", and "C", even though when matching "C", the first branch is abandoned before the option setting.

These options can also be set using the clustering (non-capturing) parentheses syntax defined earlier, by inserting the option letters between the "?" and ":". The scope of options set in this manner is limited to the subpattern contained therein. Examples:

Search for: `(?i:saturday|sunday)`
Will match: <u>SATURDAY</u> *or* <u>Saturday</u> *or* <u>SUNday</u> (and so on)

Search for: `(?i:foo)(?-i:bar)`
Will match: <u>foobar</u> *or* <u>FOObar</u>
Will not match: FOOBAR *or* fooBAR

Positional Assertions

Positional assertions "anchor" a pattern, without actually matching any characters. Simple assertions have already been described: those which are invoked with the escape sequences \b, \B, \A, \Z, \z, ^ and $. For example, the pattern `\bfoo\b` will only match the string "foo" if it has word breaks on both sides, but the \b's do not themselves match any characters; the entire text matched by this pattern are the three characters "f", "o", and "o".

Lookahead and lookbehind assertions work in a similar manner, but allow you to test for arbitrary patterns to anchor next to. If you have ever said to yourself, "I would like to match 'foo', but only when it is next to 'bar'," lookaround assertions fill that need.

Positive lookahead assertions begin with "`(?=`", and negative lookahead assertions begin with "`(?!`". For example:

    \w+(?=;)

will match any word followed by a semicolon, but the semicolon is not included as part of the match.

    foo(?!bar)

matches any occurrence of "foo" that is *not* followed by "bar". Note that the apparently similar pattern:

    (?!foo)bar

does not find an occurrence of "bar" that is preceded by something other than "foo"; it finds any occurrence of "bar" whatsoever, because the assertion `(?!foo)` is always true when the next three characters are "bar". A lookbehind assertion is needed to achieve this effect.

Positive lookbehind assertions start with "`(?<=`", and negative lookbehind assertions start with "`(?<!`". For example:

    (?<!foo)bar

does find an occurrence of "bar" that is not preceded by "foo". The contents of a lookbehind assertion are restricted such that all the strings it matches must have a fixed length. However, if there are several alternatives, they do not all have to have the same fixed length. Thus

    (?<=Martin|Lewis)

is permitted, but

```
(?<!dogs?|cats?)
```

causes an error. Branches that match different length strings are permitted only at the top level of a lookbehind assertion. This is different compared with Perl 5.005, which requires all branches to match the same length of string. An assertion such as

```
(?<=ab(c|de))
```

is not permitted, because its single top-level branch can match two different lengths, but it is acceptable if rewritten to use two top-level branches:

```
(?<=abc|abde)
```

The implementation of lookbehind assertions is, for each alternative, to temporarily move the current position back by the fixed width and then try to match. If there are insufficient characters before the current position, the match is deemed to fail. (Lookbehinds in conjunction with non-backtracking [a.k.a. "once-only"] subpatterns can be particularly useful for matching at the ends of strings; an example is given in the section on once-only subpatterns below.)

Several assertions (of any sort) may occur in succession. For example,

```
(?<=\d{3})(?<!999)foo
```

matches "foo" preceded by three digits that are not "999". Notice that each of the assertions is applied independently at the same point in the subject string. First there is a check that the previous three characters are all digits, and then there is a check that the same three characters are not "999". This pattern does not match "foo" preceded by six characters, the first of which are digits and the last three of which are not "999". For example, it does not match "123abcfoo". A pattern to do that is:

```
(?<=\d{3}...)(?<!999)foo
```

This time the first assertion looks at the preceding six characters, checking that the first three are digits, and then the second assertion checks that the preceding three characters are not "999". Assertions can be nested in any combination. For example,

```
(?<=(?<!foo)bar)baz
```

matches an occurrence of "baz" that is preceded by "bar" which in turn is not preceded by "foo", while

```
(?<=\d{3}(?!999)...)foo
```

is another pattern which matches "foo" preceded by three digits and any three characters that are not "999".

Assertion subpatterns are not capturing subpatterns, and may not be repeated, because it makes no sense to assert the same thing several times. If any kind of assertion contains capturing subpatterns within it, these are counted for the purposes of numbering the capturing subpatterns in the whole pattern. However, substring capturing is carried out only for positive assertions, because it does not make sense for negative assertions.

Conditional Subpatterns

Conditional subpatterns allow you to apply "if-then" or "if-then-else" logic to pattern matching. The "if" portion can either be an integer between 1 and 99, or an assertion. The two forms of syntax are:

if-then: `(?(condition)yes-pattern)`
if-then-else: `(?(condition)yes-pattern|no-pattern)`

If the condition evaluates as true, the "yes-pattern" portion attempts to match. Otherwise, the "no-pattern" portion does (if there is a "no-pattern").

If the "condition" text between the parentheses is an integer, it corresponds to the backreferenced subpattern with the same number. (Do not precede the number with a backslash.) If the corresponding backreference has previously matched in the pattern, the condition is satisfied. Here's an example of how this can be used. Let's say we want to match the words "red" or "blue", and refer to whichever word is matched in the replacement pattern. That's easy:

    (red|blue)

To make it harder, let's say that if (and only if) we match "blue", we want to optionally match a space and the word "car" if they follow directly afterward. In other words, we want to match "red", "blue", or if possible, "blue car", but we do not want to match "red car". We cannot use the pattern:

    (red|blue)(car)?

because that will match "red car". Nor can we use:

    (red|blue car|blue)

because in our replacement pattern, we want the backreference to only contain "red" or "blue", without the " car". Using a conditional subpattern, however, we can search for:

    ((blue)|(red))(?(2) car)?

Here's how this pattern works. First, we start with "`((blue)|(red))`". When this subpattern matches "blue", \1 and \2 are set to "blue", and \3 is empty. When it matches "red", \1 and \3 are set to "red", and \2 is empty.

Next comes the conditional subpattern "`(?(2) car)?`". The conditional test is on "2", the second backreferenced subpattern: if \2 is set, which in our case means it has matched the word "blue", then it will try to match " car". If \2 is not set, however, the entire conditional subpattern is skipped. The question mark at the end of the pattern makes this conditional match optional, even if \2 is set to "blue".

Here's an example that uses an assertion for the condition, and the if-then-else form. Let's say we want to match a run of digits of any length, followed by either " is odd" or " is even", depending on whether the matched digits end with an odd or even digit.

    \d+(?(?<=[13579]) is odd| is even)

This pattern starts with "\d+" to match the digits. Next comes a conditional subpattern, with a positive lookbehind assertion as the condition to be satisfied. The lookbehind assertion is true only if the last character matched by \d+ was also in the character class [13579]. If that is true, we next try to match " is odd"; if it is not, we try to match " is even". Thus, this pattern will match "123 is odd", "8 is even", and so on, but will not match "9 is even" or "144 is odd".

Once-Only Subpatterns

With both maximizing (greedy) and minimizing (non-greedy) repetition, failure of what follows normally causes the repeated item to be reevaluated to see if a different number of repeats allows the rest of the pattern to match. Sometimes it is useful to prevent this, either to change the nature of the match, or to cause it to fail earlier than it otherwise might, when the author of the pattern knows there is no point in carrying on.

Consider, for example, the pattern "\d+foo" when matching against the text "123456bar".

After matching all 6 digits and then failing to match "foo", the normal action of the grep engine is to try again with only 5 digits matching the \d+ item, and then with 4, and so on, before ultimately failing. Once-only subpatterns provide the means for specifying that once a portion of the pattern has matched, it is not to be reevaluated in this way, so the matcher would give up immediately on failing to match "foo" the first time. The notation is another kind of special parenthesis, starting with "(?>", as in this example:

```
(?>\d+)bar
```

This kind of parentheses "locks up" the part of the pattern it contains once it has matched, and a failure further into the pattern is prevented from backtracking into it. Backtracking past it to previous items, however, works as normal.

In most situations, such as in the example above, the time saved by using once-only subpatterns is insignificant—a few small fractions of a second, at most. With some complicated grep patterns or with humongous lines of text, however, you can save tremendous amounts of time using once-only subpatterns.

Once-only subpatterns are not capturing subpatterns. Simple cases such as the above example can be thought of as a maximizing repeat that must swallow everything it can. So, while both \d+ and \d+? are prepared to adjust the number of digits they match in order to make the rest of the pattern match, (?>\d+) can only match an entire sequence of digits.

Once-only subpatterns can be used in conjunction with lookbehind assertions to specify efficient matching at the end of a line of text. Consider a simple pattern such as:

```
abcd$
```

when applied to a long line of text which does not match (in other words, a long line of text that does not end with "abcd"). Because matching proceeds from left to right, the grep engine will look for each "a" in the subject and then see if what follows matches the rest of the pattern. If the pattern is specified as:

```
^.*abcd$
```

the initial `.*` matches the entire line at first, but when this fails (because there is no following "a"), it backtracks to match all but the last character, then all but the last two characters, and so on. Once again the search for "a" covers the entire string, from right to left, so we are no better off. However, if the pattern is written as:

```
^(?>.*)(?<=abcd)
```

there can be no backtracking for the `.*` item; it can match only the entire line. The subsequent lookbehind assertion does a single test on the last four characters. If it fails, the whole match fails immediately. For long strings, this approach makes a significant difference to the processing time.

When a pattern contains an unlimited repeat inside a subpattern that can itself be repeated an unlimited number of times, the use of a once-only subpattern is the only way to avoid some failing matches taking a very long time (literally millions or even billions of years, in some cases!). The pattern:

```
(\D+|<\d+>)*[!?]
```

matches an unlimited number of substrings that either consist of non-digits, or digits enclosed in <>, followed by either ! or ?. When it matches, it runs quickly. However, if it is attempts to match this line of text:

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

it takes a long time before reporting failure. So long, in fact, that it will effectively "freeze" TextWrangler. This is not really a crash, per se, but left to run on its own, it might take years before it finally fails. (We are not sure, frankly, because much like determining how many licks it takes to get to the center of a Tootsie Pop, we do not feel like waiting long enough to find out.)

The reason this takes so long to fail is because the string can be divided between the two repeats in a large number of ways, and all have to be tried before the grep engine knows for certain that the pattern will not match. (The example used `[!?]` rather than a single character at the end, because both PCRE and Perl have an optimization that allows for fast failure when a single character is used. They remember the last single character that is required for a match, and fail early if it is not present in the string.) If the pattern is changed to

```
((?>\D+)|<\d+>)*[!?]
```

sequences of non-digits cannot be broken, and failure happens quickly.

Recursive Patterns

Consider the problem of matching a string in parentheses, allowing for unlimited nested, balanced parentheses. Without the use of recursion, the best that can be done is to use a pattern that matches up to some fixed depth of nesting. It is not possible to handle an arbitrary nesting depth. Perl 5.6 has provided an experimental facility that allows regular expressions to recurse (among other things). It does this by interpolating Perl code in the expression at run time, and the code can refer to the expression itself. Obviously, TextWrangler's grep engine cannot support the interpolation of Perl code. Instead, the special item `(?R)` is provided for the specific case of recursion. The following recursive pattern solves the parentheses problem:

```
\(((?>[^()]+)|(?R))*\)
```

First it matches an opening parenthesis. Then it matches any number of substrings which can either be a sequence of non-parentheses, or a recursive match of the pattern itself (that is, a correctly parenthesized substring). Finally there is a closing parenthesis.

This particular example pattern contains nested unlimited repeats, and so the use of a once-only subpattern for matching strings of non-parentheses is important when applying the pattern to strings that do not match. For example, when it tries to match against this line of text:

```
(aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa()
```

it yields "no match" quickly. However, if a once-only subpattern is not used, the match runs for a very long time indeed because there are so many different ways the + and * repeats can carve up the subject, and all have to be tested before failure can be reported.

Browsers

Browsers are special kinds of windows that let you see a lot of information about files at once. Browsers typically have two panes: one pane lets you select a file, the other displays detailed information about the file (often its contents). If you have used the Batch Find option in a multi-file search, you have already seen an example of a TextWrangler browser.

In this chapter

Browser Overview

All TextWrangler browsers share the same basic structure and behavior. All browsers have a status bar, a file list, and a text pane.

List Pane

The top pane of a browser lists the items available in the browser. This pane shows different information for different kinds of browsers:

Browser	File List pane contains
Disk browser	Files and folders that TextWrangler can open
Search results	File and line number of each match

You can open both files and folders from the list pane. When you double-click a folder name, TextWrangler replaces the file list pane with the contents of the folder. When you double-click a file name, TextWrangler opens the file in an editing window. If the file list pane also included a line number, TextWrangler scrolls to that line.

Controls above the list may allow you to determine what kinds of items are displayed in the list. For example, in disk browsers, there is a pop-up menu that lets you choose to display text files, all files, or other types of files, and another that lets you return the browser to a parent directory of the current folder. In search browsers, checkboxes allow you to hide or show all search errors, warnings, or results.

For results browsers, TextWrangler can either show a hierarchy (where all the messages associated with a particular file are grouped under that file, using disclosure triangles similar to those in the Finder's list views to reveal or hide the error list), or a flat listing showing each individual message on a separate line. You can choose which of these display methods to use by default in the Browser Display preferences. To remove items from the display list, select them and press the Delete key, or choose Clear from the Edit menu.

Status Bar

The browser status bar is like the status bar in editing windows. Some browsers have additional buttons and controls in the status area as well.

These standard items—the pencil icon; the Function, Text Options, Mark, Path pop-up menus; and the Projector and Info buttons—should already be familiar to you, since they appear on TextWrangler document windows by default. See "Window Anatomy" in Chapter 4 for an explanation of these standard TextWrangler functions.

Text View Pane

When you click an item in the list pane, TextWrangler displays its contents in the text view pane. If you click a folder, TextWrangler lists the names of the files in the folder. If you click a file name, TextWrangler displays the contents of the file.

When the current focus is in the bottom pane, the space bar acts like the "Page Down" key, and Shift-space acts as "Page Up."

Splitter

You can change the size of the file list pane or the text view pane by dragging the double line that separates them. Double-clicking on the splitter bar will collapse the text view pane completely, and double-clicking on it again (in the bottom of the browser window) will restore the text pane to its previous proportions.

Disk Browsers

Use a disk browser to explore the contents of a disk or a folder without opening each file one at a time.

To open a disk browser, pull down the File menu and choose Disk Browser from the New submenu. TextWrangler opens a new disk browser that starts at your home folder, and you can then navigate to any desired location.
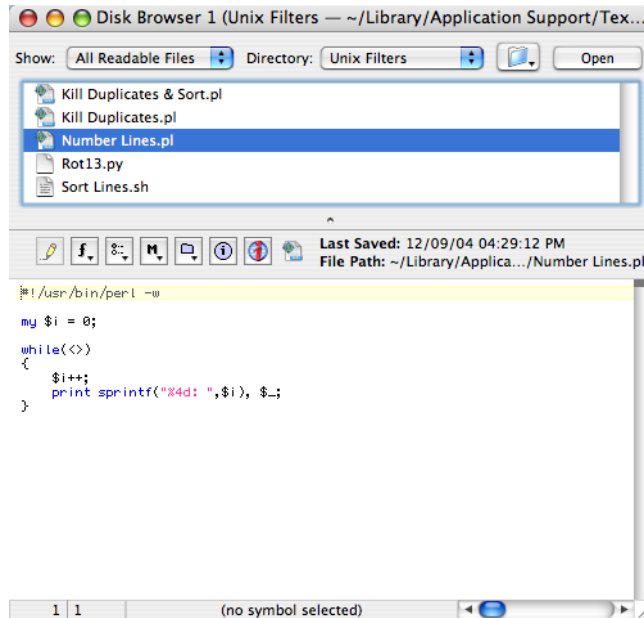
Note You can also open a disk browser starting at any particular folder, by dragging that folder onto TextWrangler's icon in the Finder.

Using Disk Browsers

The controls in the disk browser let you open files and folders, limit the kinds of files to show in the list pane, and navigate through your disks and folders.

Show Pop-Up Menu

The pop-up menu on the upper-left section of the list pane lets you specify the kinds of files you want TextWrangler to list in the browser. You can select "All Readable Files," meaning all types of files TextWrangler recognizes that it can open, or "All Files" which will display every file present in a folder regardless of type or kind. You can also select from one of the more specific types listed below: text files, PICT files, QuickTime movies, and QuickTime images.

Directory Pop-Up Menu

This pop-up menu always shows the currently selected folder. By default, when you open a disk browser, it will display the contents of your home folder. If you have opened any volume or folder within a volume, the current folder will be shown here instead.

You can use this menu to "back out" of any folder you are currently in to a higher-level folder (as you can in the Finder).

Folder Pop-Up Menu

This pop-up menu allows you to select a recently-used folder to view, or you can select Other to choose an arbitrary folder.

Open Button

Clicking the Open button in the status bar opens the selected item, or you can double-click on the item. If you open a folder, TextWrangler displays the contents of that folder in the file list pane. If you click one or more files, TextWrangler opens those files into editing windows.

Alternatively, you can press Command-Down-Arrow to open the selection if

- you have selected one item and it is a file, or,
- you have selected multiple items, regardless of type

Tip Hold down the Option key as you Open or double-click on a file name to open the file with the application that created it.

Using the List Pane in Disk Browsers

The list pane of a disk browser displays disks, files, and folders. When you are at the desktop level, the list shows all mounted volumes and the files and folders on the Desktop.

When you click a folder or disk, TextWrangler displays the names of all the files it can open in the text pane. (The names listed in the text pane cannot be double-clicked to open them. You can however select a name and use the Open Selection command to open that file.) When you click a file name, TextWrangler displays the contents of the file in the text pane.

To open a folder or disk and display its contents in the file list pane, do one of the following:

- double-click it
- select it and click Open in the status bar
- select it and press Command-Down Arrow

To go up one level to the enclosing folder or disk, do one of the following:

- choose the enclosing folder from the directory pop-up menu
- press Command-Up Arrow

Using the Text Pane in Disk Browsers

When you select a folder or disk in the file list pane, TextWrangler displays the names of all the files and folders contained in that folder or disk in the text pane. When you click on a file name, TextWrangler displays the contents of the file in the text pane if the file is of a type that TextWrangler recognizes ("TEXT", "utxt", or "UTF8").

You can search the contents of the text pane with the Find command or with the Quick Search window, and you can copy text from the text pane. You cannot edit a file's contents in the text pane. To edit a file, use the Open button in the status bar to open it in an editing window.

Search Results Browsers

If you selected the Batch Find option when performing a multi-file search, TextWrangler displays every occurrence of the search string in the searched files in a search results browser.
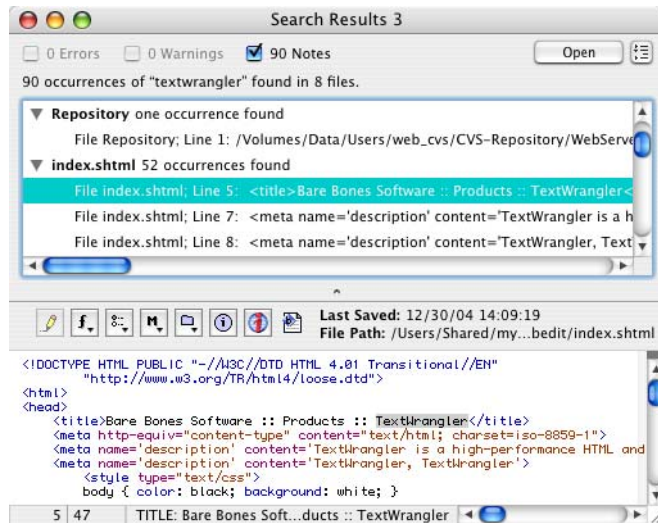
The items at the top of the window tell you how many matches TextWrangler found in the set of files you specified, as well as whether any error conditions or warnings were generated during the search. The list pane lists each line that contains the matched text. (Depending on how you have configured TextWrangler, the list may be arranged hierarchically, with every match attached to the file that contains it in a Finder-like list view, or they may be listed one after the other in a simple flat list.) Every match is identified by file and line number. To choose whether to display the search errors, warnings, and results, use the checkboxes at the top of the browser.

The Open button opens the selected item. To reveal the selected items in the Finder, hold down the Option key while clicking the Open button.

To see the contents of a file, click a line in the list of matched searches. The part of the file that contains the search string appears in the lower pane. You cannot edit text in a search results browser. To edit the file, double-click the line that contains the match you want to edit.

After you have opened the file, you can use the Find Again, Replace, Replace All, and Replace & Find Again commands in the Search menu to continue searching it, as if you had chosen a File by File search. See Chapter 7 for more information on searching.

10

Preferences

You can use the Preferences command to customize much of TextWrangler's behavior. You can decide which windows are open when you launch TextWrangler, set the default options for windows, set the default options for searches, and so on. This chapter describes TextWrangler's extensive preference options.

In this chapter

The Preferences Command

The Preferences window provides control over many aspects of TextWrangler's behavior. You can decide which windows should open when you launch TextWrangler, set the default display options for windows, set default options for editing behavior and searches, and so on.

TextWrangler 2 employs standard OS services to store its preference settings, bringing improved durability and performance. TextWrangler's use of these services also makes it possible for you to modify preference settings directly using appropriate "defaults write" commands (see page 192). However, the Preferences window remains the standard interface for making and changing preference settings.

IMPORTANT   You should **not** arbitrarily remove your TextWrangler preferences file, as doing so will require you to re-select any customized preference options you may have chosen.

Note   If you chose to modify your preferences by means of "defaults write" commands other than those documented here, without explicit advice from Bare Bones Software technical support, you take responsibility for any adverse effects.

To open the Preferences window, choose the Preferences command from the TextWrangler menu.
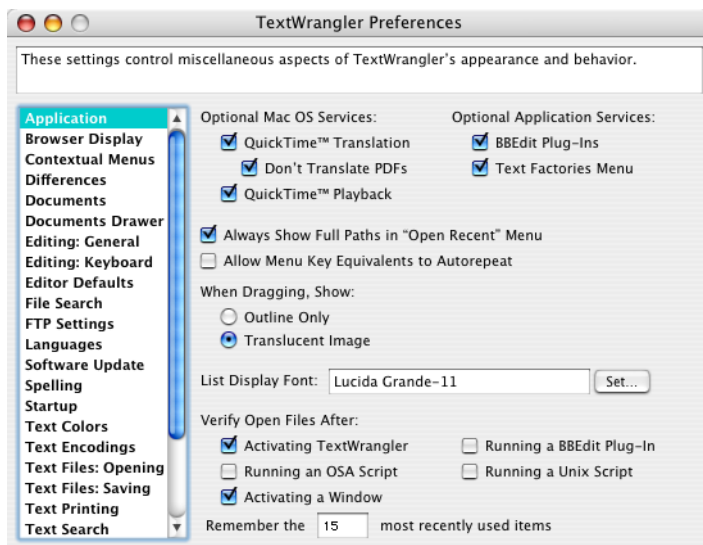
The small text area at the top left of the Preferences window gives you a brief description of the contents of the currently displayed preferences panel.

The list along the left side of the window lets you select the preferences panel to use.

TextWrangler's Preferences window is non-modal: you can leave it open and change preference settings while you work, or close it at any time by clicking its close button or by choosing Close Window from the File menu.

Any changes you make to preference settings will be stored immediately, though not every setting will take effect immediately. Some options are only checked or applied at application launch, as mentioned in their descriptions.

The following sections describe each of the preferences panels in detail.

Application Preferences

The Application preferences control which application and system services TextWrangler uses, when open files are verified, and various other global settings.

Optional Mac OS Services

QuickTime™ Translation

If you mark this checkbox and have a recent version of QuickTime installed, TextWrangler will ask QuickTime if it recognizes any files that TextWrangler does not. If QuickTime recognizes the file as one it can display, TextWrangler will open it for viewing; otherwise, the file will be opened as "raw" text. This option is on by default.

Don't Translate PDFs

If the supplemental option Don't Translate PDFs is checked, TextWrangler will not attempt to apply QuickTime translation to PDF files. This option is on by default.

QuickTime™ Playback

This checkbox controls whether TextWrangler opens QuickTime movies for playback in a movie window or just opens them as text data in an editing window. The default is to recognize QuickTime movies and display them with a player. Changes to this preference take effect immediately.

Optional Application Services

BBEdit Plug-ins

When this option is selected, TextWrangler makes the Tools menu available in the menu bar so that you can use plug-ins (code modules that add functionality to TextWrangler). Third-party plug-ins are kept in TextWrangler's application support folders. See Chapter 2 to learn more about the organization of TextWrangler's application folders, or Chapter 13 for information about plug-ins.

Script Menu

This option controls whether TextWrangler will display the Script menu in its menu bar. If you prefer to always use the global Script menu, turn this option off. TextWrangler's built-in Script menu does, however, offer a number of advantages over the system-wide Script menu, not the least of which is that it runs scripts in the application's context. Changes to this option will take effect the next time you launch TextWrangler.

Text Factories

This option controls whether TextWrangler displays the Text Factories menu in the menu bar. Changes to this preference will take effect the next time you start the application.

Always Show Full Paths in "Open Recent" Menu

Mark this checkbox to always show the complete pathname of recently opened files, rather than just their names, in TextWrangler's Open Recent submenu. If a recent file is missing, only its name will appear, regardless of this setting. Also, if two or more files have identical filenames, their complete pathnames will always be shown to prevent ambiguity, regardless of this setting.

Allow Menu Key Equivalents to Autorepeat

Mark this checkbox to enable autorepeat when typing key equivalents. This option is off by default. (According to the Macintosh Human Interface Guidelines, menu commands should not autorepeat. If you chose to enable this option, you do so forewarned.)

When Dragging, Show

These radio buttons control the way selected text is displayed while tracking mouse movements during a drag-and-drop operation: choose Outline Only to have the selection displayed as an outline, or Translucent Image to have the selection displayed as visible, translucent text.

List Display Font

This option controls the font and size used to display text in browser list panes, including disk browser, search results browsers, etc. To change this option, click Set to bring up the standard Font panel, and choose the desired font and size. The default setting is 11 point Lucida Grande.

Verify Open Files After

If you frequently make changes to files with another program while they are open in TextWrangler, or you often edit shared files that might be changed by someone else, you may want to mark one or more of these checkboxes. If the option for Activating TextWrangler is checked, for instance, TextWrangler will check the modification date of each open document every time you switch to TextWrangler, alerting you if any changes have been made to the file while TextWrangler was not the frontmost application. Similar functions are available to perform this check after running an AppleScript, a TextWrangler plug-in, or a Unix script, or activating a text window.

The effects of the Revert command (from the File menu), and of a file Reload (which occurs when a document is reloaded by a verify action) are now both undoable.

Remember the N most recently used items

This text field lets you choose how many files appear on the Open Recent sub-menu of the File menu, and how many folders appear on the folder search pop-up menu in the Find Differences folder lists.

Browser Display Preferences

The Browser Display preferences control the initial appearance of TextWrangler's built-in browsers for search results, compile errors, disk contents, and so forth.

Results Lists

For search results browsers, you can have TextWrangler display either a hierarchical listing (where all the matches associated with a particular file are grouped under that file, using disclosure triangles similar to those in the Finder's list views to reveal or hide the match list) or a flat listing showing each individual match encountered on a separate line. In the former case, you can also tell TextWrangler to expand all the file nodes initially, by checking Expand Listing. This preference also governs the default display of hierarchical or flat error listings in error browsers.

Disk Browsers

When the Show File Icons option is active, disk browser windows will display small icons for every item (files, folders, and disk volumes) in the file list. When this option is off, disk browsers will list only the names of files and folders.

Contextual Menu Preferences

The Contextual Menu preferences control which commands TextWrangler will present on its contextual menu. You can show or remove commands in any category by enabling or disabling the appropriate option.

Differences Preferences

The Differences preferences control the way the Find Differences command places its windows.

Arrange Windows on

The Arrange Windows radio buttons let you choose which screen TextWrangler uses to display Differences windows. You can choose the main screen, the largest screen, or the smallest screen.

When you mark the Use Two Screens checkbox, TextWrangler displays the Differences window on the second screen. This option gives you the largest view of the items you are comparing.

Keep Windows Arranged provides control over whether TextWrangler rearranges the windows being compared when you click a difference item.

Differences Window Placement

The Beneath/Above Compared Files radio buttons let you choose whether the differences window should be placed below or above the windows of the documents you are comparing.

Arrange Windows

You can specify whether differences windows are arranged left to right or top to bottom using the Arrange Windows radio buttons.

Multi-File Differences

When comparing multiple files, the Show File Icons checkbox determines whether the files' Finder icons are shown in the Differences window.

Options

The Keep Windows Arranged option controls whether TextWrangler should attempt to automatically resize and reposition windows during a Find Differences operation.

Documents Preferences

The Documents preferences control how TextWrangler handles opening text documents and creating text windows.

New & Opened Documents

You can specify whether newly created or opened documents always should be opened into the frontmost text window, or whether each document should be opened into its own text window.

Documents Opened from Other Applications

You can specify whether documents opened from other applications should be opened into the frontmost text window, or whether each document should be opened into its own text window.

Examples of opening documents from another application include using the "Edit in TextWrangler" command from an FTP client such as Interarchy, and double-clicking files in the Finder.

Warn Before Closing a Window Containing Multiple Documents

Choose this option to have TextWrangler warn you when you attempt to close a window with more than one document in it. This warning will not occur if:

- TextWrangler is quitting

- Any of the documents in the window contain unsaved changes, since TextWrangler already asks for confirmation to save changes.

Documents Drawer Preferences

Open the Documents Drawer

You can specify when a new text window will display the Documents Drawer: always, only if the window contains two or more documents, or never.

Open the Documents Drawer on the

You can specify whether the Documents Drawer should be displayed on the left-hand or right-hand side of text windows by default.

Next Document and Previous Document Navigate in

You can specify whether the Next Document and Previous Document commands should select documents according to their display order, or in order of most recent use.

Allow Documents Drawer to Acquire Keyboard Focus

Choose this option to prevent the Documents Drawer from also gaining keyboard focus when you click in it. When this option is on, you will not be able to use typeahead or the arrow keys to move among open documents in the drawer.

Editing: General Preferences

The Editing: General preferences control the behavior of various general editing behaviors.

Allow Single-Click Line Selection

If the checkbox is turned on, clicking in the left margin of an editing window selects an entire line. (If you have line numbers displayed, via the Show Line Numbers option in the Text Status Display preferences panel, you can click in the line number as well.) The pointer changes to a right arrow when it is in the left margin. Click and drag to select multiple lines. Double-click to select an entire paragraph; double-click and drag to select a range of paragraphs.

If this option is off, clicking in the left margin moves the insertion point to the beginning of the clicked line.

Double-Click to Balance

When this preference is turned on, you can double-click any opening or closing parenthesis, brace, or bracket— ( ) { } [ ] —to select the entire range of text enclosed by a balanced pair.

Confirm Non-Undoable Editing Actions

To have TextWrangler stop warning you when you are about to perform an action that cannot be undone, turn this checkbox off.

Include Delimiter Characters when Balancing

This option controls whether TextWrangler selects delimiter characters (parentheses, braces, brackets, etc.) when you use the Balance command (either by choosing it from the Text menu or by double-clicking on a delimiter). This option is on by default.

Use "Hard" Line Numbering in Soft-Wrapped Views

If this option is turned on, the line number bar, cursor position display, and Go To Line commands in editing views will use line and character position numbers that correspond to the "hard" line breaks actually present in the document, rather than the soft-wrapped line breaks.

Additionally, when this option is on, line selection commands and gestures, including the Select Line command, triple-clicking, and click selection in the left margin, will treat only "hard" line breaks as line boundaries.

Extra Space in Text Views

To have TextWrangler leave extra empty space when you scroll to the end of a text view, choose Half Window or Full Window here.

Turn Off Text Smoothing

This option allows you to choose the font sizes for both fixed width and proportional fonts at and below which TextWrangler will not employ text smoothing.

Editing: Keyboard Preferences

The Editing: Keyboard preferences control TextWrangler's response to the use of various special keys.

Enable Shift-Delete for Forward Delete

When this option is selected, holding down the Shift key with the Delete key makes the Delete key work the same way as the Forward Delete key on extended keyboards. This feature is particularly useful on PowerBooks.

Use Numeric Keypad for Cursor Movement

To use the numeric keypad to move the insertion point, select this option.

start of line	up	scroll up
7	**8**	**9**

left **4**	show selection **5**	right **6**
end of line **1**	down **2**	scroll down **3**

You can use the Shift key with the keys on the numeric keypad to extend a selection. You can use the Command and Option keys with the 2, 4, 6, and 8 keys as you would the arrow keys.

To toggle the behavior of the keypad between moving the cursor and entering numbers, hold down the Option key and press the Clear key in the upper-left corner of the keypad. (This key is also labeled Num Lock on some keyboards.)

When Auto-Indenting

This option controls whether TextWrangler should remove any existing leading whitespace from lines which it applies auto-indentation to.

"Home" and "End" Keys

Choose "Scroll to Beginning and End of Document" to have the Home and End keys perform these respective actions. This is the default setting, which reflects the standard key motion behavior in Macintosh applications.

Choose "Move Cursor to Beginning and End of Current Line" to have the Home and End keys perform these respective actions instead. This option may be useful for those accustomed to Windows editing key behavior.

Exchange Command and Option Key Behavior

These two checkboxes let you swap the meaning of the Option and Command keys when used with cursor navigation keys to move through a window's contents. You can set this separately for horizontal and vertical cursor movement. For details on using cursor navigation keys, see Chapter 4 and Appendix B.

Use Emacs Key Bindings

If turned on, this option allows you to use the basic Emacs navigation keystrokes to move around in editing views. It is not a full Emacs emulation mode; rather, it is more of a comfort blanket for individuals with Emacs key bindings hard-wired into their muscle memory. See Appendix B, "Editing Shortcuts," for a list of the Emacs commands TextWrangler supports.

If you turn on the Display Status Window option, a small palette will appear when you type an Emacs shortcut, indicating which command you have applied.

Option-¥ on Japanese Keyboards

This option controls whether typing Option-yen on a Japanese keyboard generates a yen symbol "¥" or a backslash "\".

Editor Defaults Preferences

The Editor Defaults preferences control the behavior of newly created document windows and documents without saved state information. Many of the preferences in this panel are the same as those in the Text Options dialog and in the Text Options pop-up menu in the status bar. The difference is that the text options control the behavior of the active window, while the Editor Defaults preferences control the behavior of new windows.

Auto-Indent

When this option is selected, pressing the Return key in new windows automatically inserts spaces or tabs to indent the new line to the same level as the previous line.

Tip To temporarily invert the sense of the Auto Indent option while typing, hold down the Option key as you press the Return key.

Balance While Typing

When this option is selected, TextWrangler flashes the matching open parenthesis, brace, bracket, or curly quote when you type a closing one. This option is useful when you are editing source files, to ensure that all delimiters are balanced.

Smart Quotes

When this option is on, TextWrangler automatically substitutes curly (or typographer's) quotes (" " ' ') for straight quotes (" ').

Tip To type a straight quote when this option is selected (or to type a curly quote when the option is deselected), hold down the Control key as you type a single or double quote.

Note You should avoid using Smart Quotes when creating or editing HTML documents and email message content.

Auto-Expand Tabs

When this option is selected, TextWrangler inserts an appropriate number of spaces instead of a tab character every time you press the Tab key.

Show Invisibles

This option shows or hides non-printing characters in the window. Select this option when you want to see line breaks, tabs, and gremlins (invisible characters). TextWrangler uses these symbols to represent non-printing characters:

Symbol	Meaning
∆	tab
◊	space
•	non-breaking space
¬	line break
¶	page break
¿	other non-printing characters

Show Spaces

If this setting is selected (and Show Invisibles is active), TextWrangler will display placeholder characters for spaces. Deselect this option to suppress the display of spaces, which will reduce clutter when you are displaying invisible characters.

Note   Non-breaking spaces (typed by pressing Option-space) will not be displayed with a placeholder.

Syntax Coloring

When this option is selected and the editing window contains a document in one of the languages that TextWrangler knows how to parse, TextWrangler displays keywords and other language elements in color.

The languages that TextWrangler knows about are those listed in the Languages panel of the Preferences window. Remember that the document must be saved to a file and that the file must end with a suffix (extension) that maps to a language that TextWrangler can parse.

You can select the colors that TextWrangler uses for syntax coloring in the Text Colors panel of the Preferences window.

Soft Wrap Text

When this option is selected, TextWrangler soft-wraps the text in the file to the right margin that you choose: the Page Guide, the window width, or a specific number of characters, as selected by the options below the checkbox.

Default Font

This option controls the standard font and font size, and the number of spaces per tab, which TextWrangler uses to display the contents of text windows. To change this option, click Set to bring up the standard Font panel, and choose the desired font and size, or tab width. The default setting is 9 point Monaco, with 4 spaces per tab.

File Search Preferences

The File Search preferences control the way TextWrangler searches for files when you use the Open File by Name or Open Selection command from the File menu.

Find All Matching Files

When this option is selected, TextWrangler looks for all the files that match the entered or selected text. Otherwise TextWrangler stops looking as soon as it finds the first file that matches the selected name.

Skip (…) Folders

When this option is selected, TextWrangler does not search folders whose names are enclosed in parentheses.

Search Folders

The Search Folders box displays a list of folders which TextWrangler will search in response to an Open File by Name or Open Selection command, or a Control-Tab to locate the corresponding source or header file. The contents of each listed folder will be searched recursively, i.e. the contents of any subfolders will also be searched.

To add folders to the list, do any of the following:

- Click the Choose button and select the folder from the directory selection dialog.
- Drag the icon of the folder to the path box.

To change the target folder for an existing entry, select it from the list, click "Change", and choose a new folder using the directory selection dialog.

To remove folders from the list, select them, and click "Remove".

Unix Search Paths

The Unix Search Paths box displays a list of folders which TextWrangler will search in response to an Open File by Name or Open Selection command, or a Control-Tab to locate the corresponding source or header file. The contents of each listed folder will only be searched directly, i.e. without recursion.

Unix search paths are designed to make it easier to work with Unix source code, which uses include statements of the form

```
#include <xxx/yyy.h>
```

As a more concrete example: the canonical Unix include directory is "/usr/include". It contains its own subdirectories, but since Unix command line compilers do not usually do recursive searches, you need to qualify the include file's name if you want to include a file out of one of the subdirectories:

```
#include <sys/ioctl.h>
```

With the Unix Search Paths settings, you can add "/usr/include" to the list (actually, this is one of the factory defaults). When you select "sys/ioctl.h" and choose Open Selection, TextWrangler attempts to construct a file path using each of the directories shown in the Unix Search Paths list. If one resolves to a file, TextWrangler will open the resulting file. Thus, the partially qualified selection "sys/ioctl.h" resolves to

```
/usr/include/sys/ioctl.h
```

and the file opens.

FTP Settings Preferences

The FTP Settings preferences let you change the default settings of some options in the Open from FTP/SFTP Server and Save to FTP Server dialogs.

Remember Bookmark Passwords

Mark this checkbox to have TextWrangler store the passwords for FTP sites.

Include Passwords in Proxy URL Drags

Dragging the window proxy icon from an editing window corresponding to a file opened from an FTP server will drag that file's URL, rather than a representation of the local temporary file. To control whether the dragged URL includes the FTP account password, set the Include Passwords in Proxy URL Drags checkbox. This setting is turned off by default so that you do not accidentally create clippings containing passwords, as this may be a security risk.

List FTP Files on the "Open Recent" Menu

Mark this checkbox to show files opened from FTP sites on the Open Recent submenu of the File menu. (If this box is not checked, the Open Recent submenu lists only local files.)

Passive FTP

Mark this checkbox to make passive FTP mode (where the server determines the port number for the FTP connection) the default. Use Passive FTP whenever possible.

Show Document Icons

Mark this checkbox to display icons for the files in the Open from FTP/SFTP Server and Save to FTP Server dialogs. Since FTP servers do not provide Macintosh type and creator information, TextWrangler determines the displayed icon based on the file's name suffix (.html, .sit, and so on).

Note   Mac OS X does not currently provide any direct interface for configuring these suffix-to-type mappings. However, you can use Internet Explorer's File Helper preferences, or a third-party System Preferences pane such as RCDefaultApp for this purpose.

Show Files Starting with "."

The Unix convention for creating invisible or hidden files is to begin their names with a period. Often, configuration files and scripts (such as .newsrc or .login) begin with periods so that they do not clutter most directory listings. This setting is off by default, so that you will not see such files in FTP listings. To display them, mark this checkbox.

FTP Bookmarks

This list displays the bookmarks you have defined for FTP servers you use frequently with TextWrangler. Click Add to create a new bookmark, click Change to edit the selected bookmark, or Remove to delete it.

When adding or editing a bookmark, set the SFTP option in the Edit Bookmark dialog to have TextWrangler connect to this server via SFTP; if the option is unchecked, TextWrangler will use FTP instead.

Note   If you have the Preferences window open, you will not be able to add bookmarks in the Open From/Save To FTP Server dialogs. To work around this, close the Preferences window before using the FTP dialogs to add new bookmarks.

Languages Preferences

The Languages preferences allow you to configure how TextWrangler maps file names to language types (e.g. ".html" to HTML), and to configure certain parameters for each supported language.

Installed Languages

The list of installed languages includes both languages intrinsically supported by TextWrangler, and those provided via additional language modules.

Choose a language and click "Make Default" to use that language as the default when creating new text documents, and when opening text documents for which the language cannot otherwise be guessed (by mapping the file's suffix or examining its content). The default setting is "(none)".

To configure the comment-start and comment-end strings used by the Un/Comment command on the Text menu for a particular language, choose it from the list, click "Options", and edit its comment strings as desired.

Suffix Mappings

By default, TextWrangler offers a set of file suffix-to-language mappings which covers the common usages for each supported language.

To add a new suffix mapping:

1 **Click Add.**

   The Add Suffix dialog appears.

2 **Enter the suffix, choose the associated language from the pop-up menu, and click a radio button to tell TextWrangler whether this type of file is a source file, an include file, or neither.**

3 Click Add to save the new mapping.

Note You can use wildcards in the suffix to indicate single characters (?), any number of characters (*), or a single digit (#). For example, "page.#html" could map to a different language from ".html".

To change an existing suffix mapping:

1 Select an item from the list.

2 Click Change.

The Change Suffix dialog appears.

3 Fill in the Change Suffix dialog with the appropriate suffix, choose a language from the pop-up menu, and select a radio button to indicate whether this type of file is a source file, an include file, or neither.

To delete a suffix mapping:

1 Select an item from the list.

2 Click Remove.

To reset all suffix mappings to default settings:

1 Click Reset All.

Software Update Preferences

The Software Update preferences control the integrated version-checking capability built into TextWrangler.

The Check Automatically checkbox controls whether TextWrangler automatically looks to see if there is a new version available from Bare Bones Software. Regardless of the setting of the checkbox, you can always check manually by clicking the Check Now button.

The version checking mechanism used by TextWrangler protects your privacy. It works by requesting information about the currently available version from Bare Bones Software's web server. The server will log the date, time and originating address of the request, and which versions of the OS and TextWrangler you are using. This information is used to guide the future development of TextWrangler; it is not personalized and will not be disclosed.

Spelling Preferences

The Spelling preferences control whether TextWrangler uses the Mac OS X system spelling checker, or an external spelling checker.

Built-In

This option specifies that TextWrangler should use the Mac OS X system spelling checker when you choose the Check Spelling command from the Text menu.

Word Services

When this option is selected, TextWrangler uses an external spelling checker when you choose the Check Spelling command from the Text menu. The external spelling checker must support Apple's Word Services Suite.

To set the external spelling checker, click the Set button and select the application's icon. You can also drag the spelling checker icon to the path box to the left of the Set button.

Startup Preferences

The Startup preferences control what TextWrangler does when you launch it, or when you click on TextWrangler's Dock icon or double-click its icon in the Finder and there are no open windows (even if the application is already running).

Do Nothing

This option specifies that TextWrangler should not open any windows or perform any other actions.

New Text Document

This option specifies that TextWrangler should open a new, empty text editing window.

New Disk Browser

This option specifies that TextWrangler should open a disk browser starting at the desktop.

New FTP Browser

This option specifies that TextWrangler should open an FTP browser.

Open

This option specifies that TextWrangler should bring up the standard Open dialog, allowing you to select and open a file.

Open from FTP/SFTP Server

This option specifies that TextWrangler should bring up the Open from FTP/SFTP Server dialog, allowing you to connect to an FTP server and open a file.

You can hold down the following modifiers during launch to override these actions.

Modifier(s)	Function
Option	Suppress startup items only
Shift	Disable all plug-ins, tools, external services, and startup items

Text Colors Preferences

The Text Colors preferences let you select the colors that TextWrangler uses when the syntax coloring option is on. You can also select a custom window background color and text color, as well as custom highlight colors, which will apply to all text windows, although we feel compelled to point out that there are good reasons why the default Macintosh text color scheme is what it is.

Guide Contrast

You can use this sliding control to adjust the contrast level of the page guide display region. (See "Show Page Guide" on page 188..)

Activating Syntax Coloring

You can turn on syntax coloring globally by setting the Syntax Coloring option in the Editor Defaults preferences panel. You can choose the colors which TextWrangler uses for various content elements, such as keywords, string constants, etc., in the Text Colors preferences panel.

Note   For HTML content, you can turn on the Color Attributes Separately option to have TextWrangler use different colors for HTML attribute names, attribute values, and processing instructions, in addition to anchor, image, and other tags. If this option is turned off, TextWrangler will color HTML attributes identically to their corresponding tags.

How to Change Colors

The color bars show the colors that TextWrangler uses to display different interface and language elements. To change colors, click the color box. TextWrangler will open the system Color Picker dialog box that you can use to select a new color. To restore all the colors to their default settings, click the Reset to Factory Colors button.

Highlight Insertion Point

When this option is turned on, TextWrangler will highlight the line currently containing the insertion point using the indicated color. (The highlighting will be temporarily dismissed whenever a text selection is made.)

Text Encodings Preferences

When opening documents, TextWrangler can automatically recognize and appropriately handle files that use character set encodings other than Mac Roman, including multi-byte character sets.

The Text Encodings preferences panel presents an alphabetical list of every character set encoding available in Mac OS X, and allows you to choose which of these items should appear in the pop-up menus that list encodings. These pop-up menus are:

- The Read As pop-up menu in the Open dialog

- The Encoding pop-up menu in the Options dialog within the Save dialog

- The Encoding submenu of the status bar's File Options pop-up menu

- The encoding pop-up menu in the Text Files: Opening and Text Files: Saving preferences panels

To include an encoding in those menus, select it and click Enable. To remove an encoding from those menus, select it and click Disable. To add all the encodings or remove all the encodings from the menus, use the Enable All and Disable All buttons. To restore the contents of those menus to the factory defaults, use the Restore Defaults button.

All the Unicode encodings are permanently enabled and cannot be turned off.

Tip   To keep the length of the encoding menus manageable, you should add only those encodings which you use frequently.)

Text Files: Opening Preferences

The Text Files: Opening preferences control TextWrangler's behavior when it opens files.

Translate Line Breaks

When this option is selected, TextWrangler translates DOS or Unix line breaks when opening a file. Otherwise TextWrangler leaves the original line breaks untranslated.

If a File's Type Is Unknown

These settings tell TextWrangler how to deal with files whose type and creator codes are not set. Usually, such files are created by Unix programs, but they may also be downloaded from remote servers. You can choose:

- Ignore It: TextWrangler will ignore any file whose type is not 'TEXT' or which is not otherwise recognizable by type as a text file in situations where a text file is required.

- Assume It's Text: TextWrangler will assume that the file should be treated as text. This will give maximum exposure to such files, at the expense of occasionally seeing a file that does not actually contain text (such as binary data or Unix executable files).

- Map the File Name: TextWrangler will inspect the file's name to see if it can figure out whether the file is text or not. TextWrangler will first attempt to map the file name to the list of suffix-to-language mappings specified in the Languages preferences panel. If a file name matches up with a language (even if the language is "None"), the file is assumed to be a text file.

  Thus, you can use TextWrangler's own suffix mappings to convince it to recognize as text any files whose suffixes are not in the system's built-in list of file-suffix-to-file-type mappings. If no match is found in the Languages preferences, TextWrangler will next apply the system's Internet preferences file name mappings.

Note Mac OS X includes a default set of mappings, but provides no direct interface for configuring them. However, you can use Internet Explorer's File Helper preferences, or a third-party System Preferences pane such as RCDefaultApp, for this purpose.

You should select whichever setting makes the most sense for the sorts of files you work with. For example, if you often download or work with files which lack filename extensions, but you *know* that they always contain text, you can select "Assume It's Text".

Link File's Encoding to HTML/XHTML Character Set

When this option is selected, TextWrangler will use the character set specified in the appropriate HTML meta tag or XML declaration to determine a file's encoding when opening the file. Also, when this option is on, changing an HTML or XML document's character set with the Character Set markup command will adjust the file's encoding to match (as indicated on the Encoding submenu of the File Options popup in the status bar), and changing the file's encoding will adjust the character set declaration (if one exists). This option is on by default.

When this option is off, then TextWrangler does not attempt to use the character set specified in the HTML meta tag or XML declaration, but will follow the usual procedure for determining the file's character set. (See "Choosing the Encoding for a Document" on page 37.) The only reason you might want to turn this option off is if you routinely put characters into your document that cannot be represented in the declared character set, e.g. if you will be post-processing the file by some other means which modifies these characters.

If the File's Encoding Can't Be Guessed, use

This menu determines which character set encoding to use when TextWrangler cannot determine the proper encoding by examining the file. This setting also establishes the default setting of the "Read As:" pop-up menu in the Open dialog.

Warn of Malformed UTF-8 Files

When this option is selected, TextWrangler warns you if you open a UTF-8 file that contains an invalid UTF-8 character sequence.

Honor Saved State

When this option is selected, TextWrangler honors state information that may be stored in a file. The following suboptions let you fine-tune which state information TextWrangler honors.

Window Position

When this option is selected, TextWrangler restores the window of the document to the same position as when the file was closed. Otherwise TextWrangler opens the window in its default position.

Font Settings

When this option is selected, TextWrangler restores the font information stored with a document. Otherwise it uses the default font settings.

Selection Range

When this option is selected, TextWrangler restores the insertion point or selection range to the same position as when the file was closed. Otherwise the insertion point is at the beginning of the file.

Emacs Local Variables

Emacs (the popular Unix text editor) supports a convention in which you can define Emacs-specific settings in a block of text near the end of the file, or in the first line of the file. TextWrangler now recognizes the Emacs variable block, and can optionally read and set certain variables.

If this option is selected, TextWrangler will use the "coding" variable, whose value is the Internet text encoding name in which the file is written, e.g. "iso-8859-1" or "utf-8" to interpret the file's contents when opening it.

Scrollbar Position

When this option is selected, TextWrangler restores the scroll bar position to the same position as when the file was closed. Otherwise TextWrangler opens the file with the top of the file showing.

Option Settings

When selected, TextWrangler reads document-specific options, such as soft wrap, show invisibles, and line numbering, from the saved state information.

Text Files: Saving Preferences

The Text Files: Saving preferences control TextWrangler's behavior when it saves files, including file backup settings.

Force New Line at End

When this option is selected, TextWrangler will always add a line break at the end of the file if it does not already end with one.

Default Line Breaks

These options let you choose what kinds of line breaks TextWrangler writes when you save the file. You can choose:

- Macintosh line breaks (ASCII 13) if you are using the file only on a Macintosh.

- Unix line breaks (ASCII 10) if the file resides on a Unix file server or if you are sending it to someone who uses Unix.

- DOS line breaks (ASCII 13/10) if the file resides on a DOS file server or if you are sending it to someone who uses a DOS system

Additionally, if the Use Unicode Line Breaks option is selected, TextWrangler will use Unicode line breaks by default for newly created (or converted) Unicode documents, instead of the chosen platform-specific line breaks.

Default Text Encoding

This menu determines which character set encoding TextWrangler will save a document with, if the document does not contain an encoding specification.

Make Backups Before Saving

Select this checkbox to tell TextWrangler to make automatic backup copies of every file as you save it. This preference setting establishes the defaults for newly created documents, and for documents in which there are no saved backup settings.

If you prefer that TextWrangler create backup files in the same folders that your documents are already saved in, choose the option for Use Document's Folder. This is the default setting. If you prefer to save backups in a different folder, choose the Use Folder setting instead. You can click the Choose button to select the desired backup folder using a dialog, or just drag the folder to the box.

Even if you do not set TextWrangler to perform general backups here, you can still set backup options for individual files by using the Backup Options command in the File menu.

Save Document State

Select this option to have TextWrangler store document state information. A document's state information includes various display properties, such as window position, font settings, etc. This information is now stored centrally by TextWrangler; it is no longer a property of a document's file. You can control how TextWrangler makes use of state information via the Honor Saved State options in the Text Files: Opening preferences panel.

Emacs Local Variables

Emacs (the popular Unix text editor) supports a convention in which you can define Emacs-specific settings in a block of text near the end of the file, or in the first line of the file. TextWrangler now recognizes the Emacs variable block, and can optionally read and set certain variables.

If this option is selected, TextWrangler will change the value of the "coding" variable in the variable block to be the Internet text encoding name which corresponds to the document's encoding (and will write out a variable block if necessary).

Text Printing Preferences

The Text Printing preferences control how TextWrangler prints your documents.

Default Font

To set the default printing font, click the Default Font button. TextWrangler opens the Font panel, in which you can specify the font and font size, and the width of tab stops. The current printing font settings appear in the display box to the right of the button.

Use Document's Font

When this option is selected, TextWrangler uses the document's font and tab settings when printing.

Frame Printing Area

When this option is selected, TextWrangler draws a box along the edges of the printed text.

Print Page Headers

When this option is selected, TextWrangler prints the page number, the name of the file, the time and date printed in a header at the top of each page.

Print Full Pathname

When this option is selected, TextWrangler prints the full pathname of the file being printed in the header.

Print Line Numbers

When this option is selected, TextWrangler prints line numbers along the left edge of the paper.

1-Inch Gutter

When this option is selected, TextWrangler leaves a one-inch margin along the left edge of the paper. Use this option if you usually put your pages in three-ring binders.

Print Color Syntax

If this checkbox is marked, TextWrangler prints colorized text in color. You should generally use this option only on color printers, as colorized text may come out in difficult-to-read dithered shades of gray on black-and-white printers.

Time Stamp

The Time Stamp options let you choose whether the date that appears in the header is the date that the file was last modified or the date that the file was printed.

Print Rubber Stamp

The Print Rubber Stamp option allows you to specify a text string which will print on your documents in enlarged, low density (grey) form using the selected font.

Text Search Preferences

The Text Search preferences let you set default options to use with the Find command.

Report Single-File "Replace All" Results

When this option is selected, TextWrangler displays a dialog telling you how many replacements it made when you perform a Replace All operation on a single file.

Remember Find Dialog's "Start At Top" Setting

Mark this checkbox to have TextWrangler remember the state of the Start At Top checkbox in the Find & Replace dialog from one invocation to the next.

Color Grep Patterns in Find Dialog

When this preference is turned on, and Use Grep is turned on in the Find & Replace dialog, TextWrangler will apply syntax coloring to grep search and replace patterns.

Grep Patterns

This list displays all the grep patterns (regular expressions) you have defined for use in the Find & Replace dialog's Patterns pop-up menu. Click Add to add a new one, click Change to edit the selected pattern, or click Remove to delete the selected pattern.

Note   If you have the Preferences window open, you will not be able to add grep patterns from the Find & Replace dialog. To work around this, close the Preferences window before using the Find & Replace dialog to add new grep patterns.

Text Status Display Preferences

The Text Status Display preferences let you choose which icons and pop-up menus appear in the status bar of TextWrangler text windows.

All Windows:

The following options determine which controls and navigation items TextWrangler should display by default in text windows and text panes.

Show Status Bar

Enable this option to have TextWrangler display the status bar by default. (You can always show it or hide it independently for each window.)

Show Navigation Bar

Enable this option to have TextWrangler display the multiple-document navigation bar by default. (You can always show it or hide it independently for each window.)

Show Cursor Position

This option adds a panel at the bottom-left corner of the window that displays the location (line and column) of the insertion point, or the endpoint of a just-changed selection range (if the cursor has not been moved yet after the change).

Show Current Function

This option displays the name of the function the insertion point is in (if any) at the bottom-left corner of the window.

Defaults

The following options control the presence and default position of text area visual indicators.

Show Page Guide

Enable this option to have TextWrangler display the Page Guide, which is a uniform light grey boundary region starting at the specified character width.

Note   This flexible boundary indicator replaces the historical Philip Bar margin indicator.

Show Tab Stops

Enable this option to have TextWrangler display tab stops as vertical grid lines within the content area of text windows, using the tab width set in the Fonts panel.

Show Line Numbers

This option displays line numbers along the left edge of the window.

Status Bar Controls:

These options let you choose which controls should appear on the status bar. To quickly enable or disable the visibility of all controls, you can click either Show All or Hide All, respectively.

Function Popup

This option displays the Function pop-up menu, which lists the functions (for source code files) or named anchors (for HTML files) in the document, allowing you to jump directly to any function or anchor.

Text Options

This option displays the Text Options pop-up menu, which allows you to set wrapping, display, and editing preferences for the current window, as with the Text Options command in the Edit menu.

Markers

This option displays the Mark pop-up menu, which lets you define and locate named markers in a document.

File Options

This option displays the File pop-up menu, which lets you set end-of-line and other compatibility options.

Insert Menu

This option displays the Insert pop-up menu, which lets you insert the contents of another text file (or a toolbox call template) into the current document.

File Path

This option displays the Path pop-up menu, which shows the location of the file being edited within the folder hierarchy. You can choose any folder from this menu to open it in the Finder.

Get Info Icon

This option displays the Info button, which opens an Info window showing the file's path, along with character, word, line, and page counts for the file and the selection.

Super Get Info Icon

This option displays the Super Get Info button, which asks Super Get Info to display file information pertaining to the current document. This option is available only if you have Super Get Info installed on your system.

Document Icon

This option displays the Document Icon button, which serves as a proxy for the document file. You can click the icon to reveal the current file in the Finder, or drag it anywhere the original file can be dragged.

Toggle Documents Drawer

This option displays the Document Drawer Toggle control at the right-hand side of the status bar.

Function Popup:

The following preference options control how the Function pop-up menu (left) in the status bar behaves.

Show Includes

When this option is selected, the Function pop-up menu includes the files named in #include directives.

Sort Items by Name

When this option is selected, the items in the Function pop-up menu are sorted by name. Otherwise, items in the pop-up menu appear in the same order as they appear in the file.

Show Function Prototypes

When this option is selected, the Function pop-up menu includes the names of function prototypes as well as the function definitions. Otherwise, the pop-up menu does not include entries for function prototypes.

Tools Preferences

The Tools preferences control the way TextWrangler integrates with programming tools, such as AppleScript editors, development environments and Toolbox references.

Script Editor

To set the script editor for use with the Open Script Editor command in the Script menu, do one of the following:

- Click the Set button and select the script editor from the dialog box.

- Drag the icon of the script editor to the path box.

Coding Tools

The Coding Tools option lets you choose whether TextWrangler will display the Shebang menu for working with Unix scripts (see page 220). In order for a change of this option to take effect, you must quit and relaunch TextWrangler.

Install Command Line Tools

The "Install Command Line Tools" button installs the current version of the "edit" and "twdiff" tools for invoking TextWrangler from the Unix command line. The first time you run TextWrangler after installation, it offers to install these tools for you. If you choose not to do so, you can use this button in the Tools preferences panel to install the tools at a later time.

If the tools are already installed, the button will update them to a newer version if one is available; it will not overwrite existing versions of the tools with older versions. See Chapter 12 for further details on the "edit" and "twdiff" command line tools.

Unix Scripting Preferences

The Unix Scripting preferences control how TextWrangler handles files which are run as Unix filters or scripts.

Use UTF-8 for Unix Script I/O

Mark this checkbox to have TextWrangler always send UTF-8 to Unix filters, and interpret Unix script output as UTF-8. This precludes the previous need for TextWrangler to create and execute a temporary copy of the script file, which could result in different behaviors for a script when run inside and outside of TextWrangler.

Warn About Non-Unix Line Breaks Before Running

Mark this checkbox to have TextWrangler post an alert when you attempt to run a filter or script which does not have Unix line breaks. Although TextWrangler transparently handles this case, most external environments do not; thus, it is recommended that you leave this option on.

Use Affrus for Perl Debugging

Mark this checkbox to have TextWrangler use Affrus in preference to the command line Perl Debugger. Affrus is an integrated Perl development environment from Late Night Software.

```
http://www.latenightsw.com/
```

If you choose this option, TextWrangler will pass complete environment parameters and control to Affrus, but will not attempt to retrieve output. If Affrus is not installed, this option will be disabled.

Windows Preferences

The Windows preferences control the size and appearance of both newly created windows and windows that do not contain their own display state information. (See the State panel to tell TextWrangler how to store state information in files.)

Window Zooming

Always Zoom Windows

Mark this checkbox to always expand windows to their maximum size when opening them, regardless of their saved window size state.

Move As Little As Possible

When this option is selected, TextWrangler keeps windows as close to their original location as possible when you zoom them.

Maximum Width

This pop-up menu lets you specify the maximum width of a zoomed window. The default setting is Classic Display.

Zoom Windows To

On systems with multiple monitors, this pop-up menu lets you choose which screen TextWrangler should use when zooming. You can choose the main screen (the one containing the menu bar), the nearest screen (the one containing most of the active window), the largest screen, or the smallest screen.

Window Menu

Always Show Full Paths

Mark this checkbox to show the complete pathname of open documents, rather than just their names, in TextWrangler's Window menu. If two or more files have identical filenames, their complete pathnames will always be shown to prevent ambiguity, regardless of this setting.

Group by Window Kind

Mark this checkbox to group windows of the same kind together in the Window menu. For example, text editing windows, disk browsers, and search results browsers are all different kinds of windows. Within each group, windows will be sorted as determined by the Sort Windows By radio buttons (see below).

Auto-Assign Shortcut Keys

Mark this checkbox to have TextWrangler automatically assign shortcut key equivalents of Command-0 through Command-9 to text windows or shell worksheets as these windows are created. If you leave this option off, these key equivalents will become available for assignment to other menu items via the Set Menu Keys command.

Sort Windows By

Select Name to list windows alphabetically by name, or Creation Order to list them in the order they were opened in the current TextWrangler session.

Window Stacking

These icons determine how TextWrangler stacks windows: down and to the left, straight down, directly atop, or down and to the right. TextWrangler stacks windows down and to the left by default.

Leave Room for Palettes

When this option is on, TextWrangler leaves room for any open floating palettes when creating or rearranging windows, if the palettes are stacked together such that at least one is against either the right-hand or left-hand edge of the screen. This option is on by default.

Leave Room for DragThing Docks

When this option is on, TextWrangler will also leave room for any visible DragThing docks when creating or rearranging windows.

Leave Room for Finder

When these options are selected, TextWrangler leaves room at the right side and/or the bottom of the screen so that you can see icons on the desktop. These options are off by default.

Optional settings via 'defaults write'

In addition to the preference settings which can be made through the Preferences window, TextWrangler permits the following additional modifications to its behavior, which you can make by issuing an appropriate "defaults write" command.

Remove FTP/SFTP Commands from Menus

If you never use TextWrangler's FTP/SFTP-related commands, such as Open from FTP/SFTP Server, Save (A Copy) to FTP/SFTP Server, etc., you can remove them from the menus by issuing the following command in the Terminal, and then quitting and relaunching TextWrangler.

```
defaults write com.barebones.textwrangler FTP:HideFTPMenuCommands -bool
true
```

If you have applied this change, but want TextWrangler to display the FTP/SFTP menu commands again, issue the following command in the Terminal, and then quit and relaunch TextWrangler.

```
defaults write com.barebones.TextWrangler FTP:HideFTPMenuCommands -bool
false
```

Add Temporary Files to the Open Recent Menu

Ordinarily, TextWrangler will not add files to the Open Recent menu if they are located in the system temporary directories (/tmp or /private/tmp). However, if you routinely work with such files and want them to appear on the Open Recent menu, you may issue the following Terminal command.

```
defaults write com.barebones.textwrangler RememberRecentTempFiles -bool
true
```

If you have applied this change, but no longer want to have such files added to the Open Recent menu, issue the following command in the Terminal.

```
defaults write com.barebones.textwrangler RememberRecentTempFiles -bool
false
```

Scripting TextWrangler

TextWrangler offers access to nearly all of its features and commands via AppleScript. This chapter provides a brief overview of AppleScript, discusses TextWrangler's scripting model, and explains how you can use scripts with TextWrangler.

An excellent way to learn how to script TextWrangler is to look at the scripts others have written for it (or for BBEdit), or to turn on recording in your script editor while you perform actions in TextWrangler. A number of example scripts are available from the Bare Bones Software web site. The TextWrangler-Talk mailing list is also a good resource for learning more about scripting. To subscribe to this list, please visit the support section of our web site.

```
http://www.barebones.com/support/lists.html
```

IIMPORTANT   Regardless of whether you are new to scripting TextWrangler or are familiar with scripting previous versions, we strongly recommend that you carefully review the sections "TextWrangler and AppleScript" and "Working with Scripts" in this chapter.

In this chapter

AppleScript Overview

If you are familiar with AppleScript, you should have little difficulty scripting TextWrangler. It has a robust and highly flexible object model. If you do not know much about scripting, though, read on for an introduction to the necessary concepts.

About AppleScript

AppleScript is an English-like language which you can use to write scripts that automate the actions of applications, and exchange data between applications. Although AppleScripts can manipulate applications' user interfaces by taking advantage of Mac OS X's GUI Scripting capability, this is not their primary function. Rather, scripts talk directly to a application's internals, bypassing its user interface and interacting directly with its data and capabilities.

If you want to insert some text into a document, emulating a user typing into an editing window is not the most efficient way of accomplishing this. With AppleScript, you just tell the application to insert the text directly. If you want the application to save the frontmost document, you need not mime choosing Save from the File menu, but rather just tell the application to save its frontmost document.

Note AppleScript is actually a specific language, which resides atop the general Open Scripting Architecture (OSA) provided by Mac OS X. Although AppleScript is by far the most popular OSA language, there are others, including UserLand Frontier and a JavaScript version. All OSA languages are capable of accomplishing similar things, although the actual commands you would use differ from one language to the next. In this chapter, we will focus exclusively on AppleScript, since it is the standard scripting language, but you should bear in mind that there are other options.

Scriptable Applications and Apple Events

Since AppleScripts must have direct access to an application's internal data structures, any application that will be used in an AppleScript must be designed to allow this access. We say such applications are *scriptable*. TextWrangler is scriptable, as are many, many other programs. However, it is important to note that not *every* application is scriptable, and AppleScripts are not the best solution for automating applications that are not.

What goes on in an application that is scriptable? The foundation of AppleScript is something called the *Apple Event*. Macintosh applications are designed around an event loop; they go around in circles waiting for you, the esteemed user, to do something (choose a menu command, press some keys, and so on). These actions are passed to the application by the operating system in the form of an *event*. The application decodes the event to figure out what you did, and then performs an appropriate operation. After an event has been handled, the application goes back to waiting for another one. (At this point, the Mac OS may decide to give some time to another application on your computer.)

Apple Events are special events that applications send to each other, enabling a feature called *inter-application communication* (IAC). (It's a mouthful, but it just means applications can talk to each other.) Apple Events are also the way AppleScripts tell applications what to do, and which data to retrieve. So to be scriptable, an application must first support Apple Events.

Apple Events in their naked form are raw and cryptic things—bits of hieroglyphics only a programmer could love. So a scriptable application also has a *scripting dictionary*. The scripting dictionary tells any application that lets you write AppleScripts, such as the standard Script Editor, the English-like equivalent for each Apple Event and each event's parameters.

It is important to note that because Apple Events were originally designed to allow applications to communicate with each other, AppleScripts automatically inherit the ability to talk to more than one application. It is common in the publishing industry, for instance, to write scripts that obtain product information from a FileMaker Pro database and insert it into a Quark XPress file. This integration is one of the Macintosh's primary strengths.

You use AppleScript's *tell* verb to indicate which application you are talking to. If you are only sending one command, you can write it on one line, like this:

```
tell application "TextWrangler" to count text documents
```

If you are sending several commands to the same application, it is more convenient to write it this way:

```
tell application "TextWrangler"
    count text documents
    repeat with x from 1 to the result
        save text document x
    end repeat
end tell
```

The Script Editor automatically indents the lines inside the *tell* block for you so you can more easily follow the organization of the script.

Reading an AppleScript Dictionary

To display an application's AppleScript dictionary, you can simply drag that application onto the Script Editor icon, or use the Script Editor's Open Dictionary command. As we noted earlier, all scriptable applications include a dictionary that tells AppleScript how to convert English-like commands into the Apple Events actually expected by the application. The Script Editor uses this same information to display a sort of "vocabulary guide" that helps you write your scripts.

We will naturally use TextWrangler's dictionary, shown below, to illustrate how to read a dictionary.
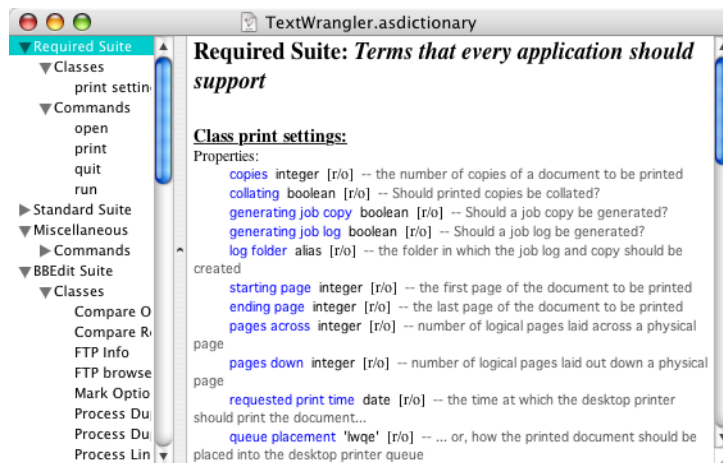
(You will probably want to make the window bigger if you have room on your screen.)

Down the left side is a list of every event and object supported by the application. An event is a verb—it tells the application what to do. A class is a noun: a piece of data, or a structured collection of data, inside the program. In TextWrangler, for instance, classes are things like files, windows, the clipboard, browsers, and so on.

Suites

The first thing you will notice is that the events and classes are divided into *suites*. A suite is just a collection of related events and classes. Apple, for instance, has decreed that all applications should support particular events, which together are called the Required Suite. Another Apple-defined suite is the Standard Suite: if an application has certain common functions, they should use these standard terms, so that scripters do not need to learn a new term for each application they work with. After that, it is a free-for-all—each developer is free to organize their events and classes however they think best.

In addition to the Required and Standard suites, TextWrangler has a Miscellaneous suite, a BBEdit Suite, and aText suite. Additionally, if you have any scriptable plug-ins installed (as many of the supplied ones are), you will see additional suite entries for each such plug-in.

Within each suite, events—verbs—are displayed in normal text, while classes—nouns—are italicized. Most commands sent to TextWrangler will start with one of the verbs. (In some cases, *get* might be implied.)

Events

Let's look more closely at one of the events—*Save* is a good one to start with. It is shown below.
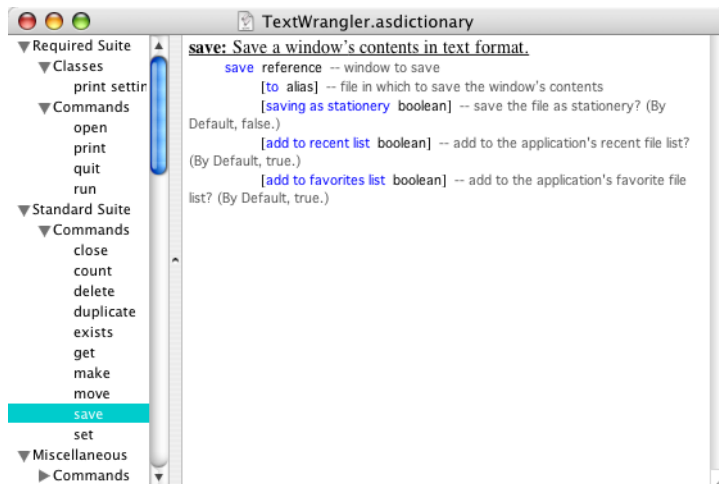
The right side of the window shows the syntax of the selected event, as well as a brief description of its function. The boldface words are keywords; they must be included exactly as shown or the script will not compile. The normal text tells you what kind of information goes after each keyword. For example, after *save* you must give a reference; the italicized comment next to that line indicates that it is a reference to the window to be saved. In other words, some window object, which in TextWrangler would be *window 1* for the frontmost window, or *window "Text File"* if you want to specify a window by name. (we will show you how to figure all that out in a moment—you have to look at the window class's dictionary entry.)

Anything in square brackets is optional. Most of the rest of the *save* event is optional, in fact. The basic event just saves the frontmost window to the same file from which it was opened. However, you can also optionally include the word *to* followed by a file reference. (You specify a file simply by using the word *file* followed by the path name of the file, as in *file "Hard Disk:Users:BBSW:Documents:My file".)* If you specify a file to save the window to, the text will be saved into that file instead of the file it came from—like using Save As instead of Save.

The last three optional parts of the *save* event are denoted as boolean. That means they take either a true or a false value. In AppleScript, there are a couple of different ways to specify boolean values. You can write *saving as stationery true* to tell TextWrangler to save the file as a stationery document. Or you can write *with saving as stationery.* You will notice that the last two parameters default to true if you do not specify them as false. To do that, you would use *add to recent list false* or *without add to recent list.* Whichever way you write it, you will notice that when you compile the script, AppleScript rewrites it using "with" or "without". Since that is the syntax AppleScript seems to like best, that is probably the one you should get used to thinking in.

Let's take a look at another one: the prosaic *get.* Select *get* from TextWrangler's dictionary listing and take a quick look at its class definition.You use *get* to retrieve information from an application. You must specify a reference to the object you want to retrieve, and you can specify a *coercion*—a condition that tells AppleScript to treat one type of data as if it were another—by adding the *as* clause. However, after that is the *Result:* line, which we have not seen before. This line tells you what type of value the command returns. (This value is placed in the AppleScript system variable called *the result.) Get* can retrieve any kind of object, so it can return anything, as indicated here. Other events might return a specific type of result, or none at all. *(Save* did not have a *Result:* line in its dictionary entry, which means it does not return a result.)

Classes and the Class Hierarchy

Let's look now at a typical class definition: *window* will do nicely. It is in the BBEdit Suite, toward the bottom.
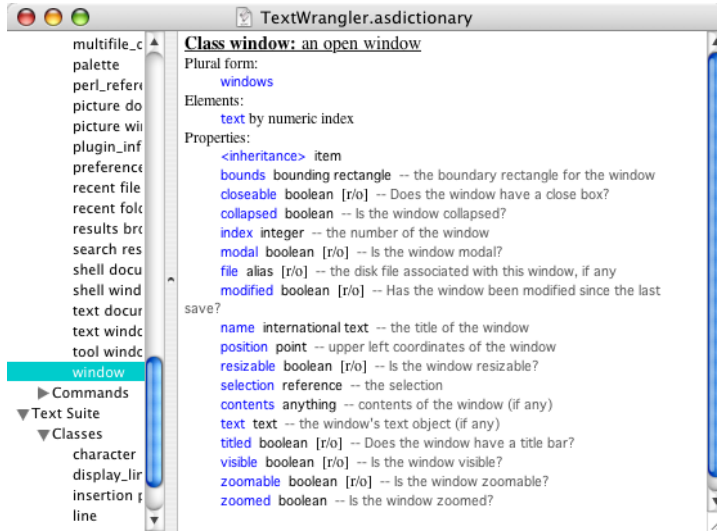
All windows in TextWrangler belong to this class. A class defines a particular kind of object; a particular example of an object belonging to the class is said to be an instance of that class, or just an object of that class. So here we are looking at the class itself; each individual window object has all these properties.

After a tag line that tells you about the class ("an open window") comes the plural form. AppleScript lets you refer to windows either singly or as a group, so it needs to know what the plural of every term is. For example, try this little script:

```
tell application "TextWrangler" to count windows
```

The result of this script is the total number of window objects currently displayed by TextWrangler.

After the plural form comes a list of properties. Some objects do not have properties—for example, a string—but many applications do. An object's properties are merely a collection of data that describes that particular object. For example, as you look down the list of window properties, you will see that every window has a name, every window has a position, every window has bounds (the area of the screen it covers), and so on.

The first item on the list, though, is *<inheritance> item*. This tells you that a window is a kind of item, and that it therefore has all the properties of an item. Take a quick look at *item's* class definition, shown below.
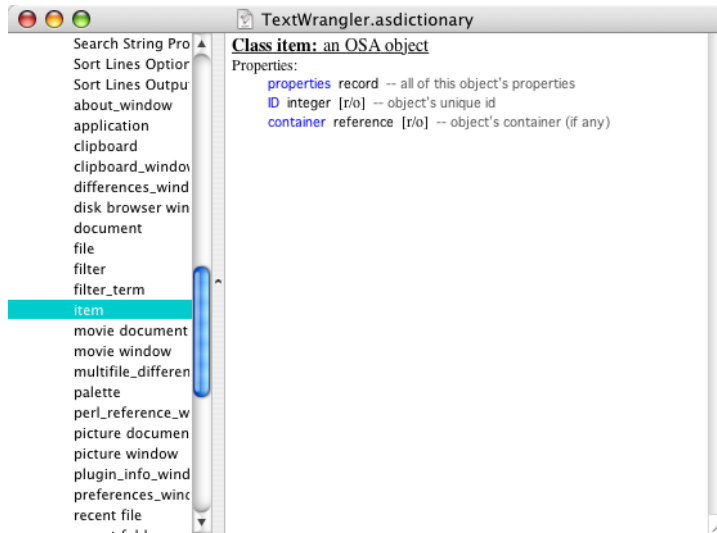
You will see three properties: *properties, ID*, and *container*. The first entry *properties* is a record containing all the object's properties. In other words, because a window is an item, it has, in addition to all its listed properties, another property which returns all the other properties as a record—a single piece of data that can be stored in a variable. Every class in TextWrangler is part of a hierarchy with the *item* class at the top, so every object in TextWrangler "inherits" the *properties* property. This catch-all property can be handy for making exact duplicates of objects, among other uses.

You may realize that TextWrangler has several kinds of windows; you can see their classes listed in the dictionary: clipboard window, differences window, disk browser window, text window, tool window, and the like. Let's look at *text window:*
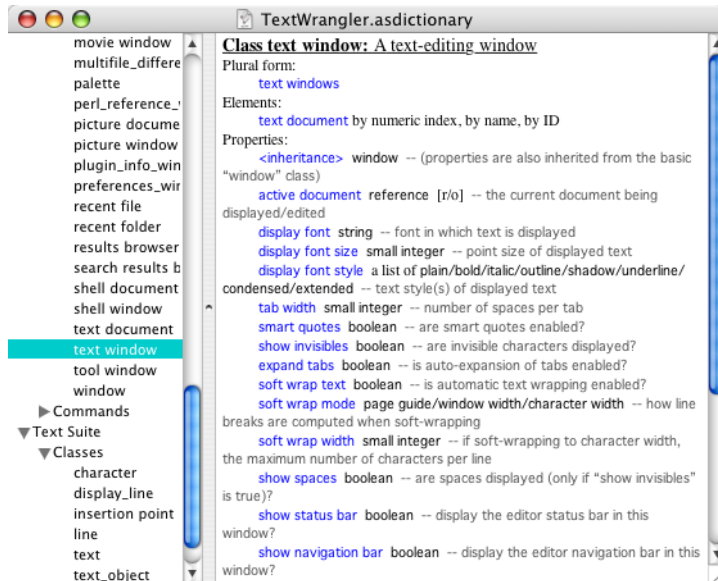
You can see that a text window inherits all the properties of the *window* class. And, since the *window* class inherits all the properties of the *item* class, this means that the *text window* class also has the *properties* property defined by the *item* class.

To make explicit what you might have already gathered, classes in AppleScript form a hierarchy. That is, classes can be based on other classes. Such a class is called a *subclass*, and the class on which a subclass is based is referred to as its *parent class*. (In AppleScript, classes can only have one parent. Multiple inheritance is a feature found in more complex languages.)

The idea of a class hierarchy makes it easier for us to add new features to TextWrangler, since when we want to create a new kind of window, half the work is already done. However, when scripting, you may need to flip back and forth between two or more class definitions to find all the properties of the object you are working with. (This is, technically speaking, a limitation of Apple's Script Editor. There is no reason the inherited properties could not automatically be included in a subclass listing by a smarter editor, for example, Script Debugger, which does this.)

Now that we have the class hierarchy under control, let's look at the properties themselves more closely. we will stick with the *text window* class at this point.

Properties of an object are referred to using the preposition *of.* For example, the following line of script returns the font of the frontmost text window.

```
tell application "TextWrangler" to get display font of
text window 1
```

Note   In this specific example, you can just write *get display font of window 1*. AppleScript will figure out that window 1 is more specifically a text window, and therefore has a *display font* property, even though the generic *window* class does not have any such property. All the properties of the object are available even if you did not use its specific class name. However, in most cases, you should specify exactly the object you want; this distinction is especially important when dealing with text documents (content) versus text windows (display elements).

You can set the properties using the *set* event, like so:

```
tell application "TextWrangler" to set display font of text
window 1 to "Geneva"
```

Let's go back to the *window* class for a moment. Most of the properties of this class are marked with the abbreviation *[r/o]*. That stands for Read-Only. In other words, you can only *get* these properties, not *set* them.

Recordable Applications

Once an application accepts Apple Events, it actually makes a good deal of sense for an application to be designed in two parts: the user interface that you see, and the "engine" that does all the work. (An application designed this way is sometimes said to be *factored*.) The user interface then communicates with the engine via Apple Events.

The design of the Apple Event system makes it possible to "record" events into a script. This feature not only lets you automate frequently performed tasks with little hassle, it also can be an enormous aid in writing larger and more complicated scripts, because the application tells you what events and objects to use for the kind of task you record.

Because of the important recording functionality they enable, applications that have been factored and use Apple Events to let the two halves communicate are said to be *recordable.* It is important to note that not all scriptable applications are recordable.

Saving Scripts

Any AppleScript can be saved in what's called a *compiled script file*. A compiled script file contains the actual Apple Events; by generating these events when you save the file, the operating system does not have to convert your English-like commands into events each time you run the script, which means it loads faster. When double-clicked in the Finder, a compiled script file automatically opens in the Script Editor, where it can be run. A script can also be saved as a stand-alone application, or *applet*, in which case double-clicking the script's Finder icon automatically runs the script. Both types of files can be saved with or without the English-like *source code;* if you save it without the source code, other users you give the script to will not be able to make any changes to it (of course, you should also keep a copy of the script *with* the source for yourself).

Using Scripts with Applications

Although you can place a script applet in the global Scripts menu, or in any folder, and use it any time you need it, many applications (including TextWrangler) provide a special menu that lets you launch compiled scripts intended specifically for use with that one application. Since you do not have to save them as applets, they take up less disk space and launch more quickly. They also show up only in the application you use them with, rather than cluttering your global Scripts menu.

Some applications go even further, allowing you to define scripts to be run when certain things happen in the program. For example, an application might let you define a script to be executed when the user chooses *any* menu item. The script might then perform some pre-processing, and then exit by telling the application whether to continue with the menu command or to cancel it. As a simple example, a script might check to see what printer is selected when the user chooses the Print command. If it is the expensive color dye-sublimation printer, on which printing a page costs several dollars, the script could remind the user of that fact and confirm their intention (through an alert) before continuing with the print operation.

An application that supports such a feature (or any method of integrating user-written scripts seamlessly into its user interface) is said to be *attachable*, because the scripts become "attached" to the features of the program. (TextWrangler is now attachable; more details about using this feature are provided later in this chapter.)

Scripting Resources

Covering all the details you might need to write your own AppleScripts is not something we can reasonably do in this manual. AppleScript, despite its deceptively simple English-like syntax, is a sophisticated object-oriented language with many subtleties. For this reason, we suggest you consult supplemental documentation and resources if you are a beginning scripter.

A good place to start is with someone else's script: find a script that does *almost* what you want it to and repurpose it. Even if you cannot find a script that does anything close to what you want, reading others' scripts is a good way to learn how AppleScript "thinks" and how TextWrangler's particular AppleScript implementation behaves.

In addition to the basic AppleScript documentation included with the system, you may find the following resources useful in your quest to understand scripting.

Books
AppleScript: The Definitive Guide, Matt Neuberg. O'Reilly and Associates, 2003.
ISBN: 0-59600-557-1

AppleScript in a NutShell, Bruce W. Perry. O'Reilly and Associates, 2001.
ISBN: 1-56592-841-5.

Mailing Lists

AppleScript Users
`http://www.lists.apple.com/applescript-users.html`
The official list for AppleScript users run by Apple Computer.

TextWrangler-Talk, BBEdit-Scripting
`http://www.barebones.com/support/lists.html`
The BBEdit-Scripting discussion list often covers scripting topics which also apply to TextWrangler; either of these lists is a good place to ask questions about TextWrangler's AppleScript implementation.

Mac Scripting
`http://www.its.unimelb.edu.au/hma/pub/macscrpt/`
Unofficial list covers AppleScript and other Macintosh scripting languages, with occasional forays into peripheral topics.

Web Sites

AppleScript at Apple Computer
`http://www.apple.com/applescript/`
This is the starting point for AppleScript from the people who invented it. Includes a tutorial and a good amount of technical information.

The AppleScript Sourcebook
`http://www.AppleScriptSourcebook.com/`
An extensive collection of links and articles about AppleScript.

AppleScript Primer
`http://www.maccentral.com/columns/briggs.shtml`
MacCentral columnist Bill Briggs offers an ongoing series of lessons for beginning scripters. Quite a range of topics covered, increasing in difficulty as time goes on. Note that the oldest columns are on the bottom.

MacScripter.Net
`http://macscripter.net/`
 and
`http://osaxen.com/`
A good selection of AppleScript-related news and topics, including "MacScripter's Magazine" (a stand-alone multimedia tutorial for AS beginners), and a very comprehensive list of scripting additions on its related site.

ScriptWeb
`http://www.scriptweb.com/`
This site covers all scripting languages, not just AppleScript. Also, it has an extensive directory of scripting additions.

Software

Script Debugger
`http://www.latenightsw.com/`
Despite the name, it is more than a debugger; it is actually an enhanced replacement for Apple's Script Editor, featuring variable monitoring, step/trace debugging, an object browser for an application's objects, and much more.

Using AppleScripts in TextWrangler

In addition to providing extensive script access to its commands and data, TextWrangler is both attachable *and* recordable, making it an extremely flexible and customizable tool.

This section describes how you can create and employ AppleScripts within TextWrangler via recording and TextWrangler's various scripting facilities, while the following section covers TextWrangler's scripting commands and other issues related to preparing scripts for use.

Recording Scripts in TextWrangler

Any language is easier to read than to write, easier to understand than to speak. AppleScript is no different. That's because, even though all the commands it uses are English words arranged in ways that more or less make grammatical sense, you still have to know (or find out from the application's dictionary) exactly which words to use, and what order they should go in. But it is easy to get started making scripts by recording them.

First, launch both TextWrangler and the Script Editor.

When you launch the Script Editor, a new, blank script window appears. Click the Record button, circled in the illustration below.
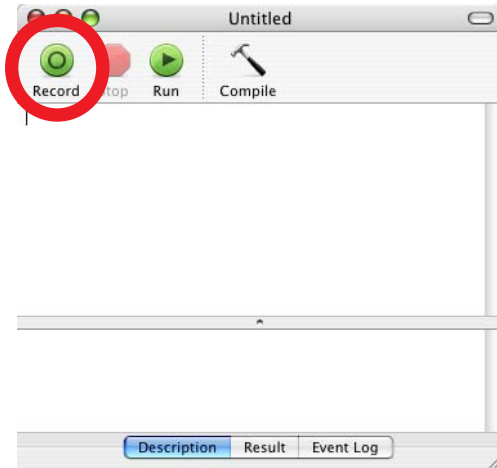
Now switch to TextWrangler and perform your task. Remember that the Script Editor is recording *everything* you do in every recordable application you are running, not just TextWrangler. If you do something in the Finder, for instance, that will get recorded too. Since almost everything you do is recorded, remember that if you make an error, and then Undo it, your recorded script will faithfully make the same mistake and undo it when you run it later. It will be possible to fix minor errors later, but things always go more smoothly if you do not make any mistakes, so take your time and try to do it right the first time.

Now switch back to the Script Editor and click the Stop button. After a brief pause, your script is compiled and ready for use. Try clicking the Run button to see it work. (It might not work correctly. If you recorded a search and replace operation changing every "cat" to "dog", you already changed the document while recording the script, and of course the script will not do anything when you run it.)

Finally, save the script in the TextWrangler Scripts folder so that it shows up in TextWrangler's script menu. Choose Save As from the File menu, and then use the Script Editor's Save dialog to put the script in your TextWrangler Scripts folder. Now try selecting it from the script menu in TextWrangler.

The Scripts Menu

The Scripts menu (left) in TextWrangler's menu bar contains several commands. It also lists all AppleScripts present in the Scripts folder within TextWrangler's application support folder, providing a quick way to access frequently used scripts. You can place scripts within subfolders (up to 4 levels deep) of the Scripts folder to organize them.

Note   Scripts written for use in the menu should be saved as compiled script documents, not script applications.

In addition to the list of available scripts, the Scripts menu provides the following commands.

Open Script Editor

Choose this item to switch to your preferred AppleScript editor (as chosen in the Tools panel of the Preferences window). If the script editor is not running, TextWrangler launches it.

Open Scripting Dictionary

Choose this item to switch to your preferred AppleScript editor and open TextWrangler's scripting dictionary for viewing. If the script editor is not running, TextWrangler launches it.

Open Scripts Folder

Choose this item to open the Scripts folder which is located within TextWrangler's application support folder. (See "TextWrangler's Application Support Folders" on page 25.)

Start Recording

Select this item to record all available actions that you perform within TextWrangler (or any other recordable applications which you switch to). When this command is active, the menu item will change to Stop Recording, and a tape icon will flash over the Apple menu. When you choose Stop Recording, TextWrangler will display a Save dialog which allows you to save a script file containing the recorded actions.

Running and Editing Scripts

Choose the item corresponding to any script to run that script. Hold down the Option key when choosing a script item to have TextWrangler open the script for editing in your preferred script editor, or hold down the Shift key when choosing a script item to have TextWrangler reveal the script file in the Finder. If you choose a folder node rather than a script item, TextWrangler will open the corresponding folder in the Finder.

The Scripts Palette

The Scripts command, located in the Palettes submenu of the Window menu, opens a palette listing all available scripts. Names that are too long to fit within the width of the window are truncated with ellipses (…).

"Hovering" the mouse over such a truncated name displays a tool tip showing the full name. If you hold down the Option key, the tool tip will appear instantly, with no hovering delay. Names that fit entirely within the window without truncation do not display a tool tip.

Organizing Scripts

Items in the Scripts menu or Scripts window are displayed in alphabetical order by default, but you can force them to appear in any desired order by including any two characters followed by a right parenthesis at the beginning of their name. (For example "00)Save All" would sort before "01)Close All.") For names of this form, the first three characters are not displayed in the window. You can also insert a divider by including an empty folder whose name ends with the string "–***". (The folder can be named anything, so it sorts where you want it.) These conventions are the same as those used by the utilities FinderPop and OtherMenu.

Attaching Scripts to Menu Items

TextWrangler lets you attach scripts to menu items. By this, we mean that you can write scripts that TextWrangler automatically calls before or after performing a menu command. For example, if you want TextWrangler's Open from FTP/SFTP Server command to launch your favorite FTP client, you can simply attach a script to that menu item. Scripts can return a value that tells TextWrangler whether to continue with the command that was selected, or to cancel the operation (in which case only the script is executed).

Scripts attached to TextWrangler menu items must be stored in the Menu Scripts folder of TextWrangler's application support folder. These files should be compiled scripts, not script applications. Scripts are named to indicate which menu item they go with: first the name of the menu (or the submenu) upon which the item is immediately located, then a bullet "•" (Option-8) character, then the name of the menu item. For example, to attach a script to the Open from FTP/SFTP Server menu item, you would name it "File•Open from FTP/SFTP Server", while to attach a script to the New Document menu item, you would name it "New•Text Document".

Some of TextWrangler's menus have icons rather than names. TextWrangler uses the following names for its icon menus: "#!" [the 'Shebang' menu], "Scripts", and "Text Factories". Furthermore, the New With Stationery submenu is named "Stationery" for purposes of attachability.

When you choose a menu item with an attached script, TextWrangler runs its MenuSelect handler, if it has one, passing it the menu name and item name of the selected menu item as parameters. If no MenuSelect function is present, TextWrangler executes the script's run handler. The MenuSelect handler can return a boolean value to indicate whether TextWrangler should continue by performing the action usually invoked by the menu command ("false" means yes, "true" means stop after executing the script). If MenuSelect returns false, TextWrangler will call the script's PostMenuSelect handler, if it has one, after it performs the menu command.

Here is a simple example, which adds a confirmation dialog to the Save command (addressed as "File•Save"). Note that we test the menu and item names to make sure the script is attached to the Save command—if it is attached to some other command, it does nothing.

```
on menuselect(menuName, itemName)
  if menuName = "File" and itemName = "Save" then
    set weHandledCommand to true
    display dialog "Are you sure you want to save?" ¬
    buttons {"No", "Save"} default button 2
    if button returned of the result is "Save" then
    -- the application should do its work
    set weHandledCommand to false
    else
    -- we handled the command, app does no work,
    -- postmenuselect doesn't get called
    display dialog "The document was not saved." ¬
      buttons {"OK"} default button 1
    end if
    return weHandledCommand
  end if
end menuselect

on postmenuselect(menuName, itemName)
  -- this is called after the application has processed
  -- the command
  display dialog "The document was saved." ¬
    buttons {"OK"} default button 1
end postmenuselect
```

TextWrangler's Scripting Model

This section provides a high-level overview of TextWrangler's scripting model that will, where appropriate, contrast the current scripting framework against older versions of TextWrangler, and suggest how you can modify your existing scripts for compatibility.

IMPORTANT   Because TextWrangler's scripting dictionary changes whenever we add features, it should be considered the definitive reference in any situation where it and this document differ. We have found Script Debugger from Late Night Software to be an excellent tool for browsing and navigating TextWrangler's scripting dictionary, as well as for preparing and testing scripts.

```
http://www.latenightsw.com/
```

Script Compatibility

Now that TextWrangler supports opening multiple documents in a single text window, its scripting model has changed significantly. Consequently, you may need to revise scripts prepared for earlier versions in order to have them work properly.

Distinguishing Between Script Elements

Because different applications handle different types of data, you should be aware that the actual data, or the interface items, referred to by a particular name may not be consistent from application to application. The following sections describe how several common elements are handled in TextWrangler.

"Lines" and "Display_lines"

The "line" element refers to a "hard" line, that is, a stream of characters that begins at the start of file or after a line break, and which ends at the end of file or immediately before a line break. This is consistent with the previous semantics of "line" in hard-wrapped documents, and these semantics now apply in soft-wrapped documents as well.

The "display_line" element refers to a line of text as displayed on screen (bounded by soft and/or hard line breaks).

The "startLine" and "endLine" properties of a text object now always refer to the "hard" start and end of lines. In other words, if a text object crosses multiple soft-wrapped lines, the startLine and endLine properties will be the same.

Both "startDisplayLine" and "endDisplayLine" properties are now part of the text object class. These serve the same purpose as the startLine and endLine semantics for soft-wrapped views in older versions of TextWrangler.

Documents vs. Windows

In older versions of TextWrangler, the object classes *document* and *window* could be used interchangeably, and generally had the same properties listed in the scripting dictionary. This is no longer the case.

The class *window* now corresponds to a window (of any type—text or otherwise) on screen, and thus the properties of the *window* class now refer strictly to properties of a window on screen. If a document is associated with a window, the document is accessed as the *document* property of the window:

```
document of text window 1
```

The class *document* refers to a document, and as with a window, the document's properties pertain strictly to the condition of a document (that is, something that can be saved to disk and opened later). Note that this does not mean a document must be saved to a file, only that it could be.

As a rule, documents and windows are associated with each other, but it is important to remember that there is not a one-to-one correspondence between windows and documents. For example, the About box is a window which has no document associated with it. Furthermore, in current versions of the application, there is no such thing as a document with no associated window.

Here is a general overview of the object classes used in TextWrangler:

Classes of Windows

- *window*: the basic window class contains properties that can be fetched and set for any window on screen: position, size, and so forth.

- *palette*: the palette class refers to windows that float above all others on the screen, i.e. any window opened from the Palettes submenu of the Window menu.

- *text window*: the text window class provides properties which are specific to text-editing windows as on-screen entities. These properties pertain mostly to the display of text in the window: *show invisibles*, *auto_indent*, and so on. In addition to the text-editing-specific properties, the basic window properties are also accessible.

- *disk browser window*: provides a way to reference windows corresponding to open disk browsers. A disk browser window does not present any properties beyond the basic *window* class, but provides a way to differentiate disk browser windows from other types of window.

- *results browser*: provides a way to reference results generated by a batch operation. A results browser does not present any properties beyond the basic *window* class, but provides a way to differentiate results windows from other types of window.

- *search results browser*: a subclass of results browser, referringspecifically to the results of a single-file Find All command or a multi-file search.

Classes of Document
As with windows, there are various classes of document:

- *document*: the basic document class contains properties that apply to any sort of document: whether it has unsaved changes, the alias to the file on disk, and so on.

- *text document*: text documents contain information specific to text files opened for editing in TextWrangler.

- *picture document*: refers to a document corresponding to an open picture file. A picture document does not present any properties beyond the basic *document* class, but provides a way to differentiate picture documents from other types of document.

- *movie document*: refers to a document corresponding to an open QuickTime movie file. A movie document does not present any properties beyond the basic "document" class, but provides a way to differentiate movie documents from other types of document.

- *QuickTime document*: refers to a document corresponding to an imported Quicktime image file. A QuickTime document does not present any properties beyond the basic "document" class, but provides a way to differentiate QuickTime documents from other types of documents.

Getting and Setting Properties

One significant improvement in TextWrangler's new scripting framework is the ability to get and set multiple properties of an object with a single scripting command. Every object has a property called *properties*. This property returns a record which contains all of the properties which can be fetched for that object. For example, the script command

```
properties of text window 1
```

will return a result like this one:

```
{{id:55632400, container:application "TextWrangler", bounds:{31,
44, 543, 964}, closeable:true, collapsed:false, index:1,
modal:false, file:alias "Hard
Disk:Users:Shared:doc_examples:index copy.html", modified:false,
name:"index copy.html", position:{31, 44}, resizable:true,
selection:"", contents:"..."}
```

Conversely, to set one or more properties at once is very easy:

```
set properties of text window 1 to { show invisibles: true, show
spaces : true, soft wrap text : true }
```

Only the properties specified will be changed. The rest will not be modified.

It is important to note that when setting properties in this fashion, you can only set modifiable properties. If you attempt to set any read-only properties, a scripting error will result:

```
set properties of text window 1 to { show invisibles: true,
modal: false, expand tabs: true }
```

The above script command will turn on Show Invisibles and then report a scripting error, since *modal* is a read-only property.

Performing Actions

The following sections provide basic information on how to perform various common actions via AppleScript.

Scripting Searches

The ability to script searches presents you with a very powerful tool, since you can prepare a script which instructs TextWrangler to perform a whole series of search or search and replace operations.

Consider the scripting command below:

```
tell application "TextWrangler"

find "TextWrangler(.+)$" searching in document of text window 1 ¬
  options { search mode: Grep } with selecting match

end tell
```

In previous versions, the *find* command always operated on the front window. Now, you must explicitly specify the text to be searched, either by specifying an explicit tell target, or by supplying a *searching in* parameter. So the following scripts are equivalent:

```
tell application "TextWrangler"
    find "TextWrangler" searching in document of text window 1
end tell
```

and

```
tell application "TextWrangler"
    tell document of text window 1
        find "TextWrangler"
    end tell
end tell
```

Note that either the tell-target or the *searching in* parameter must resolve to something that contains text. As a shortcut, you can specify a window, and if the window contains text, the search can proceed. You can also specify a text object:

```
find "Search Text" searching in (lines 3 thru 5 of document of
text window 2)
```

Also unlike previous versions of TextWrangler, the defaults for parameters not specified in the *find* command are no longer controlled by the user interface (that is, the Find & Replace dialog).

When performing a *find*, TextWrangler will return a record describing the results of the search. This record contains a Boolean which indicates whether the search was successful, a reference to the text matched by the search, and the text string matched by the search. Given the first example above, the results might look like this (after reformatting for clarity):

```
{found:true,
found object:characters 55 thru 60 of text window 1 of
application "TextWrangler",
found text:"TextWrangler"}
```

Scripting Single Replaces

To do a single find and replace via AppleScript, you can write:

```
tell application "TextWrangler"

set result to (find "TextWrangler" searching in text window 1¬
    with selecting match)

    if (found of result) then
       set text of (found object of result) to "Replacement"
    end if

end tell
```

When performing a grep search, you cannot just replace the matched pattern with a replacement string; the grep subsystem needs to compute the substitutions. The *grep substitution* event is provided for this purpose; given a preceding successful Grep search, it will return the appropriate replacement string. So if you perform a grep search, the script would look like:

```
tell application "TextWrangler"

set result to find "TextWrangler(.+)$" searching in text window 1
¬
    options {search mode:grep}

    if (found of result) then
        set text of (found object of result) to ¬
            grep substitution of "\\1"
    end if

end tell
```

Note that when using a backslash "\" character in AppleScript, it needs to be "escaped" by means of another backslash; thus, in the above example, "\\1" used in the script, will become the grep replacement string "\1" when passed to TextWrangler.

Scripting Multi-File Searches

In TextWrangler, a multi-file search is a simple extension of the *find* scripting command. To search a single file or folder for all occurrences matching the search parameters, specify the file or folder as the *searching in* parameter of the search.

For example, to find all occurrences of "index.html" in a web site, one might use the following scripting command:

```
find "index.html" searching in (alias "Files:WebSite:")
```

Likewise, to find JavaScript line comments:

```
find "//.+$" searching in (alias "Files:WebSite:") ¬
    options {search mode: Grep}
```

To search in a single file:

```
find "crash" searching in (alias "Files:WebSite:index.html")
```

Scripting the Clipboard

TextWrangler has multiple clipboards. These are fully accessible via the scripting interface. Due to operating system constraints, most clipboard operations require TextWrangler to be frontmost.

Here are some examples:

```
count clipboard
```

- Returns the number of clipboards supported by the application

```
clipboard 1
```

- Returns {index:1, contents:"Files:WebSite:", length:14, is multibyte:false, display font:"Monaco", display font size:9, style:{plain}}

```
clipboard 1 as text
```

- Returns "Files:WebSite:"

```
clipboard 1 as reference
```

- Returns clipboard 1 of application "TextWrangler"

```
current clipboard
```

- Returns the current clipboard as a record (you can coerce it to reference or text or get individual properties)

To set the text in a given clipboard to literal text:

```
set contents of clipboard 3 to "foobar"
```

To set the text in a clipboard to text represented by an object specifier:

```
set contents of clipboard 3 to selection of window 2
```

To copy the contents of one clipboard to another:

```
set contents of clipboard 5 to clipboard 3
```

or, to set the current clipboard to the contents of a different clipboard, (thus making it exportable to the system clipboard):

```
set current clipboard to clipboard 3 as text
```

or finally, with even less typing involved:

```
set current clipboard to clipboard 5
```

To make any clipboard the current clipboard, select it:

```
select clipboard 5
```

Scripting Text Factories

You can now apply a text factory to a file via the AppleScript interface. The minimum invocation is:

```
apply text factory <file reference> to <reference>
```

The "to" parameter can be a single reference or a list of references, as for the multi-file "find" or "replace" events.

Optional parameters include "filter", "saving", "recursion", "text files only", "skip shielded folders", "search invisible folders", all with the same meanings as in the multi-file "replace" event.

Common AppleScript Pitfalls

Here are some things to watch out for when scripting TextWrangler with AppleScript.

The Escape Issue

AppleScript uses the backslash character as an escape character. You can use \r to indicate a carriage return or \t to indicate a tab character. More importantly, you can use \" or \' to include a quote mark or apostrophe in a string that is delimited by quotes or apostrophes. If you want to specify a real backslash, you must write \\.

That's not all that confusing until you start writing AppleScripts that call on TextWrangler's powerful grep searching capability. TextWrangler *also* uses the backslash as an escape character. If you want to search for an actual backslash in a document, you have to tell TextWrangler to search for \\. However, if you do that in AppleScript, you must keep in mind that AppleScript will first interpret the backslashes before passing them to TextWrangler. To pass one backslash to TextWrangler from AppleScript, you must write two in AppleScript.

So to tell TextWrangler to search for a single literal backslash from an AppleScript, you must write no fewer than *four* backslashes in the script. Each pair of backslashes is interpreted as a single backslash by AppleScript, which then passes two backslashes to TextWrangler. And TextWrangler interprets those two backslashes as a single one for search purposes. (This proliferation of backslashes can make your scripts look a bit like a blown-over picket fence.)

The Every Item Issue

When writing a script that loops through every item of a TextWrangler object (for example, every line of a document), do not do it like this:

```
repeat with i in every line of text document 1
    -- do stuff here...
end repeat
```

This forces TextWrangler to evaluate "every line of document 1" every time through the loop, which will slow your script significantly. Instead, write

```
set theLines to every line of text document 1
repeat with i in theLines
    -- do stuff here...
end repeat
```

CHAPTER

12

# Unix Scripting and the Command Line

This chapter describes how to use TextWrangler to work with Unix scripting environments and the command line. TextWrangler offers a variety of capabilities to support your development tasks, beginning with syntax coloring and function browsing support for numerous languages, and extending to direct integration with the native Mac OS X Perl and Python environments, as well as other Unix scripting tools such as Ruby or shell scripts. Additionally, you can invoke TextWrangler from the command line via the optional 'edit' and 'twdiff' tools.

In this chapter

Configuring TextWrangler

If the Shebang (#!) menu is not available, you may need to activate TextWrangler's support for Unix scripting tools. To do this, open the Preferences window (by choosing Preferences from the TextWrangler menu), and turn on the Unix Scripting Tools option. Then, quit and relaunch TextWrangler to have this change take effect.
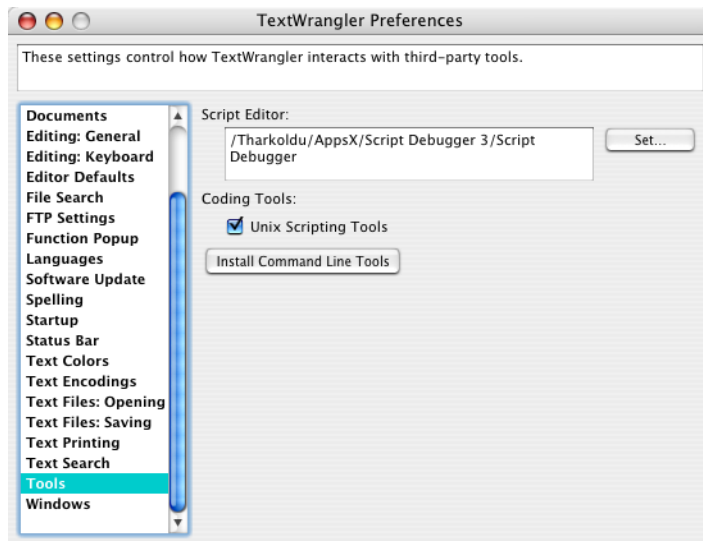
Syntax Coloring

Although it is not essential, you may want to turn on syntax coloring when you use TextWrangler with a development environment. When syntax coloring is on, TextWrangler displays keywords and other language elements in color. You can turn on syntax coloring by setting the Syntax Coloring option in the Editor Defaults preferences panel.

Switching Between Source and Header Files

When you edit a C or C++ source file, you can press Control-Tab to switch to the corresponding header file and vice versa. TextWrangler uses the information in the Languages section of the Preferences window to determine whether a file is a source or header file.

TextWrangler and the Unix Command Line

This section describes how you can use the "edit" and "twdiff" command line tools to invoke TextWrangler from the command line.

The "edit" Command Line Tool

You can use the "edit" command line tool to open files in TextWrangler via the Unix command line. The first time you run TextWrangler after installation, it offers to install the "edit" tool for you. If you choose not to do so, you can use the "Install Command Line Tools" button in the Tools preferences panel to install the tool at a later time.

To open a file in TextWrangler from the command line, type

	edit *filename*

where *filename* is the name of the file to be opened. You may also specify a complete FTP or SFTP URL to a remote file or folder to have TextWrangler open the file, or an FTP/SFTP browser to the folder.

To launch TextWrangler without opening a file (or activate it, if it is already running), type

	edit –l

You can also pipe `stdin` to the "edit" tool, and it will open in a new untitled window in TextWrangler: for example,

	ls –la | edit

If you just type

	edit

with no parameters, the tool will accept `stdin` from the terminal; type Control-D (end-of-file) to terminate and send it to TextWrangler.

The complete command line syntax for the "edit" tool is

	edit [–bchlpuvVw --resume] [–e <encoding_name>]
	  [–t <string>] [+<n>] [file (or) <S/FTP URL> ...]

See the "edit" tool's online man page ("man edit") for a complete description of the available switches and options.

The "twdiff" Command Line Tool

You can use the "twdiff" command line tool to apply TextWrangler's Find Differences command to a pair of files or folders specified on the Unix command line. The first time you run TextWrangler after installation, it offers to install the "twdiff" tool for you. If you choose not to do so, you can use the "Install Command Line Tools" button in the Tools preferences panel to install the tool at a later time.

To invoke the Find Differences command from the command line, type

	twdiff *file1 file2*

or

	twdiff *folder1 folder2*

where *file1* and *file2* are the names of the files, or *folder1* and *folder2* are the names of the folders, to be compared. You can also specify options for how the Find Differences command will be applied, which correspond to those available in the dialog.

The complete command line syntax for the "twdiff" tool is

```
twdiff [ --<options> ] [ FILE1 FILE2 | FOLDER1 FOLDER2 ]
```

See the "twdiff" tool's online man page ("man twdiff") for a complete description of the available switches and options.

Perl, Python, and Shell Scripting

TextWrangler provides robust integration with Perl, Python, and Unix shell scripting languages.

Using Unix Scripts

TextWrangler works directly with the native Perl and Python environments provided with Mac OS X, and supports similar integration with shell scripts and any other Unix scripting language, such as Ruby.

TextWrangler's Unix shell scripting features are accessed via the Shebang menu: "#!". (Why "Shebang"? Because executable Unix scripts traditionally start with the two-character sequence "#!". Some people pronounce these two characters "hash-bang," others say "sharp-bang," but the most common pronunciation is simply "shebang.")

The "shebang line" is the first line of the script, and includes a Unix-style path to the interpreter for the language—for example, "#!/usr/bin/perl", or "#!/usr/local/bin/python".

While TextWrangler does not entirely depend upon the accuracy of the shebang line for Perl and Python files (if you have the correct language mapping set for the file contents), it is a good practice for these, and necessary for any other shell scripts, that the shebang line given must be a correct full path to the executable.

Language Resources

Perl is an acronym for Practical Extraction and Report Language (or alternatively, Pathologically Eclectic Rubbish Lister) and was developed by Larry Wall. If you are interested in learning Perl, the quintessential Perl references are:

Learning Perl (3rd Edition), by Randal L. Schwartz & Tom Phoenix.
O'Reilly and Associates, 2001. ISBN: 0-596-00132-0

Programming Perl (3rd Edition), by Larry Wall, Tom Christiansen, Jon Orwant.
O'Reilly and Associates, 2000. ISBN: 0-596-00027-8

The following are excellent Internet resources for the Macintosh implementation of Perl, and Perl in general:

Perl.com from O'Reilly and Associates
http://www.perl.com/

Perl Mailing Lists
`http://lists.perl.org/`

Picking Up Perl, a freely redistributable Perl tutorial book by Bradley M. Kuhn
`http://www.ebb.org/PickingUpPerl/`

comp.lang.perl.misc, comp.lang.perl.moderated Usenet news groups

Python is a portable, interpreted, object-oriented programming language, originally developed by Guido van Rossum. If you are interested in learning Python, consider the following books:

Learning Python, by Mark Lutz & David Ascher. O'Reilly and Associates, 1999. ISBN: 1-56592-464-9

Programming Python (2nd Edition), by Mark Lutz. O'Reilly and Associates, 2001. ISBN: 0-596-00085-5

Internet resources for Python:

Python home page
`http://www.python.org`

Python Cookbook
`http://aspn.activestate.com/ASPN/Cookbook/Python`

comp.lang.python Usenet news group

Line Endings and Unix Scripts

To execute scripts, the script interpreter for any given language requires source code to be encoded with native line endings, i.e. Unix line breaks for Perl and other shell scripting languages. TextWrangler will automatically make sure that the script source sent to each interpreter matches its line ending expectations.

Although for best compatibility, you should set the line endings of your scripts appropriately, regardless of the line endings of the file on disk, TextWrangler will always send scripts to Perl, Python, and other script interpreters with Unix line endings.

Configuring Perl

TextWrangler can make full use of the standard Mac OS X Perl install with no need for further configuration. However, if you wish to install and work with multiple versions of Perl, you will need to specify the appropriate version in your scripts' shebang lines.

Configuring Python

TextWrangler expects to find Python in /usr/bin, /usr/local/bin, or /sw/bin. If you have installed Python elsewhere, you must create a symbolic link in /usr/local/bin pointing to your copy of Python in order to use pydoc and the Python debugger.

The standard Mac OS X install of Python does not include keyword documentation. In order to employ the Find in Reference command with Python scripts, you must obtain and install this documentation as follows.

- Download the documentation files from the python.org web site:
  `http://www.python.org/doc/2.3/`

- Extract the documentation files, and place them in some suitable location, e.g.
  ~/Library/Python-Docs

- Edit your "environment.plist" file, and create an environment variable
  PYTHONDOCS to the location of the folder which contains the Python
  documentation.

Note   For information on setting environment variables for TextWrangler and other GUI
applications, refer to Apple Technical Q&A 1067. <http://developer.apple.com/qa/qa2001/
qa1067.html>

Shebang Menu

The commands in this menu are also available in a floating tool palette, named "Unix
Scripting Tools", which is accessible via the Palettes submenu of the Window menu.

Check Syntax

Checks the syntax for the frontmost window. Errors are displayed in a standard
TextWrangler error browser (see Chapter 9, "Browsers," for more details on working
with error browsers). This command is only available for Perl and Python scripts.

Run

Runs the script in the frontmost window by default. Any output from the script is
displayed in a new TextWrangler window. The output window is titled "Unix Script
Output", and the file is created in the Unix Support folder in TextWrangler's
application support folder. By default, errors for Perl and Python scripts are displayed
in an error browser; errors for other languages are displayed as text in the output
window.

Hold down the Option key while choosing Run to display the Run a Script dialog,
which allows you to set options that will be used when the command is executed.
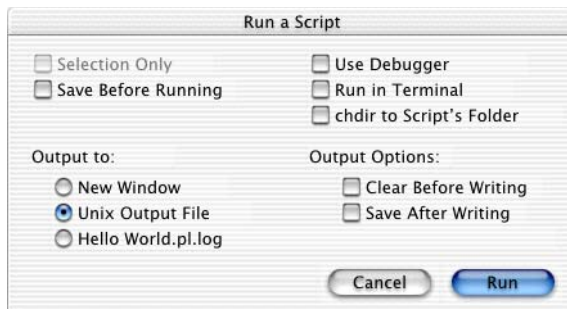
Selection Only: Check this box to execute only the selected text in the frontmost
document window.

Save Before Running: Check this box to save the source file before running the script.

Output to: Choose to display output in a new window, to direct it to the Unix Output
file, or to write it to a file in TextWrangler's Logs folder (~/Library/Logs/
TextWrangler/).

Use Debugger: Check this box to run Perl or Python scripts in the interpreter's debugger.

Run in Terminal: This command runs the script in a new Terminal window.

Chdir to Script's Folder: Check this box to set the working directory to the folder that contains the script before running it.

Output Options: Mark these checkboxes to clear the output file before writing and to save it after writing, respectively.

Run in Terminal

This command will run the script in a new Terminal window, regardless of the settings in the Run Perl Script dialog.

Run in Debugger

Runs the script in the interpreter's debugger, regardless of whether the Use Debugger option is set for the Run command; also, any output options set in the Run command will be ignored. The Run in Debugger command is only available for Perl and Python.

Run File

Runs a script from an arbitrary file rather than from a TextWrangler window. The Run a Script File dialog appears. You can select a file by clicking the File button or by dragging a file to the path box at the top of the dialog from the Finder. The options are the same as the ones described above for the Run a Script dialog.

Find in Reference

Looks up the selected text using the appropriate reference application (perldoc for Perl, pydoc for Python). If there is no selection, a dialog will open in which you can enter a search string. The Find in Reference command is not available for languages other than Perl and Python.

View POD/Python Documentation

When the frontmost document is a Perl file and you invoke this command, TextWrangler will process the document contents using by the command line pod2text tool and display the result in a new text window.

Note   POD stands for Plain Old Documentation, and is the standard Perl documentation format.

When the frontmost document is a Python file, and you invoke this command, TextWrangler will display the module documentation.

Filters and Scripts

Before you begin using Unix scripts with TextWrangler, you should locate and familiarize yourself with the Unix Support folder, both of which reside in the TextWrangler folder. (See Chapter 2 for details about the TextWrangler folder.) Inside these folders are subfolders for storing Filters, Scripts, and other files used for Unix shell scripting integration.

The contents of these two folders will be used to build the Scripts list and Filters list, two floating palettes that allow you to run your scripts or filters with a double-click. Scripts and filters placed in these folders will also appear in their respective submenus at the bottom of the Shebang menu.

Filters

Filters operate on the selected text of the frontmost window. The current selection is passed as input to the filter, and any output generated by the filter overwrites the selection. In other words, filters act like plug-ins for text manipulation.

There are two ways to run filters: through the Filters palette or the Filters submenu at the bottom of the Shebang menu. To open the Filters palette, select it from the Palettes submenu in the Window menu. You can run a filter by selecting it from the list and clicking the Run button, or you can simply double-click the filter name in the list.

Hold down the Option key while double-clicking a filter or selecting it from the menu to open the file for editing instead of running it. You can also hold down the Shift key while selecting a filter item from the menu to reveal the file in the Finder, or you can select a folder node from the menu to open that folder in the Finder.

Optionally, filter output can be sent to a different window, instead of overwriting the selection—hold down the Command key while selecting a filter from the Filters list palette, or from the Filters submenu, to open the Filter Options dialog. Changes made in the Filter Options dialog affect all filters, and remain in effect until you make changes in the Filter Options dialog again.

Using Filters with Multi-Byte Text

Filters are now two-byte savvy. If the front window is Unicode, the selection is written out as big-endian Unicode with a byte-order mark. Your filter is responsible for handling the input correctly. If the front window contains 8-bit text, the input is written out as usual.

Scripts

Scripts are similar to filters, but do not operate on the text of the frontmost window. Like filters, you can run scripts from either a submenu at the bottom of the Shebang menu, or from the Scripts list palette. The same options as for filters apply when running scripts—hold down the Command key while double-clicking a script in the list or selecting it from the menu to open the Run Options dialog; hold down the Option key while double-clicking on a script or selecting it from the menu to open the file for editing instead of running it; hold down the Shift key while selecting it from the menu to reveal the file in the Finder, or while selecting a folder node, to reveal that nod in the Finder.

Additional Notes

In addition to the features detailed above, there are some additional considerations about TextWrangler's Perl integration which it may help you to be aware of.

Setting Menu Keys for Filters and Scripts

The Filters and Scripts lists both have a "Set Key" button at the top of their palettes. Select a filter or script from the list and click this button to set a keyboard shortcut for the selected item.

Manually Sorting the Filter and Script Lists

By default, items in the Perl Filters List are displayed in alphabetical order. However, you can force them to appear in any desired order by including any two characters followed by a right parenthesis at the beginning of their name. (For example "00)Foo" would sort before "01)Bar.") For such files, the first three characters are not displayed in TextWrangler. You can also insert a divider by including an empty folder whose name ends with the string "-***". (The folder can be named anything, so it sorts where you want it.) These conventions are the same as those used by the utilities FinderPop and OtherMenu.

Canceling Perl Operations

You can press the Cancel button in the progress dialog to cancel a task directly from within TextWrangler. Since TextWrangler must kill the spawned Perl (or Python, or shell) process with a SIGINT, any unflushed data in open filehandles (including STDOUT and STDERR) will be lost unless the script takes measures to prevent this.

Using Plug-Ins

Plug-ins are code modules that enhance TextWrangler's features, and which appear within TextWrangler itself in the Tools menu. These modules are also called BBEdit plug-ins because their format was originally defined for BBEdit. This chapter describes the basic plug-ins that ship with TextWrangler. Other developers have also written many BBEdit plug-ins, which are available from various web sites (including our own). TextWrangler can use any plug-ins that were created for BBEdit 7.0 and later

This chapter describes the basic procedures for installing and using language modules and plug-ins, and provides references to information about producing such items.

In this chapter

Installing Language Modules and Plug-Ins

To install a language module, move or copy the module file into the Language Modules folder of your TextWrangler application support folder.

To install a plug-in, drag and drop it directly onto the TextWrangler application icon in the Finder. TextWrangler will launch, if necessary, and present an alert asking you to confirm that you want to install the item. If there is already a plug-in with the same name in your TextWrangler application support folder, you will be asked whether to replace that item with the version you are dragging. If you confirm this operation, the plug-in you dragged will be placed at the top level of the Plug-Ins subfolder.

Note   When you install a plug-in by drag and drop, if there is no local TextWrangler application support folder available, one will be created. (See "TextWrangler's Application Support Folders" on page 25.)

After installing a new language module or plug-in, you will need to quit and relaunch TextWrangler in order to use it.

To remove an installed language module or plug-in, you must remove the item's file from the appropriate subfolder of your TextWrangler application support folder, then quit and relaunch TextWrangler.

Using Language Modules

Language modules are add-on items which provide syntax coloring and function browsing for programming languages that TextWrangler does not natively support.

There are two types of language modules: coded, and codeless. Coded language modules must be prepared according to the requirements of TextWrangler's plug-in module interface. (See "Developer Information" on page 231.) Codeless language modules are text documents prepared in a specific plist format. (See below.)

After you install a language module and relaunch TextWrangler, syntax coloring and function browsing will be available for the language(s) supported by that module. To verify that a language module is active, or to modify or add file suffix mappings for the language(s) it provides, use the Languages preferences panel (see page 178).

Codeless Language Modules

A codeless language module is a specially-formatted text file which allows you to describe the properties of a source code language via a set of basic parameters. TextWrangler will then use these parameters to perform syntax coloring and function navigation for the specified language.

Codeless language modules are written as "property lists" (or "plists"), which is an XML format that Mac OS X uses for many purposes. You can create or edit codeless language module files with TextWrangler itself, with the Mac OS X Property List Editor (located in /Developer/Applications/Utilities/if you have installed the Apple Developer Tools package), or with a third-party editor such as PlistEdit Pro.

```
http://homepage.mac.com/bwebster/plisteditpro.html
```

You can find complete specifications for creating codeless language modules in Appendix C (see page 245), or in the Developer Information section of our web site.

```
http://www.barebones.com/support/develop/index.shtml
```

Language Module Compatibility

IMPORTANT   You will **not** be able to use any third-party language modules which do not support Unicode text. If TextWrangler encounters such a module, it will not load that module, and will log a message to the system console.

Contact the developers of such a module, or visit the Bare Bones Software web site (see above) for more information on the availability of updated modules.

Overriding Existing Modules

Language modules can override existing language definitions, including the built-in definitions . If there is more than one module present which supports a given language, TextWrangler will use the module with the most recent modification date.

Using Plug-Ins

The commands made available by all installed plug-ins appear in the Tools menu. To use a plug-in, choose it from the menu. Some plug-ins may require that there be an active text window, or an active text selection, in order to function. The menu entries for such tools may be dimmed when this condition is not met.

The Plug-In Info command in the TextWrangler menu displays a window listing all installed plug-ins and their version numbers. The Help and Web Site buttons at the bottom of the window are enabled when there is online help or a Web page available, respectively, about the selected plug-in.

The Tools Menu and Palette

The Tools palette can be displayed by choosing Plug-In Tools from the Palettes submenu of the Window menu. Any plug-ins you have installed will appear both in this Tools palette and in the Tools menu itself. Names that are too long to fit within the width of the window are truncated with ellipses (…).

"Hovering" the mouse over such a truncated name displays a tool tip showing the full name. If you hold down the Option key, the tool tip will appear instantly, with no hovering delay. Names that fit entirely within the window without truncation do not display a tool tip.

Any plug-ins you place directly in the Plug-Ins subfolder of the TextWrangler application support folder will appear as individual items in the Tools menu. If you place them in subfolders within the Plug-Ins folder, they will appear in submenus of the Tools menu that mirror this subfolder structure. In the Tools window, such subfolders will appear as separate sublists; plug-ins located at the top level of the Plug-Ins folder will appear in the sublist named Plug-Ins.

No Plug-Ins Available

If TextWrangler does not load any plug-ins at launch, either because there were none present, or because there were no compatible items available, the Tools menu will contain a single item: No Plug-Ins Available. If you choose this item, TextWrangler will display a dialog explaining why no other items are present on the Tools menu.

Setting Key Equivalents for Plug-Ins

The Set Key button in the TextWrangler Tools palette lets you assign key equivalents to a plug-in. You can use any combination of the Command, Shift, Option, and Control keys in the key equivalents.

Assigning a Key to a Plug-in

To assign a key to a TextWrangler plug-in:

1 Select the tool you wish to assign a key equivalent to in the Tools palette.

2 Click the Set Key button to display the Set Key dialog.

3 Type the key equivalent.

You can use any key combined with Command plus Shift, Option, or Control modifiers if desired. The equivalent must use at least the Command or the Control modifier key to be valid. You can also use Function keys, with or without additional modifiers.

4 Click Save.

Warning   If you try to assign a key sequence that is already used elsewhere, TextWrangler will warn you that there is a conflict and ask you whether you want to reassign that key sequence to the new item.

Removing a Plug-in's Key Equivalent

To remove the key equivalent from a TextWrangler plug-in:

1 Choose the Tool from the Tools palette.

2 Click Set Key.

3 TextWrangler opens the Set Key dialog.

4 Click Reset.

TextWrangler removes the key assignment from the plug-in.

Supplied Plug-Ins

The text processing plug-ins supplied with previous versions of TextWrangler have been superseded by equivalent commands on the Text menu. (See "Text Menu Commands" on page 91.)

IMPORTANT   You should *not* copy any of the factory-supplied plug-ins from previous versions of TextWrangler to use with TextWrangler 2. To determine whether a plug-in was factory-supplied, select it in the Finder, choose the Get Info command, and check its version information.

Third-Party Plug-Ins

A wide variety of plug-ins are available from third parties. An extensive though not exhaustive listing is available in the support section of the Bare Bones Software web site:

```
http://www.barebones.com/support/plugins.html
```

Plug-In Compatibility

IMPORTANT You will **not** be able to use any third-party plug-ins which have not been specifically updated for Mac OS X compatibility, or which do not support Unicode text. If TextWrangler encounters such a plug-in, it will not load that plug-in, and will log a message to the system console.

Contact the developers of your plug-ins or visit the Bare Bones Software web site (see above) for more information on the availability of updated plug-ins.

Developer Information

In addition to a selection of third-party plug-ins, the Bare Bones Software web site contains additional resources for developers who may wish to:

- provide an "Edit in BBEdit/Edit in TextWrangler" command in their application;

- extend the capabilities of BBEdit, TextWrangler, and/or Mailsmith with a plug-in;

- implement a language module to support syntax coloring and/or function navigation for a source code language not presently supported by BBEdit or TextWrangler

The developer information overview page is:

```
http://www.barebones.com/support/develop/index.shtml
```

Command Reference

This appendix provides a quick reference for key assignments and a comprehensive list of the commands that are available from TextWrangler's user interface.

In this appendix

Keyboard Shortcuts for Commands

Many of TextWrangler's commands have pre-defined keyboard shortcuts. TextWrangler also lets you reassign the shortcuts for any menu command or plug-in to suit your own way of working.

To change the keyboard shortcut for any menu command, you can use the Set Menu Keys command. (See "Assigning Keys to Menu Commands" on the following page.)

Many other TextWrangler features can have keyboard shortcuts assigned as well. Here's how to set them:

Feature	Set Keys in…
Menu commands	Set Menu Keys (TextWrangler menu)
Plug-ins	Plug-In Tools palette
Stationery	Stationery palette

To display any of TextWrangler's floating palette windows, use the Palettes submenu in the Window menu.

Assigning Keys to Menu Commands

You can assign your own keyboard shortcuts (key equivalents) to any of TextWrangler's menu commands by choosing Set Menu Keys from the TextWrangler (Application) menu. The Set Menu Keys dialog, below, appears.
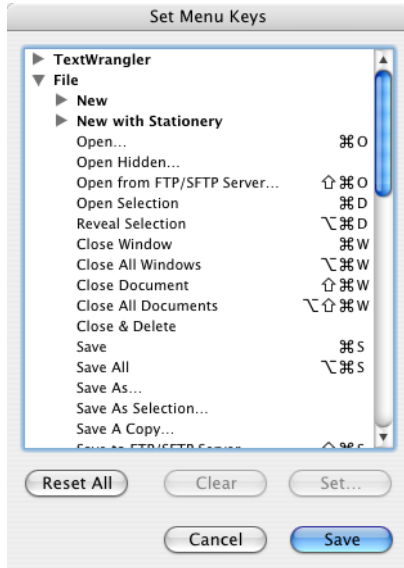
To set the key equivalent for a menu command, find it in the Set Menu Keys dialog, click the Set button, and type the desired keystroke.

To remove the key equivalent from the selected menu command, click the Clear button.

Click the Reset All button to restore all key equivalents to their default values (as listed in this Appendix).

Available Key Combinations

All menu key combinations must include either the Command key or the Control key (or both), except function keys, which may be used unmodified.

In addition to the letter, number, and punctuation keys, you can use the Help, Home, End, Page Up and Page Down keys in menu key combinations as well. The Help key can be assigned without modifiers; the others must be used in combination with at least either the Command or Control key.

Note   The key combinations Command-0 through Command-9 are reserved for file entries on the Window menu. There may also be other key combination which the operating system preempts, such as Command-Tab.

Listing by Menu and Command Name

TextWrangler Menu

About TextWrangler
Plug-In Info
Preferences   Cmd-,
Set Menu Keys…
Services   *(submenu)*
Hide TextWrangler   (*none or* Cmd-H)
Hide Others
Show All
Quit TextWrangler   Cmd-Q

File

New   *(see next column)*
New With Stationery   *(submenu)*
Open…   Cmd-O
Open Hidden…
Open from FTP Server…   Cmd-Shift-O
Open File by Name…   Cmd-D
Reveal Selection   Cmd-Opt-D
Open Recent   *(submenu)*
Reopen Using Encoding   *(submenu)*
Close Window   Cmd-W
Close All Windows   Cmd-Opt-W
Close Document   Cmd-Shift-W
Close All Documents   Cmd-Opt-Shift-W
Close & Delete   Cmd-Shift-W
Save   Cmd-S
Save All   Cmd-Opt-S
Save As…
Save As Selection…   (Opt)
Save a Copy…
Save to FTP/SFTP Server…   Cmd-Shift-S
Save a Copy to FTP/SFTP   Cmd-Opt-Shift-S
Server…
Revert
Hex Dump File…
Hex Dump Front Window…
Backup Options…
Make Backup Now…
Page Setup…
Print…   Cmd-P
Print All   Cmd-Opt-P
Print One Copy   Cmd-Shift-P

File --> New

Text Document   Cmd-N
   (with selection)
   (with Clipboard)
Text Window   Cmd-Shift-N
Disk Browser   Cmd-Opt-N
FTP/SFTP Browser

Edit

Undo   Cmd-Z
Redo   Cmd-Shift-Z
Clear Undo History   Ctl-Z
Cut   Cmd-X
Cut & Append   Cmd-Shift-X
Copy   Cmd-C
Copy & Append   Cmd-Shift-C
Paste   Cmd-V
Paste Previous Clipboard   Cmd-Shift-V
Paste Column
Clear
Select All   Cmd-A
Select None   Cmd-Shift-A
Select Line   Cmd-L
Select Paragraph   Cmd-Opt-L
Insert   *(see below)*
Show Clipboard
Previous Clipboard   Ctl-[
Next Clipboard   Ctl-]
Text Options…   Cmd-Opt-;
Printing Options…   Cmd-Shift-;
Special Characters…

Edit --> Insert

File Contents…
File Path…
Folder Listing…
Folder Path…
Page Break

Tools

(Installed plug-ins)   (none by default)

Text

Show Fonts	
Balance	Cmd-B
Exchange Characters	Cmd-`
Exchange Words	Cmd-Opt-`
Change Case…	
Change Case	(Opt)
Shift Left	Cmd-[
Shift Left One Space	Cmd-Shift-[
Shift Right	Cmd-]
Shift Right One Space	Cmd-Shift-]
Un/Comment Selection	
Hard Wrap…	Cmd-\
Hard Wrap	Cmd-Opt-\
Add Line Breaks	
Remove Line Breaks	
Apply Text Factory…	
Educate Quotes	
Straighten Quotes	
Add/Remove Line Numbers...	
Prefix/Suffix Lines…	
Sort Lines…	
Process Duplicate Lines…	
Process Lines Containing…	
Rewrap Quoted Text…	Cmd-'
Rewrap Quoted Text	Cmd-Opt-'
Increase Quote Level	
Decrease Quote Level	
Strip Quotes	
Zap Gremlins…	
Zap Gremlins	(Opt)
Entab…	
Entab	(Opt)
Detab…	
Detab	(Opt)
Normalize Line Endings	
Find Next Misspelled Word	Cmd-;
Find All Misspelled Words	Cmd-Opt-;
Show Spelling Panel	Cmd-Shift-;

View

Show/Hide Status Bar	
Show/Hide Navigation Bar	
Show/Hide Documents Drawer	
Previous Document	Cmd-Opt-[
Next Document	Cmd-Opt-]
Open in Separate Window	Cmd-Opt-O
Get Info	Cmd-I
Reveal in Finder	
Open in Super Get Info	

Search

Find…	Cmd-F
Quick Search	Cmd-Opt-F
Find Again	Cmd-G
Find Again (reverse)	Cmd-Shift-G
Find Selection	(Cmd-H *or none*)
Find Selection (reverse)	Cmd-Shift-H
Enter Search String	Cmd-E
Enter Search Pattern	Cmd-Shift-E
Enter Replace String	Cmd-Opt-E
Enter Replace Pattern	Cmd-Opt-Shift-E
Replace	Cmd-=
Replace All	Cmd-Opt-=
Replace & Find Again	Cmd-T
Go to Line…	Cmd-J
Go to Line	Cmd-Opt-J
Go to Center Line	Cmd-Shift-J
Find Differences…	
Compare Two Front Documents	
Compare Against Disk File	
Apply to New	Cmd–left arrow
Apply to Old	Cmd–right arrow
Compare Again	
Find in Reference…	Cmd-Shift--

Tools *(Installed plug-ins)*   (none by default)

Window

Minimize Window	
Minimize All Windows	(Opt)
Bring All to Front	
Palettes	(*see next page*)
Arrange…	
Arrange	(Opt)

Get Info		Start/Stop Recording	
Super Get Info *(if available)*	(Opt)	*(Installed scripts)*	
Reveal in Finder			
Send To Back	Cmd-Ctl-/		

Get Info
Super Get Info *(if available)*   (Opt)
Reveal in Finder
Send To Back   Cmd-Ctl-/
Exchange With Next
Synchro Scrolling
(Open text windows)   Cmd-1 *to* Cmd-0

Start/Stop Recording
(Installed scripts)

Text Factories (gear)

Open Text Factories Folder
(Installed text factories)

Window --> Palettes

ASCII Table
Plug-In Tools
Scripts
Stationery
Text Factories
Windows
Unix Tools
Unix Filters
Unix Scripts

Shebang (#!)

Check Syntax
Check Selection Syntax   (Opt)
Run
Run…   (Opt)
Run in Terminal
Run in Debugger
Run File…
Find in Reference…
View POD/Python
Documentation
Unix Filters   *(submenu)*
Unix Scripts   *(submenu)*

Bring Up To Date   Cmd-U
Make
Debug   Cmd-R
Run   Cmd-Opt-R

Scripts (script icon)

Open Script Editor
Open Scripting Dictionary
Open Scripts Folder

Listing by Default Key Equivalents

Key	Command
Cmd-1 *to* Cmd-0	Window: *(Open windows)*
Cmd-A	Edit: Select All
Cmd-B	Text: Balance
Cmd-C	Edit: Copy
Cmd-D	File: Open Selection File: Open File by Name
Cmd-E	Search: Enter Search String
Cmd-F	Search: Find…
Cmd-G	Search: Find Again
Cmd-H	TextWrangler: Hide TextWrangler
Cmd-I	View: Get Info
Cmd-J	Search: Go to Line…
Cmd-L	Edit: Select Line
Cmd-N	File: New: Text Document
Cmd-O	File: Open…
Cmd-P	File: Print…
Cmd-Q	TextWrangler: Quit TextWrangler
Cmd-S	File: Save
Cmd-T	Search: Replace & Find Again
Cmd-V	Edit: Paste
Cmd-W	File: Close
Cmd-X	Edit: Cut
Cmd-Z	Edit: Undo
Cmd-'	Text: Rewrap Quoted Text…
Cmd--	Search: Find in Reference…
Cmd-;	TextWrangler: Preferences
Cmd-[	Text: Shift Left
Cmd-]	Text: Shift Right
Cmd-`	Text: Exchange Characters
Cmd-/	Zoom Window
Cmd-=	Search: Replace

Key	Command
Cmd-\	Text: Hard Wrap…
Cmd–left arrow	Search: Apply to New
Cmd–right arrow	Search: Apply to Old
Cmd-Ctl-/	Window: Send To Back
Cmd-Opt-D	File: Reveal Selection
Cmd-Opt-E	Search: Enter Replace String
Cmd-Opt-F	Search: Quick Search
Cmd-Opt-J	Search: Go to Line
Cmd-Opt-L	Edit: Select Paragraph
Cmd-Opt-N	File: New: Disk Browser
Cmd-Opt-O	View: Open in Separate Window
Cmd-Opt-P	File: Print All
Cmd-Opt-S	File: Save All
Cmd-Opt-W	File: Close All Windows
Cmd-Opt-'	Text: Rewrap Quoted Text
Cmd-Opt-;	Edit: Text Options…
Cmd-Opt-`	Text: Exchange Words
Cmd-Opt-[	Text: Decrease Quote Level
Cmd-Opt-]	Text: Increase Quote Level
Cmd-Opt-/	Zoom Window Full Screen
Cmd-Opt-=	Search: Replace All
Cmd-Opt-\	Text: Hard Wrap
Cmd-Opt–down arrow	Search: Go to Next Error
Cmd-Opt–up arrow	Search: Go to Previous Error
Cmd-Opt-Shift-N	File: New: (with Clipboard)
Cmd-Opt-Shift-S	File: Save a Copy to FTP Server…
Cmd-Opt-Shift-W	File: Close All Documents
Cmd-Shift-A	Edit: Select None
Cmd-Shift-C	Edit: Copy & Append

Key	Command
Cmd-Shift-G	Search: Find Again (reverse)
Cmd-Shift-H	Search: Find Selection (reverse)
Cmd-Shift-J	Search: Go to Center Line
Cmd-Shift-N	File: New: (with selection)
Cmd-Shift-O	File: Open from FTP Server…
Cmd-Shift-P	File: Print One Copy
Cmd-Shift-S	File: Save to FTP Server…
Cmd-Shift-V	Edit: Paste Previous Clipboard
Cmd-Shift-W	File: Close & Delete
Cmd-Shift-X	Edit: Cut & Append
Cmd-Shift-Z	Edit: Redo
Cmd-Shift-;	Edit: Printing Options…
Cmd-Shift-[	Text: Shift Left One Space
Cmd-Shift-]	Text: Shift Right One Space
Ctl-Tab	Switch to Header/Source File
Ctl-[	Edit: Previous Clipboard
Ctl-]	Edit: Next Clipboard

B

Editing Shortcuts

In TextWrangler you can perform many editing functions (including word selection or deletion) directly from the keyboard. Chapter 4 contains complete details on TextWrangler's text editing features. This appendix provides a quick reference to available keyboard and mouse shortcuts for word selection and deletion.

In this appendix

Mouse Commands

	No Modifier	**Shift**
Click	move insertion point	extend selection
Double-click	select word	extend selection to word
Triple-click	select line	

Triple-clicking is the same as clicking in a line and then choosing the Select Line command from the Edit menu.

Holding down the Command or Option keys as you click or double-click triggers special actions:

	Option	**Command**	**Command/ Option**
Click	–none–	Open URL	–none–
Double-click	look up selected word in programming reference	–none–	find next instance of the selected text

Arrow and Delete Keys

You can use the arrow keys to move the insertion point right, left, up, and down. You can augment these with the Command and Option keys to move by word, line, or screens, or with the Shift key to create or extend selections. For example, pressing Shift-Option-Right Arrow selects the word to the right of the insertion point.

You can hold down the Control key while using the arrow keys to scroll through editing windows without moving the position of the insertion point.

Key	Modifier	Action
(left/right) Arrow		Move 1 character left/right
(left/right) Arrow	Option	Move 1 word left/right
(left/right) Arrow	Command	Move to beginning/end of line
(left/right) Arrow	Control	Scroll view left/right
(up/down) Arrow		Move up/down 1 line in file
(up/down) Arrow	Command	Move to top/bottom of file
(up/down) Arrow	Option	Move to previous/next screen page
(up/down) Arrow	Control	Scroll view up/down
[any of the above]	Shift	Make or extend a selection range
Delete		Deletes selection range, or character preceding (to the left of) the insertion point.
Delete	Command	Deletes all characters backwards to beginning of line
Delete	Option	Deletes all characters back to beginning of word
Delete	Shift	(same as Forward Delete)
Forward Delete		Deletes selection range, or character after (to the right of) the insertion point
Forward Delete	Command	Deletes all characters forward to end of the current line
Forward Delete	Option	Deletes all characters forward to end of word
Forward Delete	Shift	(same as Forward Delete alone)

Note   The meaning of the Command and Option modifiers listed above may be exchanged, depending on which settings you have selected for Exchange Command and Option Key Behavior in the Text Editing panel of the Preferences window.

Emacs Key Bindings

The Text Editing panel of the Preferences window contains a checkbox labeled Use Emacs Key Bindings. When this option is turned on, TextWrangler will enable the following Emacs-style keyboard navigation commands. The Escape key is specified in lieu of the Emacs "Meta" key; to use these key equivalents, press and release the Escape key followed by the specified letter key—for example, to type "Esc-V" press and release the Escape key and then type the letter V.

Key Sequence	Action
Ctl-A	beginning-of-line (Move insertion point to start of current line)
Ctl-B	backward-char (Move insertion point backward 1 place)
Ctl-D	delete-char (Delete forward 1 character)
Ctl-E	end-of-line (Move insertion point to end of current line)
Ctl-F	forward-char (Move insertion point forward 1 place)
Ctl-G	keyboard-quit (cancel pending arguments)
Ctl-K	kill-line (Delete to end of current line)
Ctl-L	recenter (Scrolls the current view so the selection is centered on screen)
Ctl-N	next-line (Move insertion point down one line)
Ctl-O	open-line (Inserts line break without moving insertion point)
Ctl-P	previous-line (Move insertion point to start of line above current)
Ctl-R	isearch-backward (Quick Search with the Backwards option)
Ctl-S	isearch-forward (Quick Search)
Ctl-T	transpose-chars (Exchange Characters)
Ctl-U	universal-argument (*See note below*)
Ctl-V	scroll-up (Page down)
Ctl-W	kill-region (Cut)
Ctl-Y	yank (Paste)
Ctl-_	undo (Undo)

Key Sequence	Action
Ctl-X Ctl-C	save-buffers-kill-emacs (Quit)
Ctl-X Ctl-F	find-file (Open file)
Ctl-X Ctl-S	save-buffer (Save current document)
Ctl-X Ctl-W	write-file (Save As)
Esc-<	beginning-of-buffer (Move insertion point to start of document)
Esc->	end-of-buffer (Move insertion point to end of document)
Esc-Q	fill-paragraph (Hard Wrap with current settings)
Esc-V	scroll-down (Page up)
Esc-W	copy-region-as-kill (Copy)
Esc-Y	yank-pop (Paste Previous Clipboard)

Using universal-argument

The universal-argument command (Ctl-U) does not work quite the same way as it does in Emacs. In TextWrangler, it is a simple repeat-count. For example, if you type Ctl-U, then a 3, and then Ctl-N, the insertion point will move down three lines. There is no visual feedback as you type the number, and no way to backspace or otherwise edit the number. If you make a mistake, the best you can do is type Ctl-G (keyboard-quit) and start over.

Codeless Language Modules

This appendix lists the syntax elements available for use in codeless language modules. For further details and example modules, visit the Developer and Plug-In Library sections of our web site.

`http://www.barebones.com/support/develop/index.shtml`

`http://www.barebones.com/support/bbedit/plugin_library.shtml`

Creating a Module

Codeless language modules are written as "property lists" (or "plists"), which is an XML format that Mac OS X uses for many purposes.

`http://developer.apple.com/documentation/Cocoa/`
`Conceptual/PropertyLists/Concepts/XMLPListsConcept.html`

You can create or edit codeless language module files with TextWrangler itself, with the Mac OS X Property List Editor, or with a third-party editor such as PlistEdit Pro.

`http://homepage.mac.com/bwebster/plisteditpro.html`

Note The Property List Editor labels boolean properties as "yes" or "no". However, the actual plisttext file must contain values of either "true" or "false" (written as "<true/>" and "<false/>").

Required Elements

At a minimum, your codeless language module file must include the appropriate XML header declaration, as well as key/value specifications for each of "BBEditDocumentType", "BBLMLanguageCode", and "BBLMLanguageSuffix" in order for BBEdit to load it. The module may then specify any other parameters you desire, including whether to color syntax elements, color a set of keywords, honor case sensitivity, and more. You should save the file with a ".plist" filename extension. If a module fails to load, TextWrangler will write some diagnostic information to the system console.

Starting from a Template

The easiest way to begin creating a codeless language module is to start from a template, or an existing module. The template provided on the following page contains all required key/value pairs, plus a selection of additional parameters which you can fill out or remove as desired.

CodelessLanguageModuleTemplate.plist

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE plist PUBLIC "-//Apple Computer//DTD PLIST 1.0//EN"
"http://www.apple.com/DTDs/PropertyList-1.0.dtd">
<plist version="1.0">
<dict>
    <key>BBEditDocumentType</key>
    <string>CodelessLanguageModule</string>
    <key>BBLMColorsSyntax</key> <true/>
    <key>BBLMIsCaseSensitive</key> <true/>
    <key>BBLMKeywordList</key>
    <array>
        <string></string>
    </array>
    <key>BBLMLanguageCode</key>
    <string>????</string>
    <key>BBLMLanguageDisplayName</key>
    <string></string>
    <key>BBLMScansFunctions</key> <true/>
    <key>BBLMSuffixMap</key>
    <array>
        <dict> <key>BBLMLanguageSuffix</key>
               <string></string>
         </dict>
    </array>
    <key>Language Features</key>
    <dict>
        <key>Close Block Comments</key>
        <string></string>
        <key>Close Parameter Lists</key>
        <string></string>
        <key>Close Statement Blocks</key>
        <string></string>
        <key>Close Strings 1</key>
        <string></string>
        <key>Close Strings 2</key>
        <string></string>
        <key>End-of-line Ends Strings 1</key> <true/>
        <key>End-of-line Ends Strings 2</key> <true/>
        <key>Escape Char in Strings 1</key>
        <string></string>
        <key>Escape Char in Strings 2</key>
        <string></string>
        <key>Identifier and Keyword Characters</key>
        <string></string>
        <key>Open Block Comments</key>
        <string></string>
        <key>Open Line Comments</key>
        <string></string>
        <key>Open Parameter Lists</key>
        <string></string>
        <key>Open Statement Blocks</key>
        <string></string>
        <key>Open Strings 1</key>
        <string></string>
        <key>Open Strings 2</key>
        <string></string>
        <key>Prefix for Functions</key>
        <string></string>
        <key>Prefix for Procedures</key>
        <string></string>
        <key>Terminator for Prototypes 1</key>
        <string></string>
        <key>Terminator for Prototypes 2</key>
        <string></string>
    </dict>
</dict>
</plist>
```

Language Keys and Properties

Key	Value Type
BBEditDocumentType	**String** ("CodelessLanguageModule")

This key/value pair must be present in the property list for the rest of the plist to be examined and loaded, since the file containing the plist need not have any specific file-type or filename-extension.

BBLMLanguageDisplayName	**String**

This is the name displayed for the language module in popup menus and preference panels. Be descriptive, but terse.

BBLMLanguageCode	**String (4 characters maximum)**

This string value should be a unique four-character code for the language that the module supports. Note that the value must be unique with respect to BBEdit's built-in languages and with respect to any installed language modules.

Unfortunately, there is no easy way to identify these potential conflicts beforehand, but just keep this all in mind if the contents of a file ends up looking as though it is being treated as some other language than intended.

BBLMColorsSyntax	**Boolean**

This must have the value 'true' for strings and comments to be colored specially by the language module. Keywords will also be colored if the value is 'true', but only if a list of keywords is also supplied in a BBLMKeywordList array (see below).

BBLMScansFunctions	**Boolean**

This must have the value 'true' for the text to be scanned to locate "function definitions" and for a popup menu of function names to be built that allows for quick navigation to those functions. This requires the 'Identifier and Keyword Characters' string described below to be properly specified.

BBLMIsCaseSensitive	**Boolean**

If this has the value 'false', letters in keywords and other strings are matched against the text without regard to whether they are both upper or lower case. The value 'true' means that an 'x', for example, will only match another 'x' and not an 'X'.

BBLMKeywordList	**Array of String**

Whenever a string is found to match one of the strings in this array, it is specially colored. For this to happen, the 'Identifier and Keyword Characters' string described below must be properly specified also.

BBLMSuffixMap	**Array of Dictionaries**

Each dictionary entry in this array should contain some or all of the following key/value pairs.

Key	Value Type
BBLMLanguageSuffix	**String**

Files with names that end with this string value are considered to be files of this module's language. The first character in the suffix string is usually a '.' (dot, period, full stop, whatever). This string must be present and non-empty or the entire dictionary entry will be ignored.

Bear in mind that if a suffix is given that overlaps with the suffix map of another language module or BBEdit's built-in languages, confusion may result. Fortunately, all of the suffix mappings can be seen in BBEdit's 'Languages' preference panel.

Key	Value Type
BBLMIsSourceKind	**Boolean**

If this key is present and its value is 'true', files with this suffix are considered 'source' files.

Key	Value Type
BBLMIsHeaderKind	**Boolean**

If this key is present and its value is 'true', files with this suffix are considered 'header' files.

If both the BBLMIsSourceKind and BBLMIsHeaderKind keys are present and have the value 'true', BBLMIsSourceKind takes precedence, but there should really be only one or the other or neither.

If the module's language has a concept of source versus header files and the appropriate values are specified (for example, files with names ending with ".h" are considered header files for C++, whereas files with names ending with ".cp" are considered source files), users will be able to jump between source and header files that share a common prefix (e.g. "foobar.h" and "foobar.cp") using command-tab.

Key	Value Type
Language Features	**Dictionary**

This dictionary is simply a collection of key/value pairs that define the language elements that the module supports.

Key	Value Type
Identifier and Keyword Characters	**String**

Most languages have keywords and identify other language elements with names that are words made up of letters, digits, and possibly other special characters. The function scanner looks for complete and unbroken sequences of such characters and then tries to decide whether the 'word' is a keyword or some other identifier. This string should contain all of the characters that can be in such a word.

Thus, a typical value for this string might be: "0123456789ABCDEFGH IJKLMNOPQRSTUVWXYZ_abcdefghijklmnopqrstuvwxyz"

which is the set of characters used in many languages for keywords and other identifiers. Note that the character need not be in any particular order.

Key	Value Type
Prefix for Functions	**String**
Prefix for Procedures	**String**

In some languages, function definitions begin with a specific keyword. For example, Pascal has functions that return values begin with the keyword 'function' and functions that return no values begin with the keyword 'procedure'. Other languages, such as C and C++, have functions begin simply with their names and other attributes, followed by a list of parameters, followed by the statement block that comprises the function body.

If one or both of these prefix strings is present in the plist and non-empty, the function scanner will look for Pascal-style function definitions, otherwise it will look for C-style function definitions.

Key	Value Type
Open Parameter Lists	**String**
Close Parameter Lists	**String**

A function's list of parameters is almost always enclosed by matching left and right parentheses (probably due to the tradition in mathematics), though there are exceptions. Note that in C-style function definitions empty parameter lists must be designated at a minimum by "()" (or whatever pair of delimiters applies), whereas in Pascal-style function definitions even the delimiters may be omitted.

Key	Value Type
Terminator for Prototypes 1	**String**
Terminator for Prototypes 2	**String**

Some languages allow for a function definition to appear without a body so that other functions that reference it know its "interface" without needing to know its "implementation". Sometimes a keyword is used as a substitute for the body -- in Pascal the keywords 'forward' or 'external' are used. In C-style languages, the function definition is usually just closed off with a semicolon after the parameter list. In either case, if one of the specified strings is encountered before the string value of 'Open Statement Blocks' described below, the function definition is considered to be a bodiless prototype and doesn't appear in the function popup menu.

Key	Value Type
Open Statement Blocks	**String**
Close Statement Blocks	**String**

Function bodies are usually "statement blocks" that begin and end with *something*. In Pascal, it is literally the keywords 'begin' and 'end'. In C and C-style languages it is usually the characters '{' and '}'. In both cases, such statement block can usually be nested inside one another and the function scanner takes this into account.

Note that some languages, such as VBScript, overload the keyword 'END' with another keyword, such as 'SUB', separating the two with one or more spaces. Visually, this is nice because it lets a human reader know what the 'END' ends, but it presents a problem for the function scanner, which is not prepared at this time to treat sequences of keywords as having special meaning. In theory, it would be possible to get by with specifying just 'END' or, more likely, just 'SUB' for the value of 'Close Statement Blocks', but in practice it's hard to say.

Key	Value Type
Open Block Comments	**String**
Close Block Comments	**String**

Block comments are multi-line comments that begin and end with special delimiters, such as '/*' and '*/' in C, and '{' and '}' in Pascal, where everything in between is ignored entirely.

Key	Value Type
Open Line Comments	**String**

Line comments begin with a special delimiters, such as '//' in C, and continue until the end of the line they begin on.

Key	Value Type
Open Strings 1	**String**
Close Strings 1	**String**
Escape Char in Strings 1	**String**
End-of-line Ends Strings 1	**Boolean**
Open Strings 2	**String**
Close Strings 2	**String**
Escape Char in Strings 2	**String**
End-of-line Ends Strings 2	**Boolean**

Most languages allow for strings to be enclosed by either matching single-quote characters or matching double-quote characters. Thus, two different sets of key/value pairs can be used for this or for any other kinds of strings that may be needed.

The "escape" values, if matched within a string, cause the character following not to be taken to have any special meaning and passed over. If the 'End-of-line Ends Strings' value is true, the "escape" can be used to pass over the end of the line and allow the string to continue on the following line. In any case, the "escape" may also be used to skip over the leading (or only) character in the 'Close Strings' value and allow that value to be included in the string.

Index

Symbols

H

I

K

L

M

N

Z