User Manual

BBEdit™ 8

Professional HTML and Text Editor for the Macintosh

Bare Bones Software, Inc.

BBEdit™ 8.2

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, Jerry Kindall, John Gruber, Jeff Mattson. Philip Borenstein, Simon Jester, Rich Siegel, Patrick Woolsey*
Additional Engineering	*Polaschek Computing http://www.polaschek-computing.com/*
Icon & Packaging Design	*Ultra Maroon Design http://www.ultramaroon.com/*
Exuberant Ctags	*© 1996-2004 Darren Hiebert http://ctags.sourceforge.net/*
PCRE Library Package	*written by Philip Hazel and © 1997-2000 University of Cambridge, England*
PopupFuncs™ Technology	*Eric Slosser*
Visibone color arrangements	*© 1999-2000 VisiBone http://www.visibone.com/*
HTML Tidy technology	*© 1998-2003 World Wide Web Consortium http://tidy.sourceforge.net/*

Bare Bones Software, Inc.
P. O. Box 1048
Bedford, MA 01730–01048

(781) 687-0700
(781) 687-0711 fax

http://www.barebones.com/

Sales & customer service: **sales@barebones.com**
Technical support: **support@barebones.com**

License Agreement:

You, the Licensee, assume responsibility for the selection of the program BBEdit 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

1 Welcome to BBEdit

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

In this chapter

Getting Started

Thank you for selecting BBEdit, the premier HTML and text editor for the Macintosh.

If you are new to BBEdit, we recommend that you read at least Chapters 1 through 4 of this manual to familiarize yourself with the installation and basic operation of BBEdit. You may also wish to read or preview any other chapters that cover features you frequently use. After you have installed BBEdit, the best way to learn it is to use it. Complete online assistance is available from the Help menu.

If you have used earlier versions of BBEdit, we recommend that you read at least Chapter 1 for an overview of significant changes in this version, and Chapter 2 for information relevant to installation and upgrading.

What Is BBEdit?

BBEdit is a high-performance HTML and 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, BBEdit does not offer fancy formatting capabilities, headers and footers, graphics tools, a thesaurus, or similar 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, BBEdit 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.

BBEdit also has features that make it easier to edit specific kinds of text, such as source files for programming languages and HTML (Hypertext Markup Language) files for the World Wide Web. In fact, since the rise of the Web, BBEdit has been the tool of choice for Macintosh Web designers who need more flexibility than visual Web authoring tools can provide.

How Can I Use BBEdit?

Use BBEdit any time you need to create or edit Web pages, source files, or text documents of any kind. Whether you need to find (or change!) all the occurrences of some text in a set of files, or modify or reformat large text files of any sort, or quickly tweak a Web page, BBEdit is the right tool for the job.

Development Environments

BBEdit found its initial following among the Macintosh programming community with its core editing- and development-oriented tools. Although we have added countless other features to BBEdit since its first incarnation, its source code editing capabilities are now stronger than ever.

In addition to offering syntax coloring and function browsing for many different languages, BBEdit integrates with the Xcode and CodeWarrior development environments, supports direct use of Perl, Python, or any other Unix scripting environment, and allows access to CVS and Perforce for source code control. Chapter 14 provides more information on how to set up BBEdit for this type of work.

Writing HTML Documents

BBEdit is an ideal tool for preparing and editing HTML documents (web pages). In addition to many options for preparing text content, such as wrapping, case changes, and searching, BBEdit offers a powerful set of tools to make editing web pages easier. The Tag Maker command speeds the creation of HTML tags, while the Edit Tag option allows quick changes to existing markup, and the HTML Tools palette makes dragging and dropping tags simple or lets you access commands with just a click.

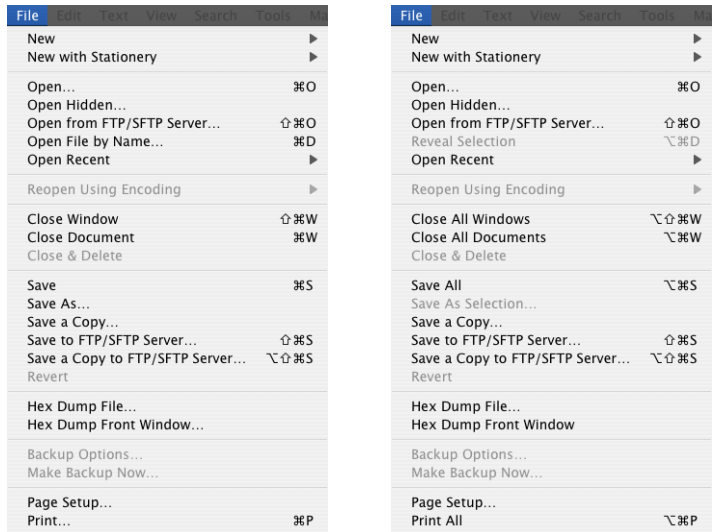
Using BBEdit, you can easily preview your work in most Macintosh Web browsers, including Safari, Internet Explorer, Netscape, Mozilla, OmniWeb, and Opera, as well as via BBEdit's native Preview feature. For more information on using the HTML Tools to create, edit, and preview Web pages, see Chapter 11.

Human Interface Notes

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

Dynamic Menus

IMPORTANT Many of BBEdit's pull-down menus are dynamic: if you hold down the Shift, Option, or Control 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, Option, or Control 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 BBEdit's commands have keyboard shortcuts. BBEdit lets you reassign the shortcuts for any menu item, glossary entry, plug-in, or script to suit your own way of working.

To change the keyboard shortcut for any menu command (as well as some document status bar options), choose the Set Menu Keys command from the BBEdit menu.

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

Feature	Set Keys in…
Menu commands	Set Menu Keys
Plug-ins	Plug-In Tools palette
Glossary items	Glossary palette
Scripts	Scripts palette
Unix filters and scripts	Unix Filters and Scripts palettes
Stationery	Stationery palette

To display any of BBEdit's floating palettes, 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, BBEdit's contextual menu will display a set of commands relevant to that location or text, as well as some appropriate standard commands (such as Cut/Copy/Paste, or Check Spelling) so you do not have to hunt around in the menu bar for them.

Snappy Palettes

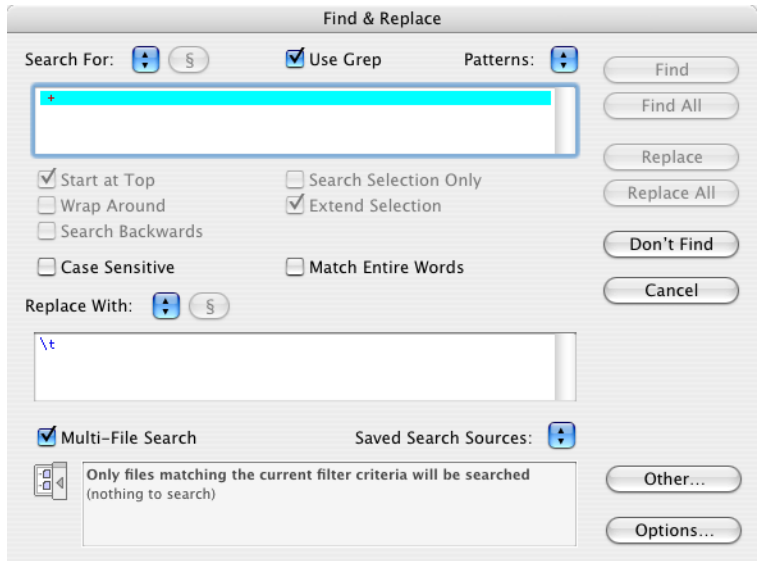
When you move or resize palettes (floating windows), they will "snap" to the edges of the screen and the edges of other palettes. You can override this behavior by holding down the Shift key while dragging or resizing.

Dialog Box and Sheet Key Equivalents

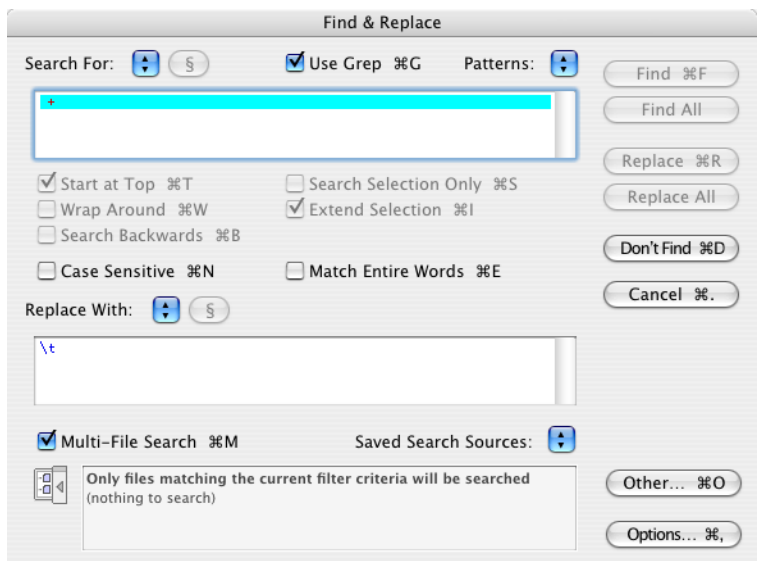
You can use key equivalents to click buttons or select options in most of BBEdit's dialog boxes and window sheets. Certain keys have the same meaning in all dialogs and sheets:

- 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 text field.

To see the other key equivalents for a particular dialog or sheet, hold down the Command key. After a brief delay, labels appear next to any buttons or options which have key equivalents. 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:



BBEdit waits briefly before displaying the Command-key equivalents so that you can type a sequence quickly without encountering visible flicker.

Feature Highlights

BBEdit 8 marks a new milestone in the evolution of the product. Here are just a few of the many features and enhancements present in this version:

- Support for opening and managing multiple text documents via a single window interface.
- Automator support for incorporating BBEdit's text processing power in workflows (requires Mac OS X 10.4)
- Text Factories for applying multiple text processing actions to diverse source documents.
- 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.
- Codeless language modules offer readily extensible syntax coloring support.
- Server-based preview option for HTML documents.
- CSS 2.1 support in the HTML Tools via new editing commands and Tag Maker.
- HTML syntax checker support for checking partial documents and excluding content sections.
- Command line tool enhancements including support in the "bbedit" tool for opening files from FTP/SFTP servers, and a new "bbdiff" tool.
- Text menu commands expanded and enhanced: imported capabilities of former plug-in tools, sheet support, improved recordability, and more.
- Integrated support for the Subversion and Perforce (P4) source control systems.
- Spell checking now uses the Mac OS X system spelling checker.
- Integrated support for Exuberant Ctags.
- Numerous CVS improvements, including CVSEDITOR support for checkin comments.
- Affrus support for debugging Perl Scripts.
- Streamlined application support folder handling allows both global and local add-in items.
- New visual Page Guide and tab stop indicators.
- Current line highlighting preference.

Info on New Features

In addition to these major enhancements, BBEdit 8 also contains a plethora of additional features and interface and performance enhancements, as well as bug fixes. For a detailed summary of changes and bug fixes, please refer to the current release notes, which are available in the BBEdit Support section of our web site.

`http://www.barebones.com/support/bbedit/current_notes.shtml`

Support Services

The Bare Bones Software web site provides up-to-date information on 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 BBEdit 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 BBEdit or any other Bare Bones product, and you can't find the information you need on our web site, send email to <support@barebones.com>

CHAPTER

2

Installing BBEdit

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

In this chapter

Basic Installation

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

System Requirements

IMPORTANT BBEdit 8 requires Mac OS X 10.3.5 or later; Mac OS X 10.4 or later is required for Automator support. The software will not run on Mac OS 9, or any earlier versions of Mac OS X.

Installing BBEdit

Depending on how you obtained BBEdit, you will receive either a disk image (a ".dmg" file) or a CD-ROM. To install BBEdit, just drag the "BBEdit" 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 BBEdit when future versions become available, you need only quit the BBEdit application, and replace it with the updated version. The first time you launch a newer version of the software, BBEdit will prompt you should any further actions, such as updating the command-line tools, be needed.

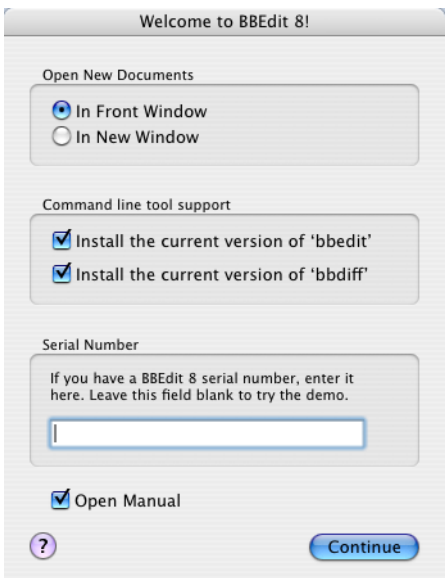
Upgrading from a Previous Version

IMPORTANT In addition to installing the BBEdit 8 application, if you are upgrading from a previous version, you will need to manually copy across any items you added to that version's "BBEdit Support" folder into BBEdit's new application support folder. You should **not** simply rename your existing "BBEdit Support" folder. (See "BBEdit's Application Support Folders" on page 12.)

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

First Run Configuration

The first time you launch BBEdit 8, it will display the "Welcome to BBEdit" dialog. This dialog allows you to choose whether to display multiple text documents in a single window versus one document per window, whether to install the command-line tools, and whether to immediately open the PDF user manual.



Opening New Documents

Since BBEdit now supports opening multiple documents into a single text window, you must decide whether the application should work in this manner, or whether it should behave in the standard fashion by opening each document into its own window. (A document represents either a file which you open for editing, a text document created by the New Document command on the File menu, or any similar item, such as a text document created via the scripting interface.)

Choose "In Front Window" to have BBEdit open all new documents into a single text window. Choose "In New Window" to have BBEdit open each new document into its own text window.

Note If you change your mind later, you can adjust these options in the Documents panel of the Preferences window.

Entering Your Serial Number

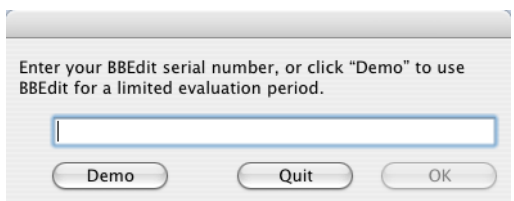
The dialog also enables you to enter the unique product serial number that you received at the time of purchase (either with your email order invoice, or on your product CD-ROM) to activate BBEdit. Once you enter the serial number, your copy of BBEdit will be activated, and all demo restrictions will be removed.

If you have not yet purchased the product and thus do not have a serial number, leave this space blank. BBEdit will operate as a fully functional demo for a limited trial period, after which you must purchase a license and enter a valid serial number in order to continue using it.

If BBEdit has already been activated, the dialog will display the associated serial number. You may continue using this serial number, or enter a different serial number if you wish to change the active license.

Activating the Demo

While running in demo mode, BBEdit will operate with full functionality for a limited period of time. Upon expiration of this trial period, you must purchase the product and receive a serial number in order to continue using it. To activate the demo, you may either enter your product serial number into the dialog that BBEdit displays at launch, or, you may choose the Enter Serial Number command from the BBEdit menu to bring up the product activation dialog at any time.



BBEdit's Application Support Folders

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

IMPORTANT BBEdit 8 does not make use of the "BBEdit Support" folder that previous versions employed, nor can you can place the application support folder alongside BBEdit itself. Instead, BBEdit's application support folder must be present in either or both of the following locations:

- Global (items available to all users):
  /Library/Application Support/BBEdit/
- Local (user-specific items):
  ~/Library/Application Support/BBEdit/

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/BBEdit/".

Using the Global Application Support Folder

You can use the global application support folder to provide a common set of supporting items in BBEdit 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 BBEdit, 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 BBEdit starts up, BBEdit 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 BBEdit, and you can remove any or all of them, they provide additional functionality that you may wish to retain.

Application Support Folder Contents

BBEdit'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, BBEdit will use the latest version regardless of location. For all other items, BBEdit lists the global and local sets separately.

To prevent alias loops, BBEdit 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 BBEdit and other applications, and that you not make aliases to plug-ins or scripts located on remote (server) volumes.

Glossary

[Global, Local]

The Glossary folder holds text files whose contents can be inserted into an editing window with the click of a button or a keystroke. Glossary files can also use special placeholders to insert varying or context-sensitive information—for example, a date or the name of the current file.

Initially, your local BBEdit application support folder will be populated with several sample glossary sets (for C source code, HTML, WML, Mac OS X Property List files, RSS, and plain text), which you can use or modify as you like. See Chapter 12 for more information on the Glossary.

Upgrading You should move or copy over any third-party glossary packages, or any custom glossary items which you have created, that you wish to preserve.

HTML Templates

This folder contains a sample set of HTML document templates for use with the New HTML Document command, which you can modify or add to as you wish. In order to use these templates, you must either choose this folder to be the Templates & Includes folder for an existing web site in the HTML Web Sites preferences, or you can copy the template files into an already-designated site templates folder. Please see Chapter 11 for more information on BBEdit's HTML tools.

Upgrading You should move or copy over any customized template or include documents that you wish to preserve.

Language Modules

[Global, Local]

This folder contains plug-in modules that tell BBEdit 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 BBEdit menu items. See Chapter 13 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 BBEdit. Each plug-in adds a command to BBEdit'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 BBEdit application icon in the Finder. BBEdit 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 BBEdit 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 BBEdit plug-ins.

Upgrading If you have any compatible third-party plug-ins that you wish to use with BBEdit 8, 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 to use with BBEdit 8, and BBEdit will warn you if it encounters any such items. (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 BBEdit may not work with BBEdit 8. (Please see Chapter 13 for more details and tips on modifying your existing scripts.)

Shutdown Items

[Local only]

The items in this folder are opened when you quit BBEdit. 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 BBEdit 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 BBEdit•Quit.

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

Startup Items

[Local only]

When launched, BBEdit 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 BBEdit knows how to handle (such as text files or file groups), BBEdit will open them directly. If you place a compiled OSA (AppleScript, or any other OSA-compliant scripting language) script in this folder, BBEdit will execute the script. If you place a folder alias here, BBEdit will open a disk browser window based at that folder.

If you place other types of items in this folder, BBEdit will ask the Finder to open them. If you often edit HTML files, for instance, you may want to place an alias to your Web browser (or your visual HTML editor) in the BBEdit Startup Items folder so that it will start up automatically whenever you run BBEdit.

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 BBEdit'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 files, which you can access via the Apply Text Factory command, the Text Factories menu, or the Text Factories palette. For more information on creating and using text factories, see "Text Factories" on page 91.

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 14, "Working with Development Tools," 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 BBEdit, it may create the files and folders noted in this section.

BBEdit Preferences File

All of BBEdit's basic preference settings are stored in the file "~/Library/Preferences/com.barebones.bbedit.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 BBEdit by using appropriate "defaults write" commands (see "Optional settings via 'defaults write'" on page 199).

Upgrading BBEdit 8 will import relevant preference settings from previous versions of BBEdit, if a suitable "BBEdit Prefs Data" file is available, i.e. can be found within a "BBEdit Preferences" folder in either the old BBEdit Support folder or in ~/Library/Preferences.

BBEdit Preferences Folder

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

Document State.plist

BBEdit stores state information for individual documents in this file.

File Filters

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

FTP Bookmarks.xml

BBEdit 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 preference panel to add, remove, or modify your bookmarks.

Grep Patterns.xml

BBEdit stores user-defined search patterns in this XML file, which is located in your active BBEdit Preferences folder.

Upgrading If you have created and saved any custom grep patterns in a previous version of BBEdit, 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 BBEdit'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.

CHAPTER

3 Working with Files

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

In this chapter

Launching BBEdit

To launch BBEdit, double-click the BBEdit application icon or a BBEdit 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, BBEdit 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, BBEdit will look for a folder named Startup Items in the its application support folder (see "BBEdit's Application Support Folders" on page 12). If this folder is found, BBEdit will open any items it finds in the folder.

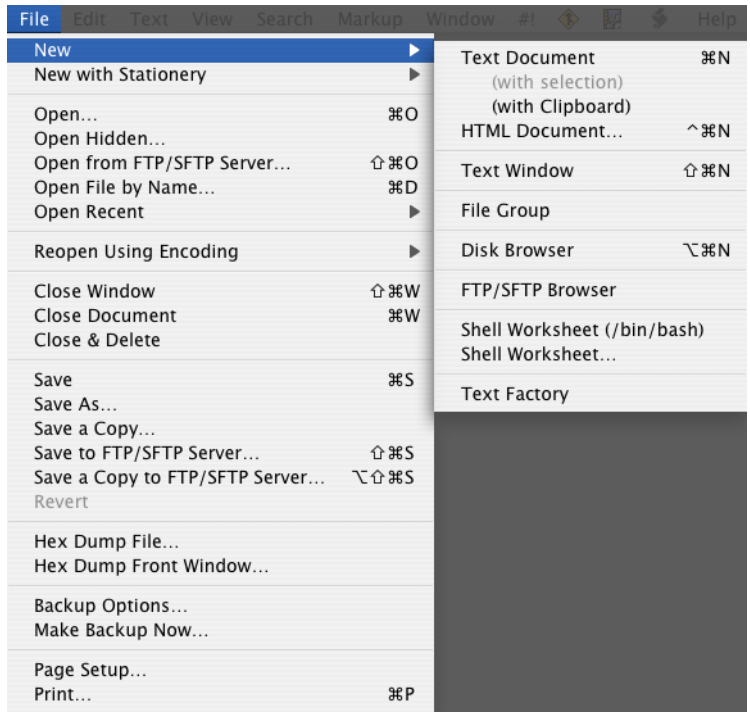
If the items present are documents of a type that BBEdit knows how to handle (such as text files or file groups), BBEdit will open them directly. If you place a compiled AppleScript in this folder, BBEdit will execute the script. If you place a folder alias here, BBEdit will open a disk browser window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

If you place other types of items in this folder, BBEdit will ask the Finder to open them. If you often edit HTML files, for instance, you may want to place an alias to your Web browser (or your visual HTML editor) in the BBEdit Startup Items folder so that it will start up automatically whenever you run BBEdit.

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 BBEdit's application support folder. Be sure to name the alias "Startup Items" so that BBEdit can locate it.

Creating and Saving Documents

To create a new text document or special-purpose window within BBEdit, pull down the File menu and open the New submenu. Since BBEdit 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.
- HTML Document: Brings up a dialog with options for creating a new HTML document (see Chapter 11 for more information on working with HTML documents).
- Text Window: Opens a new text window (see "Text Windows" later in this chapter for more information).
- File Group: Opens a new file group window (see "Using File Groups" later in this chapter for more information).
- Disk Browser: Opens a new disk browser (see Chapter 9 for more information).

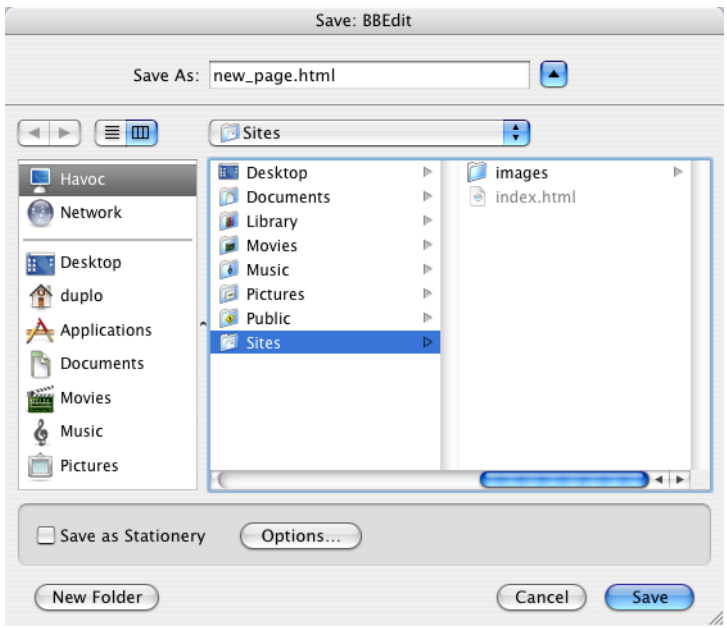
- FTP/SFTP Browser: Opens a new FTP/SFTP browser (see later in this chapter for more information).
- Shell Worksheet (*shell*): Opens a new shell worksheet using your default shell (as set in the Tools panel of the Preferences window).
- Shell Worksheet…: Opens a dialog listing all the Unix shells installed on your system; choose from the list, and a new shell worksheet will be opened using that shell.
- Text Factory: Opens a new Text Factory window (see "Text Factories" in Chapter 5 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 BBEdit submenu of the Services menu. BBEdit 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.

BBEdit 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 BBEdit'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 BBEdit 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. BBEdit displays the Save dialog with the selected text already entered as the file's name.

Saving a Copy of a File

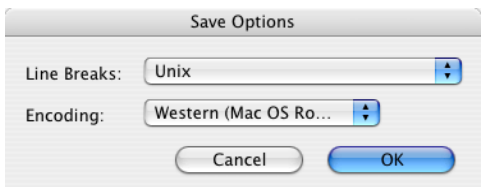
You can save a copy of a file with BBEdit's Save a Copy command in the File menu. Just like the Save As command, the Save a Copy command displays a Save dialog and lets you choose a name and location for the file. However, unlike the Save As command, where BBEdit will start working with the new file 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, BBEdit saves the changes to Test.c and not to Backup-Test.c.

File Saving Options

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

- Save As Stationery: When this option is selected, BBEdit saves the document as a stationery pad. When you later open this document, BBEdit 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 BBEdit 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 BBEdit 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 31.)

Setting the Encoding

BBEdit 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 31). 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 188).

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 BBEdit 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, BBEdit 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 BBEdit saves document states by means of the Save Document State option in the Text Files: Saving panel of the Preferences window.

BBEdit no longer saves document state in the resource fork of the document's file, and does not distinguish between the "MPW" and "BBEdit" 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

BBEdit 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

BBEdit 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 BBEdit'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, BBEdit will display a slash through the pencil icon in the status bar. To edit the file, click the pencil icon. BBEdit 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. BBEdit 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 BBEdit.

- Double-click any file with a BBEdit document icon.
- If BBEdit is running, choose the Open, Open Hidden, or Open Recent command from the File menu.
- Select the name of a file in a BBEdit 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 BBEdit icon or to an alias of the icon.
- Select a file in the Finder, and use the Open File command from the BBEdit submenu of the Services menu.

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

You can view any image files or movie files whose format is supported in QuickTime by opening them with BBEdit. You can also have BBEdit ask QuickTime to try to open non-text files (by selecting the appropriate setting in the Application panel of the Preferences window). BBEdit can then open image files and sounds in their own windows, rather than displaying them as text. If this preference is not active, BBEdit 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, BBEdit will automatically examine its contents for any indication of the proper encoding, and attempt to handle it appropriately. If BBEdit 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 BBEdit goes through to determine the proper encoding for a file:

1. **If the file is well-formed HTML or XML, BBEdit looks for an "encoding=" or <meta charset=> directive.**
2. **If the file contains a BBEdit state resource, BBEdit uses the encoding stored in the state resource.**
3. **If the file contains a UTF-8 or UTF-16 (Unicode) byte-order mark, BBEdit 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, BBEdit opens the file as a Unicode file.**
5. **If you are opening the file with the Open command, BBEdit 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.

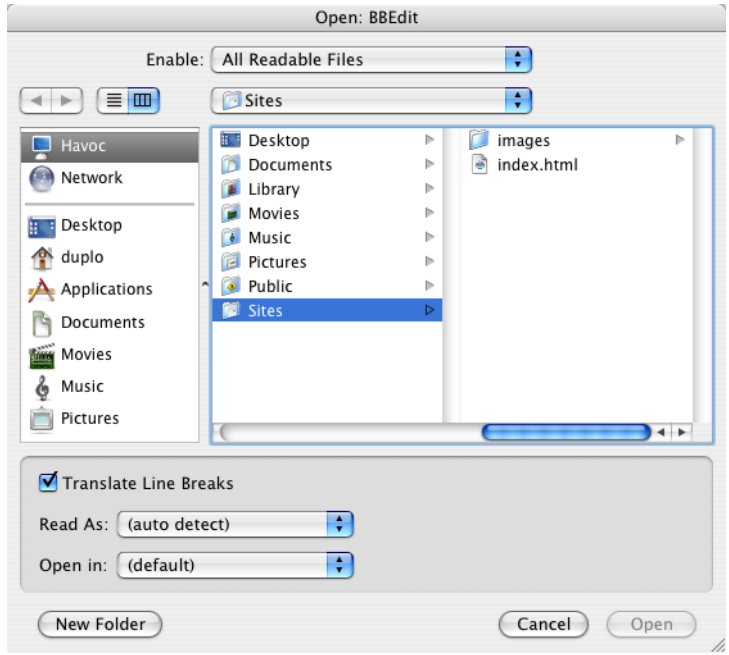
Note If an encoding change results in the conversion of a document's contents from a single-byte script to a multi-byte script, BBEdit will mark the document as being "dirty" or changed.

Using the Open Command

To open a file with the Open command:

1 **Choose Open from the File menu.**

BBEdit 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, BBEdit will inspect the file's name to see if it can determine whether the file is text or not. BBEdit 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 BBEdit'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, BBEdit will next apply the Internet Config file name mappings. (See Chapter 10, "Preferences," for more details.

Translate Line Breaks

When this option is selected, BBEdit translates DOS or Unix line breaks when opening a file. Otherwise, BBEdit 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 BBEdit what encoding to use by choosing it from this pop-up menu. Usually, BBEdit 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): BBEdit will open the selected documents according to your preference settings.
- Front Window: BBEdit 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: BBEdit will open all of the selected documents into a new text window.
- Separate Windows: BBEdit 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, BBEdit will automatically expand the selection to include the suffix.

BBEdit 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, BBEdit will look in the following locations, in order:

- If the selected file name is surrounded by angle brackets, BBEdit will start its search in the folder that you have specified in the Search Folders list of the File Search preference panel.
- If the Open Selection is being done from a shell worksheet, BBEdit will search the shell’s current working directory, followed by any subdirectories within it.
- Otherwise, BBEdit will look first in the same folder as the file containing the selected file name, and then in any subfolders within that folder.
- If the file containing the selection is contained within one of the web sites defined in the HTML Web Sites preference panel, BBEdit will next search the “Templates & Includes” and site root folders, if any, associated with that site, followed by those for the default site designated in the preference panel.
- If the CodeWarrior IDE is running and a project is open, BBEdit will ask the IDE for the search paths to the current project’s current target, and look for files in those folders as well.
- If BBEdit 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 BBEdit looks in. Normally, BBEdit opens the first one it encounters, and then stops. If you want BBEdit 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.)

BBEdit 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 preference 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 BBEdit

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

BBEdit 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 BBEdit can automatically switch to the correct input method when you change document windows.)

You can use international text throughout BBEdit—for example, in the Find & Replace dialog, in the HTML Tools, and everywhere else you would use Roman text. Likewise, BBEdit 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).

BBEdit 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, BBEdit 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, BBEdit respects any available information about each file's encoding. If a file does not contain any information about its encoding, BBEdit will use the default encoding set in the Text Files: Opening 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/
```

BBEdit 8 includes full support for and makes extensive use of Unicode, in addition to all other OS-supported text encodings. In particular, BBEdit internally represents all documents as Unicode, regardless of their on-disk encoding.

Saving Unicode Files

BBEdit 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, BBEdit 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, BBEdit 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 BBEdit 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, BBEdit will also recognize such files with type 'UTF8'.

Opening Unicode Files

When opening files, BBEdit 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 BBEdit 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', BBEdit 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, BBEdit will present an alert to warn you. When such a file is encountered during multi-file searching, a warning will be logged.

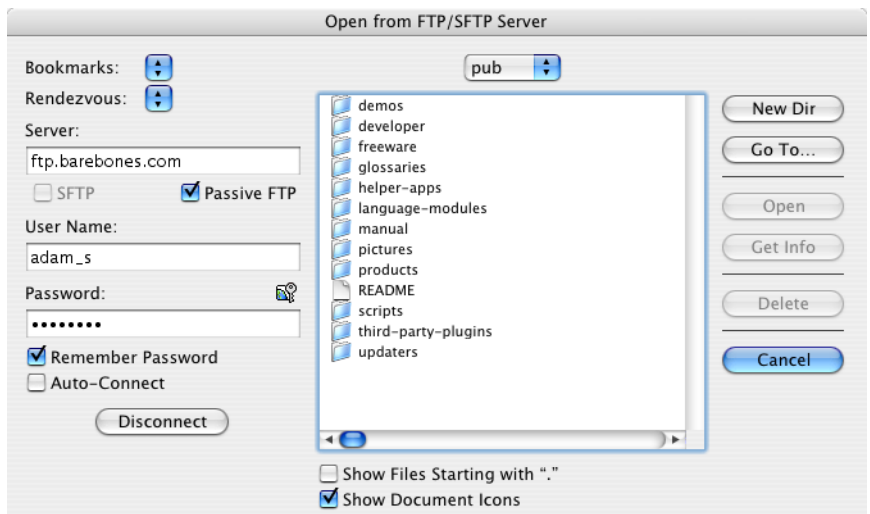
Accessing FTP/SFTP Servers

BBEdit 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 BBEdit 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 preference panel.

The Passive FTP checkbox tells BBEdit to open a connection to the FTP server in passive mode, in which the server will send BBEdit the number of the port to use for the session. If this option is turned off, BBEdit 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 BBEdit 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!)

BBEdit also supports FTP and SFTP URL clippings. You can make a clipping of the FTP or SFTP URL for a file, add the clipping to a file group, and double-click it, and BBEdit will open the specified file for editing. If the clipping contains the URL for a directory, BBEdit will open a new FTP/SFTP Browser window (see the next section for more information about FTP/SFTP Browsers). Alternatively, you can double-click an FTP or SFTP clipping in a disk browser, or drop one on BBEdit's icon in the Finder, with the same results as just described.

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.

Once you have selected a file and opened it, BBEdit 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, BBEdit downloads it "raw" (in binary mode) and then performs a standard line ending conversion on the downloaded file.

Specifying Alternate Ports

BBEdit 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

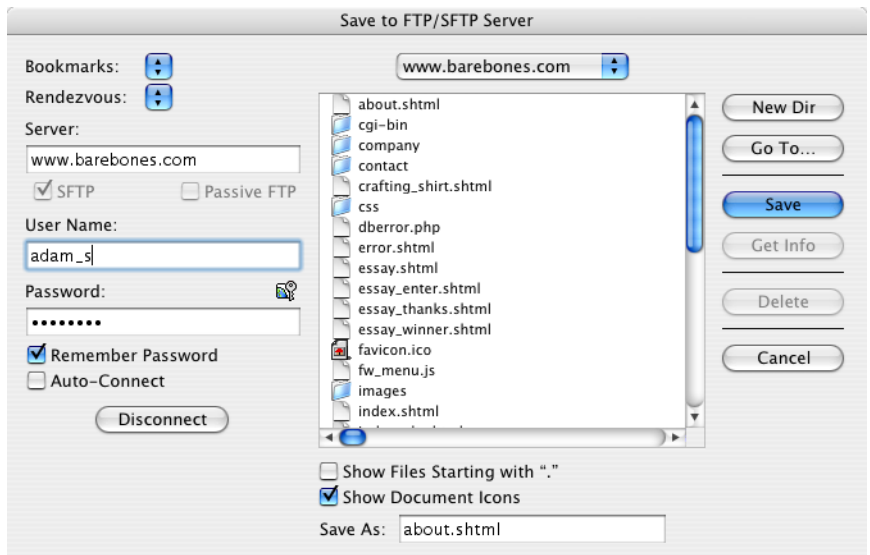
To store FTP passwords, BBEdit takes advantage of the system Keychain. As long as the Keychain is unlocked, whenever you enter a server and user name pair for which there is a record, the corresponding password will be filled in automatically. This will happen regardless of whether the Remember Password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

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.

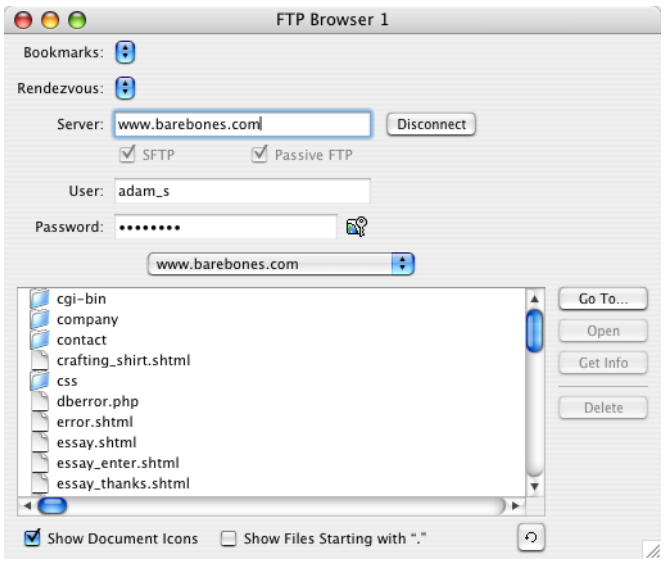


Note When you save a file to an FTP or SFTP server using either Save or Save to FTP/SFTP Server, and the file has Unix (LF) or Windows (CR+LF) line endings, BBEdit uploads the file in binary mode, preserving its line endings exactly as they are on your local machine. However, if the file has Macintosh (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, BBEdit 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.

You can double-click a folder to change directories. If you hold down the Option key when opening a folder, it will open in a new FTP/SFTP Browser window. You can select a range of files and directories by Shift-clicking, and you can select (and deselect) multiple items one at a time by Command-clicking.

Using BBEdit from the Command Line

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

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

```
bbedit filename
```

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

```
bbedit -l
```

In addition to files, you can also specify FTP or SFTP URLs to files or directories, to have BBEdit 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 "bbedit" tool, and it will open in a new untitled window in BBEdit: for example,

```
ls -la | bbedit
```

If you just type

```
bbedit
```

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

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

```
bbedit [ -bchlpsuvVw --(long_form_switches) ]
 [ -e <encoding_name> ] [ -t <string> ]  [ +<n> ]
 [ file (or) <S/FTP URL> ... ]
```

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

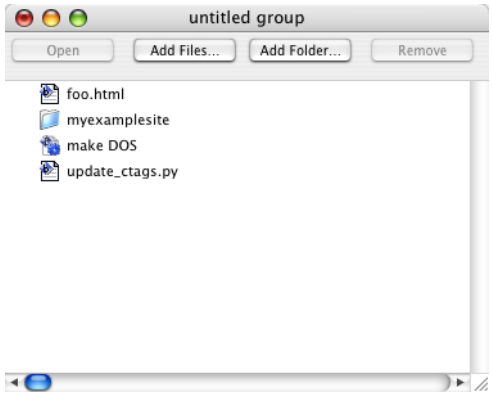
Using File Groups

If you frequently work with many related files, you may want to create a file group for them. A file group is a special kind of BBEdit file that contains references to other files and folders. These files will usually be BBEdit text files, but you can place any kind of file in a file group, including aliases to a file and URL clippings.

File groups are persistent. Once you have created one, you can save it to disk using the Save command, and then open it in a later BBEdit session to have instant access to the same set of files.

Creating a File Group

To create a new file group, pull down the File menu and choose File Group from the New submenu. A new file group window appears.



To add files to a file group, drag them from the Finder into the file group window, or click the Add Files button. When you click Add Files, BBEdit presents the Open dialog in which you can choose one or more files to add. You can also add a file by dragging its icon from a text window's status bar or document drawer to the group window, or by dragging a file entry from any results browser.

In addition to file and folder references, BBEdit supports URLs in file groups. You can drag a URL (text or clipping file) to a file group window, and the URL will be saved in the group. If you subsequently open the item, BBEdit will hand off the URL to designated helper, or open it directly if it is an FTP URL.

To add folders to a file group, drag them from the Finder into the file group window, or click the Add Folder button. BBEdit displays the following sheet:

You can click Choose to choose a folder using an Open dialog, or just drag the desired folder to the white area from the Finder. You can add both folders and files, just files, or just folders, and you can choose to also add all nested folders and, optionally, to skip folders whose names are enclosed in parentheses.

When adding a folder to a file group with the Add button, you can choose whether to add all of the folder's contents. By choosing *not* to add a folder's contents, you can create a file group that consists only of folders, which can be handy for navigating and searching project roots without worrying about keeping in sync with the folders' contents.

Once you have added items to a file group, you can save the file group to disk for later use.

Using File Groups

To open a item in a file group, double-click it, or select it and click the Open button. If the item is a BBEdit document, BBEdit opens it. If it is a folder, it is opened in a disk browser. If it is an FTP URL clipping, BBEdit will open the remote file (or open an FTP Browser if the clipping points to a directory). Otherwise, BBEdit tells the Finder to open the file.

If you added nested folders, they appear in the file group with disclosure triangles, as in a Finder list view. Click the triangle to reveal the files and folders inside that folder.

You can use a file group as the basis of a multi-file search. See Chapter 7, "Searching," for more information.

Removing Files from a File Group

To remove a file from a file group, drag it from the file group into the Trash in the Finder, or select the items you want to remove and click the Remove button.

Note Removing an item from a file group, even by dragging it to the Trash, does not delete the original file—only its entry in the file group.

Using Stationery

Like most Macintosh applications, BBEdit 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 BBEdit. 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 BBEdit's application support folder).

- Use BBEdit'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 BBEdit'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)Web template" would sort before "01)HTML Template". For such files, the first three characters are not displayed in BBEdit. 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 Document 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

BBEdit 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.

BBEdit 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.

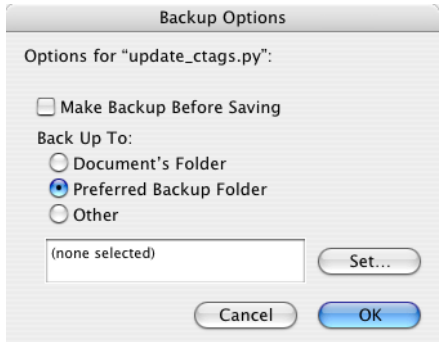
If an error occurs during the backup process, BBEdit will now report a descriptive error, rather than just an OS result code. (The usual cause of such an error is that the defined backup directory has been deleted.)

Automatic Backups

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

1 **Choose the Backup Options command from the File menu.**

BBEdit opens the Backup Options dialog box.



2 **Specify the folder in which you want BBEdit to save the backups.**

You can choose the document folder, the preferred backup folder (set in BBEdit'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.**

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

4 **Click OK.**

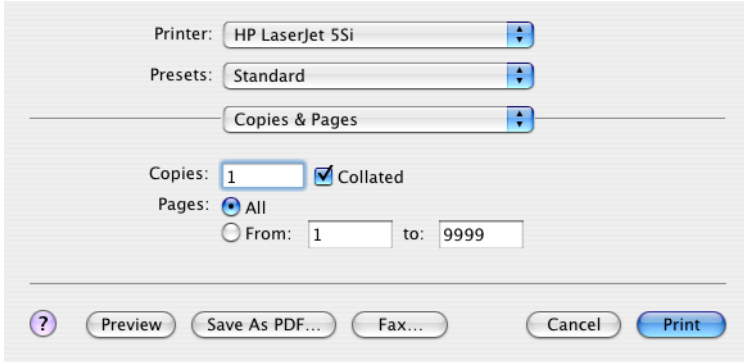
Manual Backups

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.

BBEdit 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

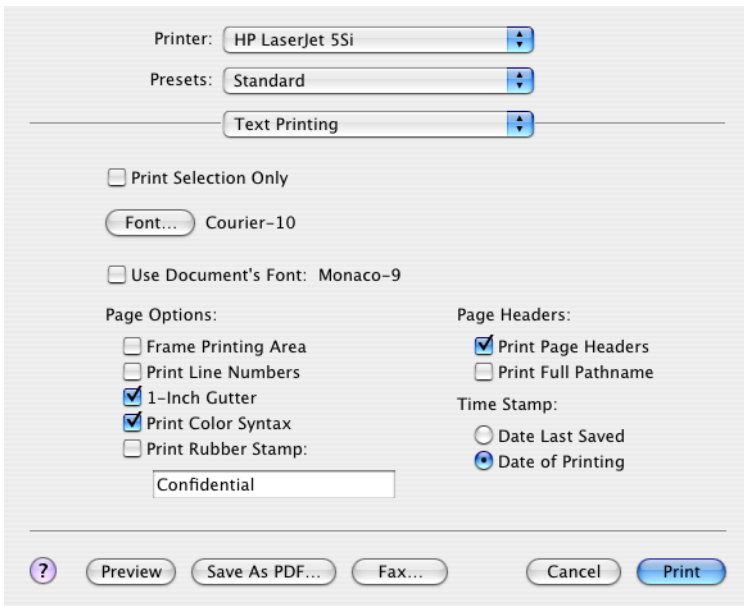
To print a document, choose the Print command from the File menu. BBEdit will display a standard print sheet in that document's window.



To print one copy of the active document without displaying the print sheet, choose the Print One Copy command from the File menu.

Text Printing Options

You can access all of BBEdit'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, BBEdit 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, BBEdit uses the document's font when printing.

Page Options:
Choose the Printing Options command in the Edit menu to drop a sheet which lets you specify various print formatting options for a document. You can also access this sheet by clicking the Options button in the BBEdit section of the Print sheet. You can set default printing options in the Text Printing preferences panel.

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

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

1-Inch Gutter
When this option is selected, BBEdit 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, BBEdit will print the document in color.

Print Rubber Stamp
When this option is selected, BBEdit 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. BBEdit chooses the right-size font to print the message.

If your printer supports grayscale printing, BBEdit 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, BBEdit 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, BBEdit 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.

CHAPTER

4

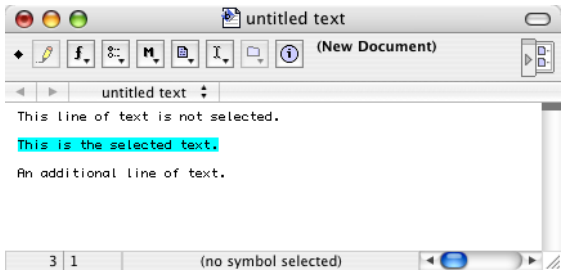
Editing Text with BBEdit

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

In this chapter

Basic Editing

BBEdit 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 using the system highlight color, which you can set in the Appearance panel of the System Preferences.



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 BBEdit 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.**

   BBEdit 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 BBEdit 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.

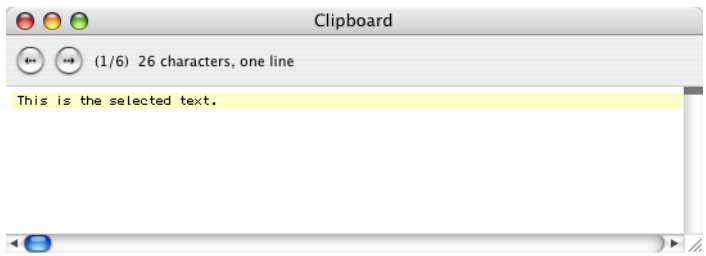
To place text on the clipboard without deleting it, choose Copy from the Edit menu.

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, BBEdit changes these commands to Cut & Append and Copy & Append.

Multiple Clipboards

BBEdit supports six separate clipboards. Each time you use the Cut or Copy command, BBEdit 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, BBEdit 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, BBEdit 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 BBEdit. 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

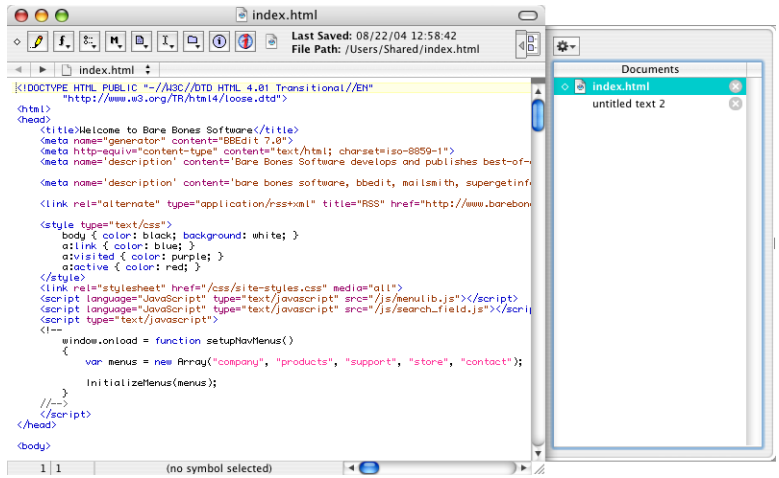
BBEdit provides the ability to undo multiple edits, one action at a time. The number of edits that may be undone is limited only by available memory. The practical limitation is determined by the extent of the edits and the amount of free memory.

BBEdit 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.

There is also a Clear Undo History menu command (Command-Control-Z), which will 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

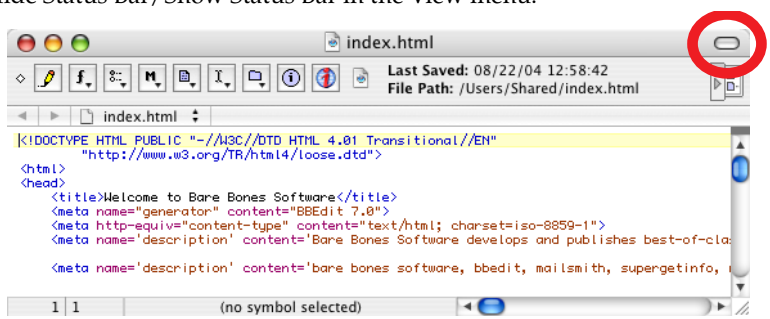
BBEdit text windows have the same controls you are familiar with from other Macintosh applications (for example, text windows are resizable and zoomable, and have both vertical and horizontal scroll bars). Some additional elements which may be less familiar are the status bar, the split bar, the navigation bar, and the documents drawer.



IMPORTANT You can choose whether BBEdit 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 preference panel (see page 168). 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.

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, the disk is locked, or the file is part of a source-control system project (such as Perforce or CVS) and is checked out. If the file is not on a locked disk, you can click the pencil icon to toggle the document's editability.
	The Function pop-up menu gives quick access to routines and functions in languages that BBEdit 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, as well as all BBEdit include files referenced by the document.
	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.
	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.
	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.
	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 Window 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 BBEdit 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.

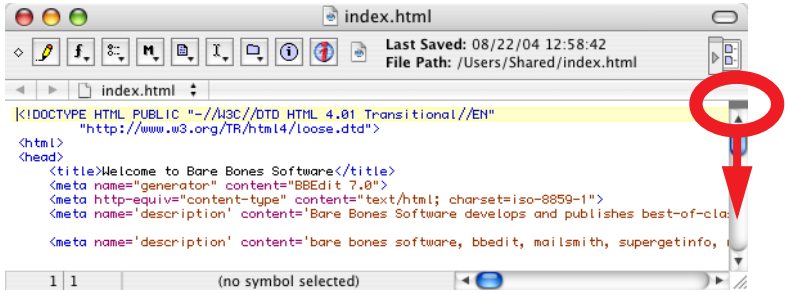
Indicator	Meaning
•	The function containing the insertion point
†	C/C++ typedef
◊	C/C++ “#pragma mark” directive
italic name	C/C++ function prototype
1-6	Heading level (in HTML files)
tag name	Tag name for the indicated name or ID attribute value (in HTML files)

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.

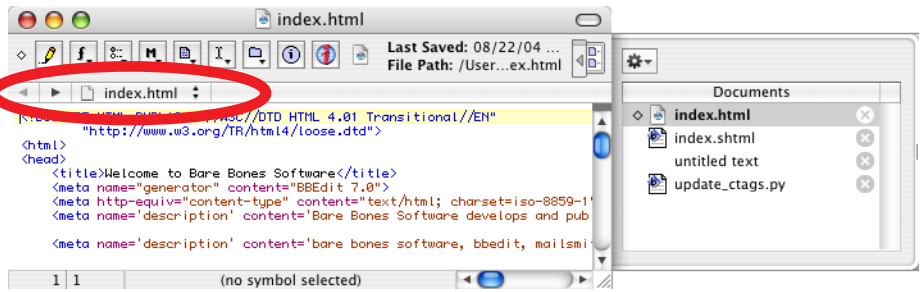


To collapse the text pane back down to a single region, drag the split bar (starting from anywhere along its length, not just at its right end) back up to its original position.

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.

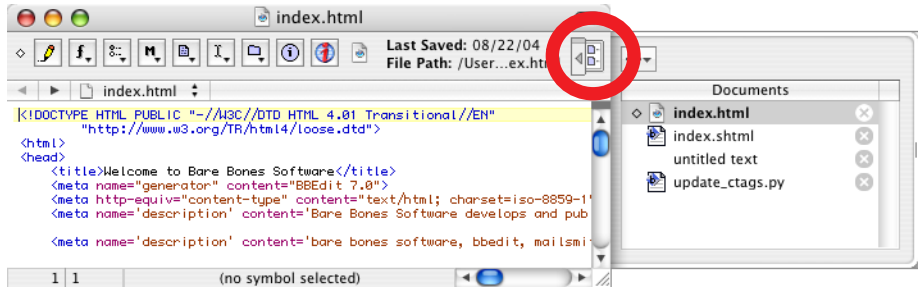


To hide the navigation bar, choose Hide Navigation Bar in the View menu, or turn off the Show Navigation Bar option in the Text Status Display preferences panel.

IMPORTANT The Previous and Next buttons in the Navigation bar, as well as the Previous Document/Next Document commands, now select documents in most-recently used order, rather than alphabetical order.

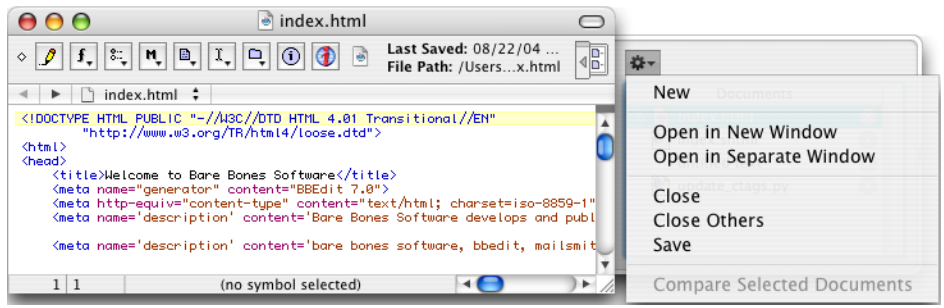
The Documents Drawer

The documents drawer is a panel which BBEdit 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.



IMPORTANT You can control whether BBEdit 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 preference 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 198..)

The document drawer also contains an action menu, much like the Finder's, which you can use to rearrange documents.



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.

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 most-recently viewed order. If the frontmost text window contains only one document, these commands will be disabled.

Note In previous releases, these commands selected documents in alphabetical order, corresponding to the display order shown in the documents drawer.

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 an Info dialog for the active document. (Equivalent to clicking the Info button in the status bar.)

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

BBEdit 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	–none–

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

You can hold down the Command or Option keys when clicking or double-clicking to trigger special actions:

	Option	Command
Click	–none–	Open URL
Double-click	Look up selected word in programming reference	–none–

BBEdit 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 preference 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 Text Editing preferences, 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 BBEdit menu).**
2. **Select the Text Editing preferences 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

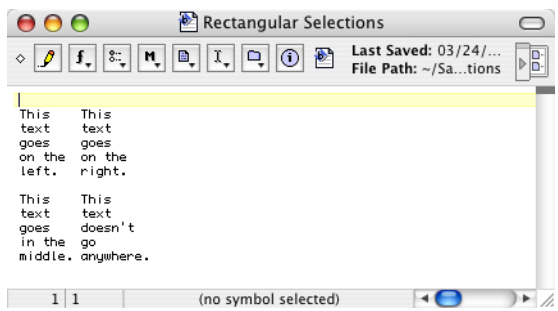
By holding down the Option key as you drag, or holding down the Shift and Option keys while clicking, you can select all text lying within a specified rectangular area. You can then perform all of the normal editing operations on this "rectangular selection," such as Cut, Copy, Paste, or drag and drop, 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 to make a selection will always result in a normal (non-rectangular) selection even if you hold down the Option key. (If you do the latter, BBEdit will display an alert to inform you that you must turn off soft wrapping in order to make a rectangular selection.) Conversely, if you have made a rectangular selection in a hard wrapped document, the Soft Wrap Text option in the Text Options pop-up menu or sheet will be disabled.

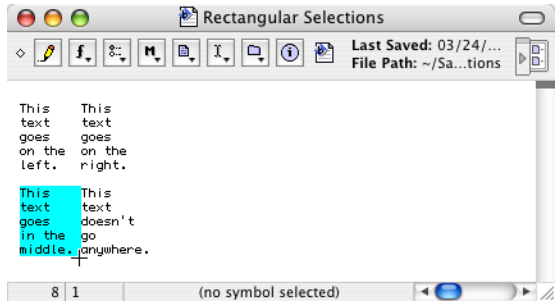
Working with Rectangular Selections

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 BBEdit's ability to manipulate rectangular selections.

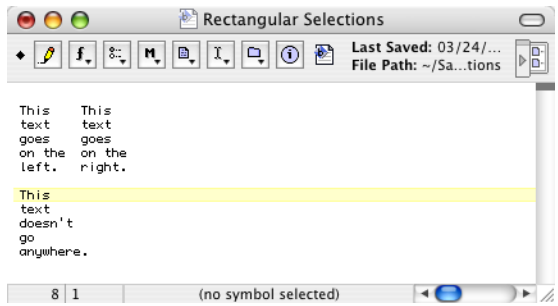
For example, consider you have the document shown below, and you want to move only the bottom left column (the one that says "This text goes in the middle") and move it in between the top left and top right columns. To do this using standard selection methods, you would have to perform five separate 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:



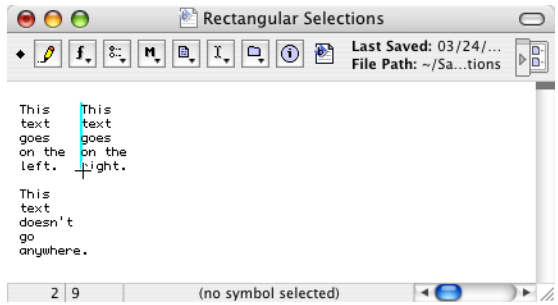
Choose Cut from the Edit menu (or press Cmd-X) to cut the selected text out of the document and place it on the Clipboard.



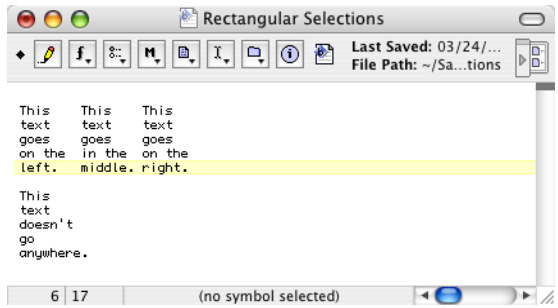
Next, you must paste in the text you just cut. You can do this in either of two ways:

- Use the Paste Column command, which will "paste down" from the current insertion point. This allows you to directly insert text without needing to make a rectangular selection first.
- Make a rectangular selection as described below, and then use the standard Paste command. This procedure is less efficient for moving columnar data than using the Paste Column command, but it allows you to select and replace a region of text as well as simply inserting text.

To manually make a rectangular selection prior to pasting text, 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.



Note Some word processors also provide support for rectangular selections which works a little differently than BBEdit'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. BBEdit does not do this. Instead, when you copy a rectangular selection to the clipboard, BBEdit turns the selection into a series of individual lines, which is why you must make a rectangular selection before pasting, so BBEdit 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 BBEdit 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 BBEdit 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.**

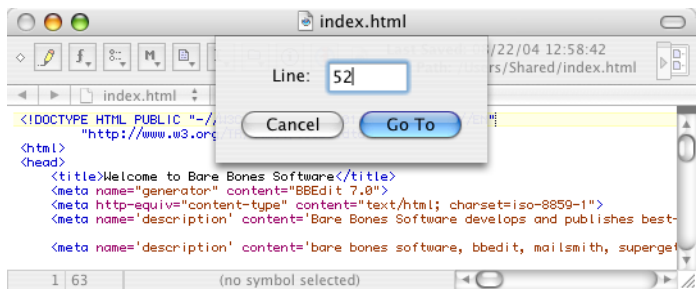
start of line **7**	up **8**	Scroll up **9**
left **4**	show selection **5**	Right **6**
end of line **1**	down **2**	Scroll down **3**

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, BBEdit 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.

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

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.

Resolving URLs

To resolve a URL (Uniform Resource Locator), you can Command-click anywhere in the URL text, or Control-click to bring up the contextual menu and choose Open URL from the menu. BBEdit will examine the URL and launch the appropriate helper application. If the URL is not valid or the helper application cannot be found, BBEdit will beep.

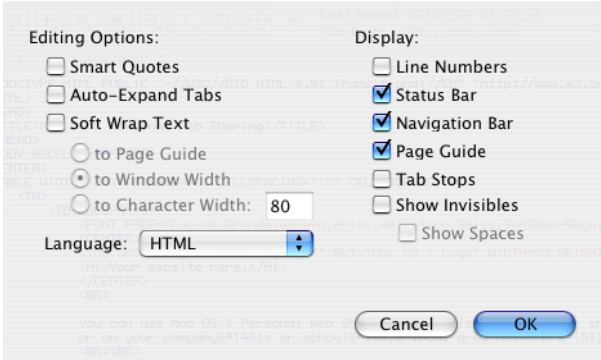
Note Some Web browsers cannot resolve URLs if the request is sent when the browser is starting up. If your Web browser does not properly resolve the URL, wait until the browser has finished starting up and then try again.

Bare Bones Software gratefully acknowledges John Norstad for providing the URL parsing code.

Text Options

You can use the Text Options command to change the way BBEdit edits text and the way it displays text and additional elements in its windows. When you choose this command, BBEdit will drop a Text Options sheet in the current text window.

The controls on the Text Options sheet are divided into two parts: the Editing options on the left control the way BBEdit behaves while you type, and the Display options on the right control the appearance of the BBEdit 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 sheet 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 preference panels.

Editing Options

These options control the way BBEdit 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 preference panel.

Smart Quotes

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

Soft Wrap Text

When this option is selected, BBEdit 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 Text Status Display panel of the Preferences.

See “How BBEdit Wraps Text” later in this chapter to learn how wrapping works in BBEdit.

Language

The Language menu lets you specify which source code language the file uses. The file’s language setting affects how BBEdit performs syntax coloring and parses function names for the function pop-up menu. BBEdit 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). BBEdit uses these symbols:

Symbol	Meaning
∆	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 BBEdit recognizes, BBEdit displays keywords and other language elements in color.

BBEdit 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, BBEdit 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 BBEdit's Languages preferences, BBEdit will attempt to guess what language the file contains and apply the appropriate syntax coloring. If BBEdit 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 sheet, 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 "BBEdit" state, the manual language selection will persist and override any suffix mapping.

By default, BBEdit 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 BBEdit recognizes, and the suffixes or extensions it expects for them, are listed in the Languages preference panel, and suffix mappings can also be changed there. You can choose the colors that BBEdit uses for syntax coloring in the Text Colors preference panel.

Note BBEdit 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 BBEdit how many spaces should occur between tab stops.

IMPORTANT The chosen display style will be used for *all* text in the window; BBEdit 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 BBEdit Wraps Text

BBEdit 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, BBEdit 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 BBEdit 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 sheet
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

Users of very old versions of BBEdit or BBEdit Lite will note that the Wrap while Typing option (which hard-wrapped text automatically by inserting a Return when you reach the right margin) has been relegated to the dustbin of history. It has been superseded by soft wrapping.

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 sheet 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 BBEdit 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

BBEdit will not insert hard line breaks into Unix or DOS-format files upon saving (although versions of BBEdit prior to 4.5 did).

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 BBEdit, 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 sheet 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.

BBEdit 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.

If the current document contains a selection range, Add Line Breaks will affect only the selected text; if there is no selection, this command will affect the entire contents of the current document.

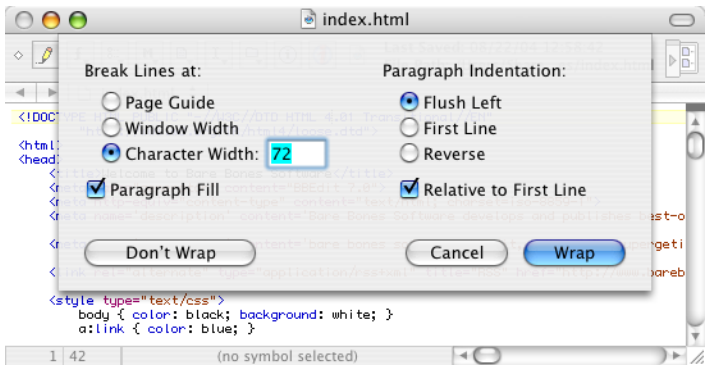
Note The Add Line Breaks command does not turn off soft wrapping.

Hard Wrapping and Filling Text

The Hard Wrap command in the Text menu offers more flexibility for hard-wrapping text than the Add Line Breaks command. Whereas Add Line Breaks merely "freezes" the line breaks displayed in a document by inserting carriage returns, the Hard Wrap command allows you to wrap text to any arbitrary width, while also reflowing or indenting paragraphs.

If the current document contains a selection range, Hard Wrap will affect only the selected text; if there is no selection, this command will affect the entire contents of the current document.

When you choose the Hard Wrap command, BBEdit opens a sheet in the frontmost document:

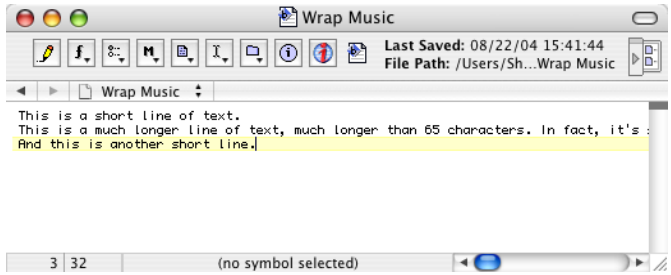


The controls in the left half of the sheet control the maximum width of lines after hard wrapping, and whether wrapped lines should be consolidated to fill paragraphs to the specified width. The controls in the right half determine how paragraphs should be indented.

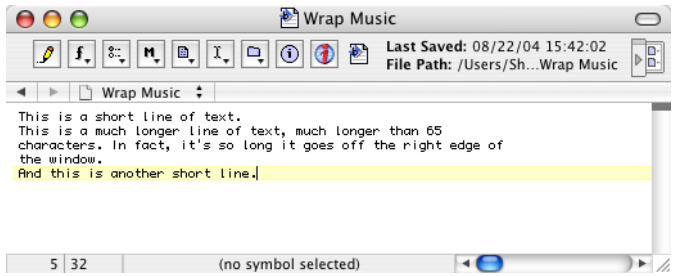
The “Break Lines at” buttons let you specify the wrapping margin.

If the Paragraph Fill option is selected, BBEdit 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:

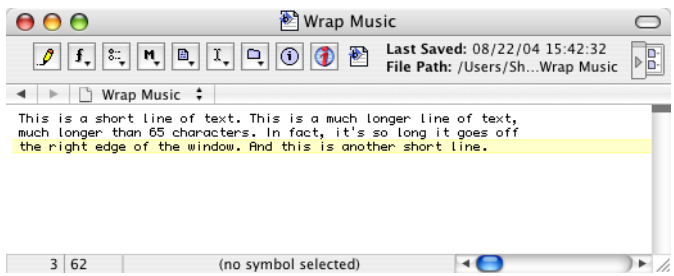


This is what happens when you wrap to 65 characters with Paragraph Fill off:



BBEdit 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:



BBEdit 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, BBEdit 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 BBEdit 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.

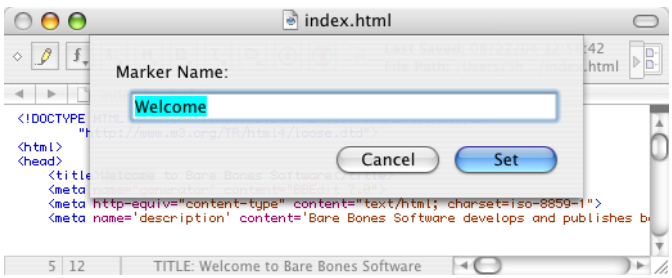
Setting Markers

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.**

BBEdit 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.**

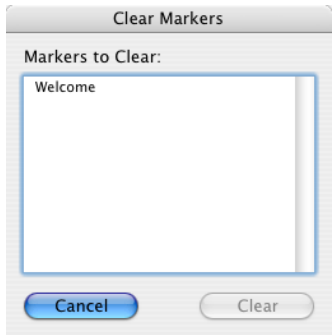
Tip If you hold down the Option key as you choose Set Marker, BBEdit sets the marker using the leading characters of the selected text as the name of the marker, without displaying a dialog box.

Clearing Markers

To clear a marker:

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

BBEdit displays the list of markers.



2 **Select the marker you want to delete.**

3 **Click Clear to clear the marker.**

BBEdit 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.

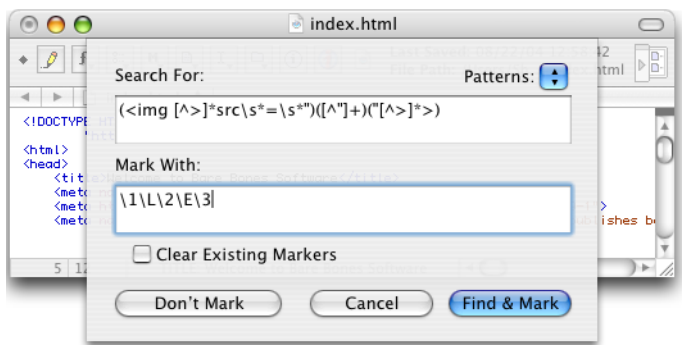
Using Grep to Set Markers

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.**

BBEdit 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.**

BBEdit 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, BBEdit 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. BBEdit 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 BBEdit 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, BBEdit displays a directory dialog box that lets you select a folder to insert. BBEdit 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, BBEdit opens a directory dialog box that lets you select the file or folder. BBEdit 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. BBEdit 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

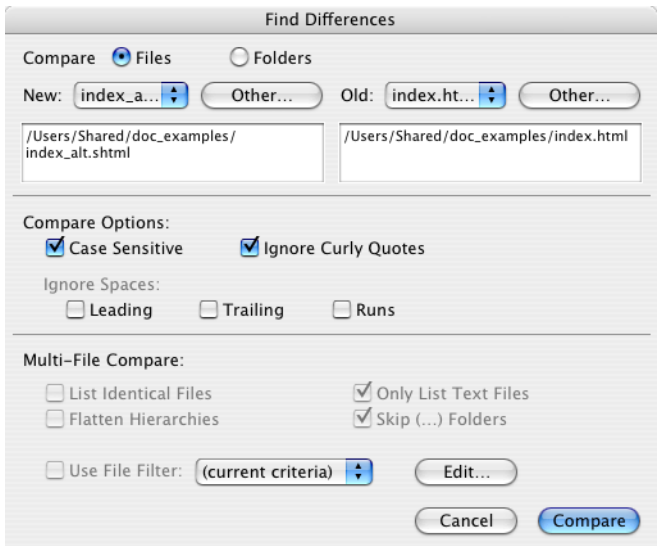
If you have ever had to reconcile changes between two different versions of a file, or even larger numbers of documents, you know how laborious this task can be. BBEdit's Find Differences command is a powerful tool for doing such comparisons faster and more effectively. Using Find Differences, you can 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. (BBEdit 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.**

BBEdit 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, BBEdit distinguishes uppercase from lowercase letters; deselect this option if you want BBEdit to consider uppercase and lowercase letters the same.

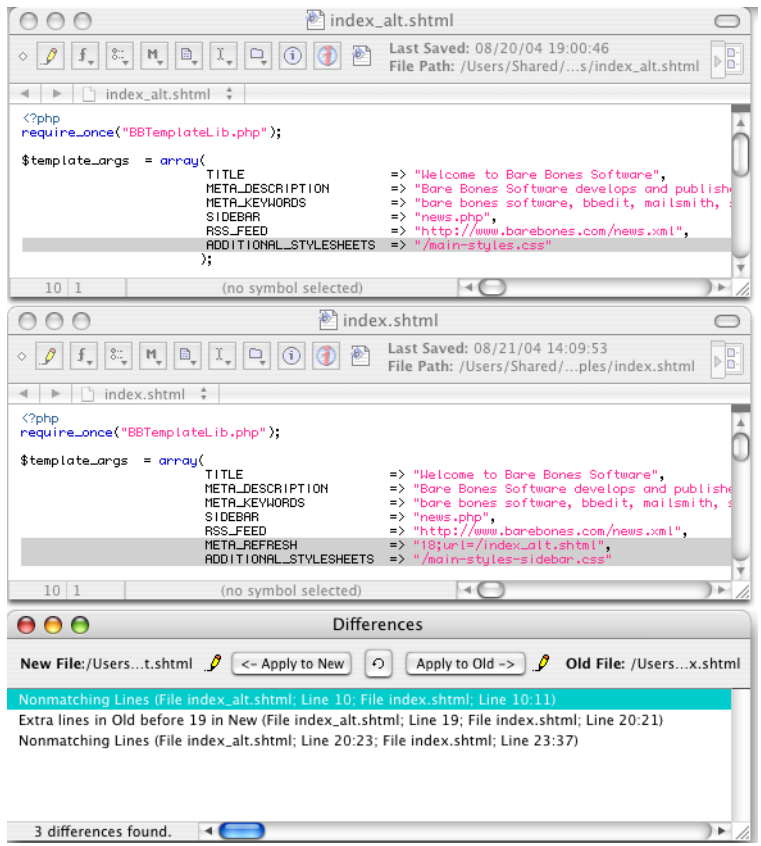
When Ignore Curly Quotes is selected, BBEdit treats typographers' quotes the same as straight quotes.

When one of the Ignore Spaces options is selected, BBEdit 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 'bbdiff' command line tool to specify two files or folders, and have BBEdit perform a Find Differences on them.

If the two files are different, BBEdit 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; BBEdit 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, BBEdit 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; BBEdit 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 BBEdit 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, BBEdit 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, BBEdit 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, BBEdit 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, BBEdit does not list non-text files when comparing folders.

Skip (...) Folders

If this option is set, BBEdit 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, BBEdit 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 Check Spelling command 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. BBEdit will check every word in the document, starting from the top, using the system spelling checker.

When BBEdit 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-;). BBEdit 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 BBEdit will use, and to forget learned spellings. To use the Spelling panel:

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

BBEdit 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.**

BBEdit 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 BBEdit 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 from the Text Menu. See the documentation for your spelling checker for details on how to use it once it has been invoked.

CHAPTER

5

Text Transformations

This chapter describes the range of powerful text transformation commands offered by BBEdit. Beyond applying individual commands to the current document, you can define and save Text Factories, which are sequences of commands that can be applied to one or more documents.

In this chapter

Text Menu Commands

BBEdit 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 BBEdit'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, BBEdit 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

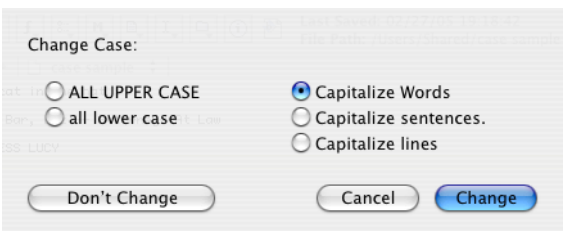
This command (once named Twiddle) swaps two characters according to the following rules:

- 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. 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.
all lower case	Every character changes to lowercase.

This button…	Changes the text like this…
Capitalize Words	The first character of every word changes to uppercase; all other characters change to lowercase.
Capitalize sentences	The first character of every sentence changes to uppercase; all other characters change to lowercase.
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 BBEdit indent or outdent the text by one space instead of one tab stop.

BBEdit 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.

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

Note If you have set custom comment delimiters for HTML in the Languages preference panel, those delimiters will be honored when you use the Un/Comment command. However, they will not affect the operation of the HTML-specific comment commands on the Markup menu.

Hard Wrap

This command wraps long lines by inserting hard line breaks and can reflow (fill) paragraphs if desired. See “How BBEdit Wraps Text” on page 66 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 BBEdit Wraps Text” on page 66 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 BBEdit Wraps Text” on page 66 for more information.

Apply Text Factory

This command presents a submenu listing the text factories stored in BBEdit's application support folder. You can apply a text factory to the current document by choosing it from the submenu. See "Text Factories" on page 91 for more information on creating and using Text Factories.

Apply Text Factory <recent>

This command applies the listed text factory to the current document. If you have not previously applied any text factory during the current launch of the application, this menu item will be disabled until you do so.

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 ‘ ’).

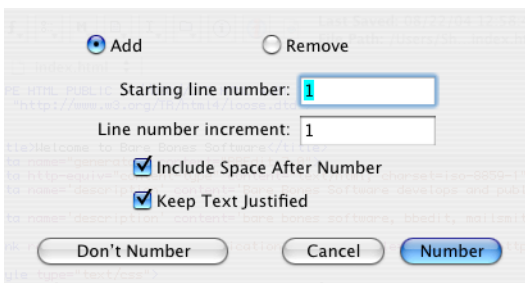
Note You should not use this plug-in to prepare text for posting on a web page or use in an email, as typographer's quotes in the Mac character set will generally not be properly displayed by applications on other platforms.

Straighten Quotes

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

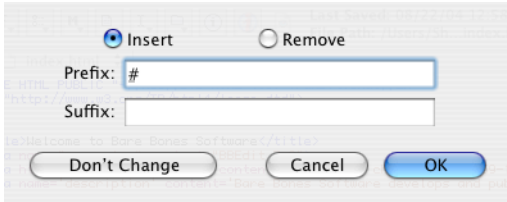
Add/Remove Line Numbers

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 add or remove trailing spaces, and 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.

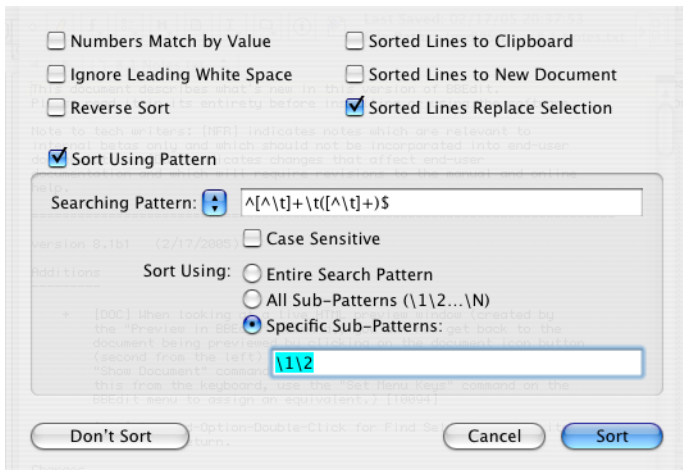


If you define both a prefix and a suffix string, BBEdit will apply them to the text at the same time.

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.

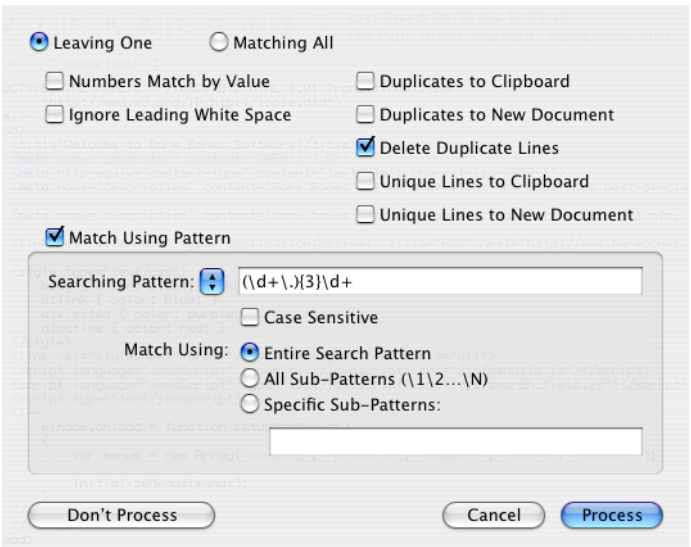
By checking the Sort Using Pattern option, you can specify a grep pattern to further filter the lines to be sorted. If the pattern contains subpatterns, you can use them to control the sort order based on the contents of the strings they match. When you sort using a grep pattern, the Case Sensitive option controls the case sensitivity of the pattern match in the same manner 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 matches one or more sets of multiple lines within in the document, but does not match any individual lines, BBEdit will not sort the contents of the document.

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.

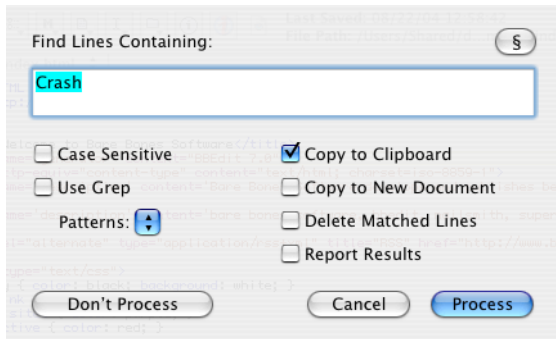
IMPORTANT When you use a grep pattern with this command, matches are not automatically anchored to line boundaries, so ambiguous patterns may produce unpredictable results. To avoid this problem, you should use the line start ^ and line end operators as necessary.

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 BBEdit 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.

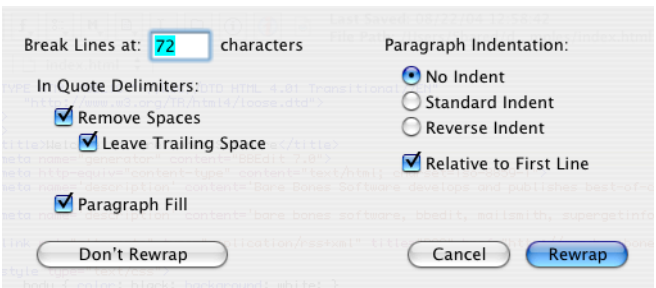
- Setting Copy to Clipboard on and Delete Matched Lines off is equivalent to applying the Copy command.
- Setting Copy to Clipboard off and Delete Matched Lines on is equivalent to applying the Clear command.

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 BBEdit 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, BBEdit 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. BBEdit 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 83), and reinserts the quote markers.

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

Increase and Decrease Quote Level

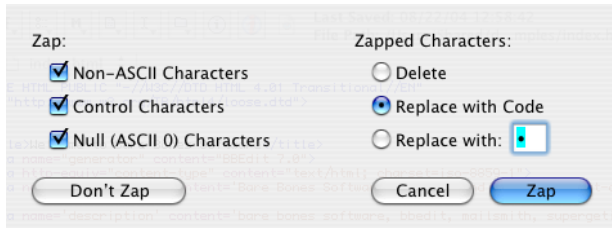
This command inserts or deletes a standard Internet quote character (">") from the selected hard-wrapped text, or for the current line if there is no selection.

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 111.)

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

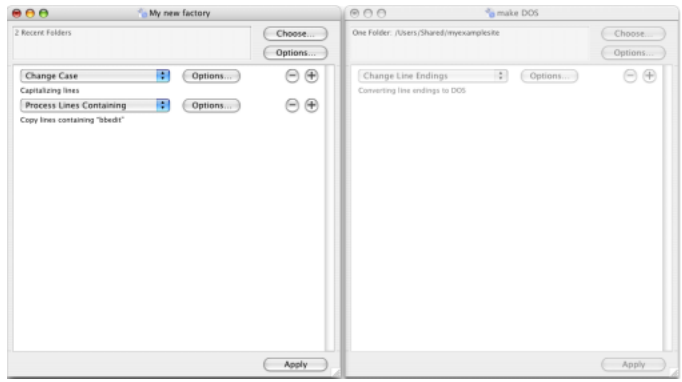
This command converts a document containing mixed line endings to have a uniform set of line endings.

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 BBEdit's line-break translation will suffice when you next open it.

Text Factories

A text factory document enables you to apply BBEdit'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.



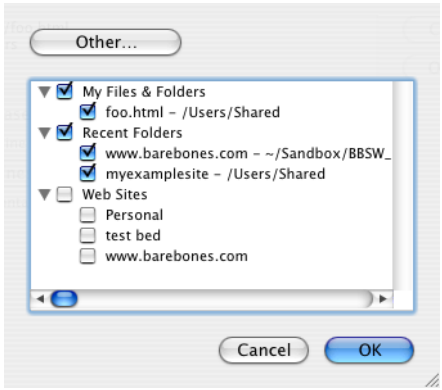
Creating and Configuring Text Factories

To create a new text factory, choose the Text Factory command from the New submenu of the File menu. BBEdit will create a text factory document window which contains a single action step (corresponding to a single operation).

Note Text factory documents are plist text files, which are distinguished from editable text files by having 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.

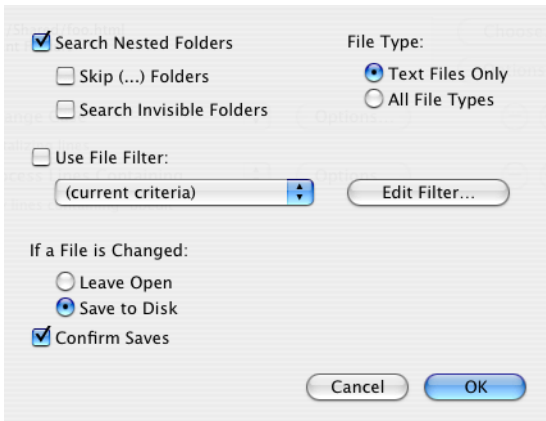
Choosing Targets

At the top of a text factory window is a summary area which displays information about the target files and folders you have chosen for processing. 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 117) and is in fact applied *before* the filter.

The last group of options controls how BBEdit treats processed documents. Choose Leave Open to have BBEdit leave all the documents open so that you can inspect the results of the operation. Choose Save to Disk to have BBEdit 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 BBEdit saves them to disk. You should not turn this off unless you are sure that the processing being applied is what you want.

Defining Actions

Each action in a text factory contains the following elements:

- a popup menu from which you select the operation to apply. You are not limited to a single use of an operation; for example, you can apply multiple Replace All operations in a single factory.
- an Options... button, which is used to configure any settable options for the command. If the command has no settable options, this button is disabled.
- Add ("plus") and Remove ("minus") buttons. Clicking the Remove button in an action will remove that action; clicking the Add button will insert a new action after the selected action. Hold down the Option key while clicking the Add button to create a new action which duplicates the action next to that button. If there is only one action in the factory, its Remove button is disabled.
- a summary line which describes the chosen command and any parameters.

Note It is up to you to make sure that your actions do not work at cross purposes. For example, it would not be useful to have an "Educate Quotes" operation followed immediately by a "Straighten Quotes" operation.

You can re-order actions by clicking the line containing an action (in any place not occupied by a button or pop-up menu) and dragging it to a new location.

Each operation available on an action's pop-menu behaves similarly to its counterpart command on the Text menu. When choosing and configuring operations, you should keep the following differences in mind.

- Change Line Endings and Change Text Encoding will only affect the line endings and text encoding of the file that BBEdit writes out when you choose Save to Disk from the factory's Options sheet. Neither operation changes any file contents in memory, so they will have no visible effect if the document is left open (either by choosing the Leave Open button in the Options sheet, or by opting to leave the document open when confirming a save).
- When you use the Run AppleScript Filter operation, your script should be written with an entry point named "ApplyTextTransform". The input parameter to this entry point is a Unicode string containing the file's contents. This entry point should return the file's contents as a Unicode string (or something which can be directly coerced to one):

```
on ApplyTextTransform (fileData

-- do something to fileData

return fileData -- or some reasonable facsimile thereof

end
```

- When you use the "Run Unix Filter" operation, you can choose to have BBEdit convert the file's contents and the output to UTF-8 by setting the "Use UTF-8 for I/O" option. If you do not set this option, BBEdit will use the system encoding instead.

Note As a point of technical interest, your program will be run on a secondary thread. This should not pose a problem as long as your Unix script avoids doing fancy GUI-ish things.)

Applying a Text Factory

Once you have configured your factory and selected the files and folders to process, click the "Apply" button in the lower right-hand corner of the window. BBEdit will apply the actions in the order you've specified, to each file in the target set. This processing happens in the background, so you can keep using BBEdit while it's going on (similar to a multi-file search operation).

You can also access and use stored text factories by several other means:

- the Apply Text Factory commands on the Text menu (applies the selected text factory to the current document)
- the Text Factory menu (runs the selected text factory against its configured sources)
- the Text Factories palette (applies the selected text factory to the current document)

Automator Actions

Automator is a feature of Mac OS X which enables you to create and reuse actions to easily perform common or time-consuming tasks. Apple's web site offers a good overview of Automator.

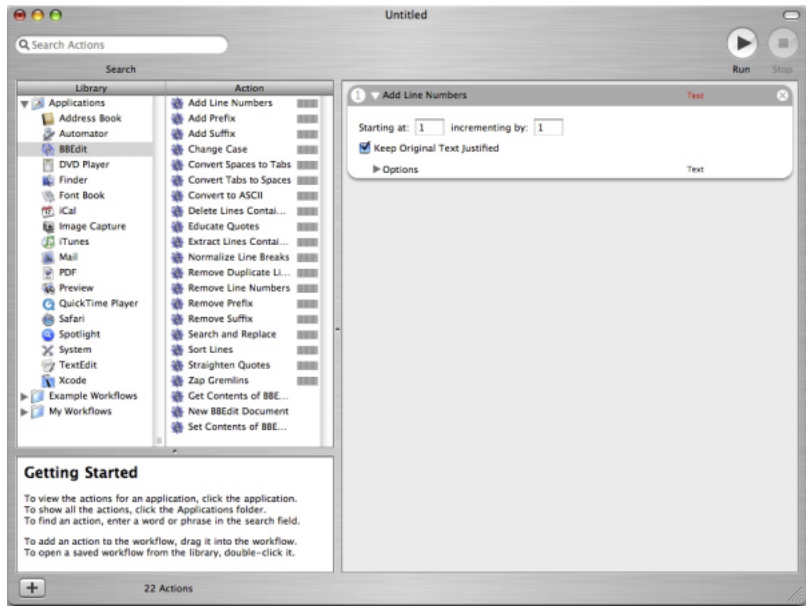
`http://www.apple.com/macosx/features/automator/`

In order to use Automator, you must have BBEdit 8.2 or later, and be running Mac OS X 10.4 or later.

Using BBEdit with Automator

BBEdit offers Automator actions which correspond to most of the text transformation commands on the Text menu. In order to use any of BBEdit's actions, BBEdit must be running.

After you launch Automator, it will create an Untitled workflow as shown below. Choose any application in the Library section to see what actions it offers. To add an action to your workflow document, you can double-click on the action, or drag it into the right-hand panel. After you add an action to your workflow, you can set options (if any) in the action's tile. To expand or collapse an action tile, click the triangle next to the action's name.



The Automator actions which BBEdit offers allow you to make use of BBEdit's unique transformations in the context of a larger Automator workflow. Please keep in mind that BBEdit currently does not include any actions which support working with files, as it's expected that you will be applying BBEdit actions to work on text acquired from other sources in the workflow. If you only need to apply BBEdit's text transformations to files or sets of files, you would be better served by using a Text Factory.

Available Actions

Add Line Numbers

This action will prepend a line number to each line processed.

Input: (Text)
Result: (Text) Text with leading line numbers added.
Related Actions: Remove Line Numbers

Add Prefix

This action will prepend the supplied string to each line processed.

Input: (Text)
Result: (Text) Text with the specified string prepended to each line.
Related Actions: Add Suffix, Remove Suffix, Remove Prefix

Add Suffix

This action will append the supplied string to each line processed.

Input: (Text)
Result: (Text) Text with the specified string prepended to each line
Related Actions: Remove Suffix, Remove Prefix

Change Case

This action will change the case of the provided text according to the specified rules.

Input: (Text)
Result: (Text) Text with the case transformation applied.

Convert Spaces to Tabs

This action will convert each run of N space characters into a tab character.

Input: (Text)
Result: (Text) Text with runs of space characters converted to tabs.

Convert Tabs to Spaces

This action will convert each tab character into the specified number of spaces.

Input: (Text)
Result: (Text) Text with tab characters converted to runs of spaces.

Convert to ASCII

This action will convert the input text into an ASCII-only representation.

Input: (Text)
Result: (Text) Text with non-ASCII characters “translated” into ASCII

Delete Lines Containing

This action will return all lines in the input string which do not the specified string.

Input: (Text)
Result: (Text) Text with the complement of the matching lines.
Related Actions: Extract Lines Containing

Educate Quotes
This action will convert ASCII quotes into their typographer equivalent.

Input: (Text)
Result: (Text) Text with ASCII quotes converted to typographer's quotes
Related Actions: Straighten Quotes

Extract Lines Containing
This action will return all lines in the input string which contain the specified string.

Input: (Text)
Result: (Text) Text with the matching lines.
Related Actions: Delete Lines Containing

Get Contents of BBEdit Document
This action will get the contents of the frontmost text document in BBEdit according to the options you set.

Requires: A text document must be open.
Result: (Text) A BBEdit document object
Related Actions: Set Contents of BBEdit Document, New BBEdit Document

New BBEdit Document
This action creates a new BBEdit document from the input text.

Input: (Text)
Result: (com.barebones.bbedit.document-object) A BBEdit document object
Related Actions: Get Contents of BBEdit Document, Set Contents of BBEdit Document

Normalize Line Breaks
This action will change all line breaks into the format specified.

Input: (Text)
Result: (Text) Text with the chosen line breaks.

Remove Duplicate Lines
This action will eliminate duplicates from the provided lines according to the specified rules.

Input: (Text)
Result: (Text) Unique Lines of Text
Related Actions: Sort Lines

Remove Line Numbers
This action will remove leading line numbers from each line processed.

Input: (Text)
Result: (Text) Text with leading line numbers removed.
Related Actions: Add Line Numbers

Remove Prefix
This action will remove the supplied string from the beginning of each line processed.

Input: (Text)
Result: (Text) Text with the specified prefix removed from each line.

Related Actions: Add Suffix, Remove Suffix

Remove Suffix

This action will remove the supplied string from the end of each line processed.

Input: (Text)
Result: (Text) Text with the specified suffix removed from each line.
Related Actions: Add Suffix, Remove Prefix

Search and Replace

This action will search the input string, and replace all matches with specified string.

Input: (Text)
Result: (Text) Text with the specified replacements done.

Set Contents of BBEdit Document

This action will replace or insert the passed text into the frontmost text document in BBEdit according to the options you set.

Requires: A text document must be open.
Input: (Text)
Result: (com.barebones.bbedit.document-object) A BBEdit document object
Related Actions: Get Contents of BBEdit Document, New BBEdit Document

Sort Lines

This action will sort the provided lines according to the specified rules.

Input: (Text)
Result: (Text) Sorted Lines of Text.
Related Actions: Remove Duplicate Lines

Straighten Quotes

This action will convert typographer quotes into their ASCII equivalent.

Input: (Text)
Result: (Text) Text with typographer's quotes converted to ASCII quotes
Related Actions: Educate Quotes

Zap Gremlins

This action will convert each non-ASCII character as specified in the configuration.

Input: (Text)
Result: (Text) Text with non-ASCII characters removed.
Related Actions: Convert to ASCII

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 BBEdit's extensive set of tool and utility palettes.

In this chapter

Window Menu

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

BBEdit 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 BBEdit windows to the front.

Palettes

The Palettes submenu provides quick access to all of BBEdit'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.

Glossary

BBEdit's powerful Glossary provides an easy way to store and access frequently used text of any sort. For details on using the Glossary, including its language-sensitive mode, please refer to Chapter 12.

Plug-In Tools

The Tools palette displays a list of all the plug-ins in your BBEdit 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 15, "BBEdit Plug-Ins," for more information on installing plug-ins.

Scripts

The Scripts palette displays all the currently installed AppleScripts in your BBEdit Scripts folder. See Chapter 13, "Scripting BBEdit," for information about using AppleScript with BBEdit.

Stationery

The Stationery List is a palette that displays all the stationery pads you have placed inside the Stationery folder of BBEdit'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 BBEdit. 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)Web template" would sort before "01)HTML Template.") For such files, the first three characters are not displayed in BBEdit. 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 Glossary, 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 factories currently available in BBEdit's application support folder. For more information, see "Text Factories" on page 91.

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 preference panel (see Chapter 10). You can open a file by dragging its icon from the Finder or from a file group window into the Windows window.

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).

HTML Markup Tools

The main HTML Markup Tools palette is a comprehensive listing of BBEdit's numerous HTML markup commands. See "HTML Tools Palette" on page 240 in Chapter 11 for details on what these commands do. You can choose which commands appear on the main HTML Tools Palette in the HTML Palette panel of the Preferences window.

Several other HTML palettes are available, each with a specific focus. These include Block, CSS, Entities, Font Style, Forms, Inline, Phrase, Tables, Utilities, and Web Safe Colors. For more information on these tools, please see Chapter 11, "BBEdit HTML Tools."

Unix Scripting Tools, Unix Filters, and Unix Scripts

BBEdit 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 14, "Working with Development Tools."

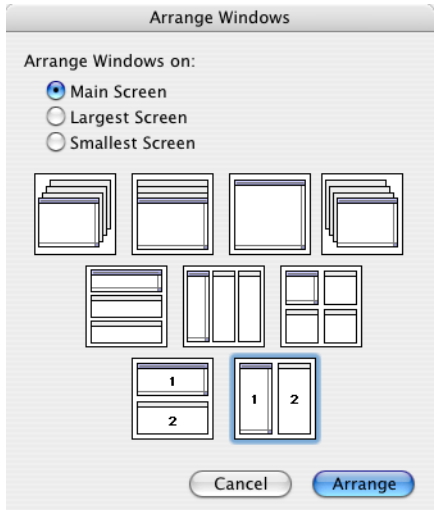
Workspace

The Workspace submenu gives you the ability to save and reopen sets of palettes. You can choose the Save Workspace command to create a named workspace, the Delete Workspace command to delete a named workspace. If you choose a named workspace from this submenu, BBEdit will restore all persistent windows (all palettes, and the Preferences window) to the visibility state and positions they had when the workspace was created.

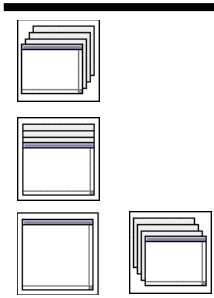
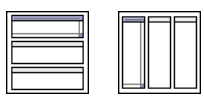
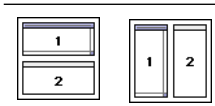
For example, you might create an "HTML" workspace with the HTML Tools and Windows palettes open, and a "Perl" workspace with the Unix Scripts and Unix Filters palettes open. Then, starting with only the Windows palette displayed, when you choose the "HTML" workspace, BBEdit will open the HTML Tools palette and position both palettes according to their saved locations). If you then choose the "Perl" workspace, BBEdit will close both these palettes, and open the Unix Scripts and Unix Filters palettes.

Arrange

The Arrange command gives you several ways to organize text windows and shell worksheets. This command has no effect on any other types of windows, such as browsers or file groups. When you choose the Arrange command, BBEdit opens the Arrange Windows dialog box.



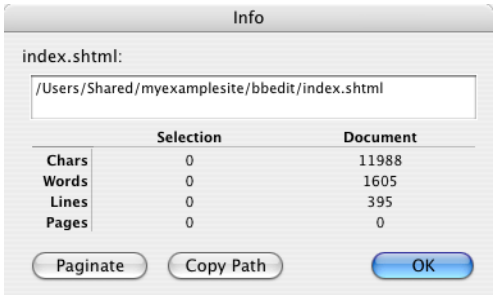
The radio buttons at the top of the dialog specify which screen the windows will be arranged on. You can choose the main screen, the largest screen, or the smallest screen.

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, BBEdit leaves room along the right side or bottom of the screen for Finder icons. BBEdit 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.
	BBEdit tiles the windows in two or three rows (or columns). If you have more than three windows open, BBEdit stacks additional windows behind the three front windows.
	BBEdit figures out how many rows and columns it needs to tile windows. The larger your screen, the more rows and columns BBEdit uses. The windows are never narrower than half of a classic Macintosh screen.
	BBEdit 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.

Get Info

The Get Info command displays 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.

Zoom

There is no longer a Zoom command in the Window menu, but the key equivalent Command-/ (which users of earlier versions of BBEdit may be accustomed to) still works. Zoom will 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, BBEdit 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 preference panel controls this behavior; turning this switch off will revert the behavior to be the same as previous versions of BBEdit.

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 palette to quickly select any open window.

CHAPTER

7

Searching

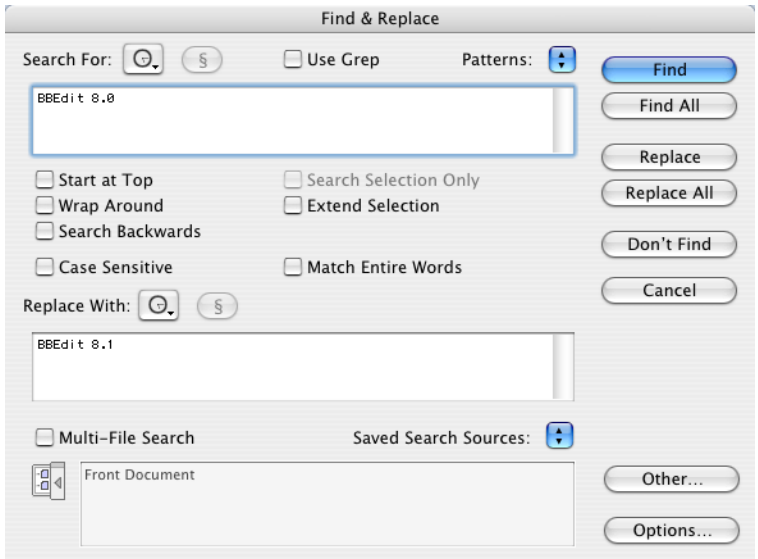
This chapter describes BBEdit's powerful Find command, now enhanced with a flexible file filtering mechanism. It tells you how to search for text in the active window or within a set of files. BBEdit can also do advanced pattern, or grep, searching. To learn about pattern searching, you should read this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

BBEdit 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.**

BBEdit 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-.

BBEdit closes the Find & Replace dialog and then selects the search string in the active window if the search was successful. Once BBEdit 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 BBEdit 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, BBEdit treats the search and replace strings as grep patterns. Otherwise, BBEdit 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 127.

Start at Top
When this checkbox is selected, BBEdit 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, BBEdit 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, BBEdit 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, BBEdit searches from the insertion point to the beginning of the document. Otherwise, BBEdit 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, BBEdit searches only the selected text. Otherwise, BBEdit searches the entire document. (Unavailable if Wrap Around or Start at Top is checked.)

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

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

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

Multi-File Search
When this checkbox is selected, BBEdit 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 BBEdit, 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. BBEdit gives you a great deal of flexibility in how to do this. You can search all the files in a given folder or defined web site, in a file group, in open editing windows, or in an existing search results browser. For even greater control, you can select a diverse set of search sources, or apply BBEdit's advanced multi-criteria file filtering option.

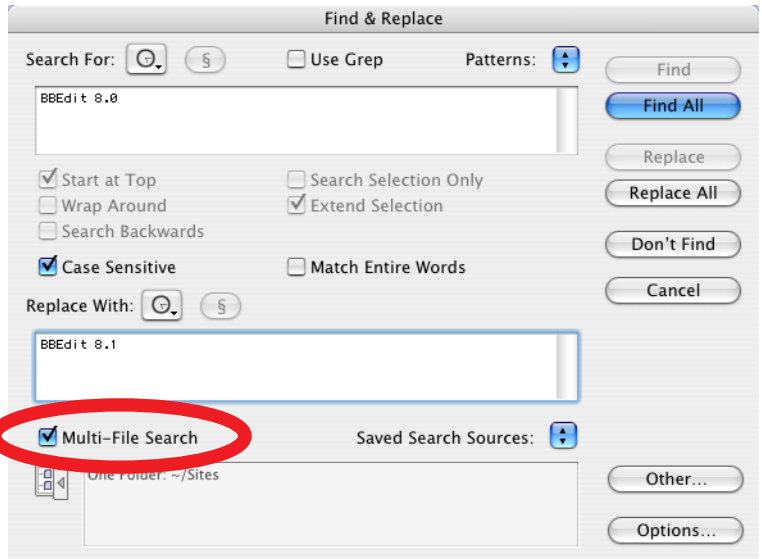
IMPORTANT Multi-file searching is no longer a modal operation. When you start a search, BBEdit will display a search progress window and return control, so that you can continue working. You can perform more than one multi-file searches at a time; each search will have its own progress window. Closing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

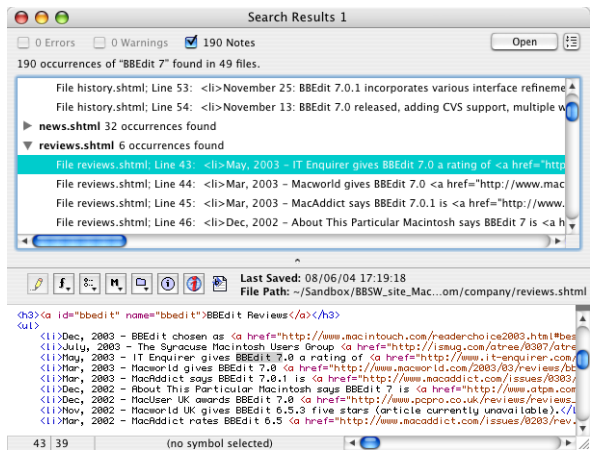
8 **Click one of the buttons along the right side of the dialog box to begin the search.**

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. BBEdit 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, BBEdit 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 BBEdit 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 BBEdit, 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.

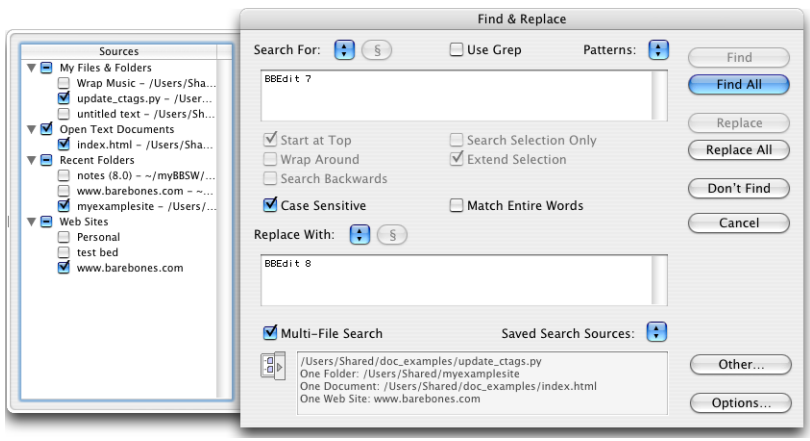
To see the contents of a file, click a line in the list of matched searches. The part of the file that contains the search string appears in the lower pane. You cannot edit text in a search results browser. To edit a file, double-click the line that contains the match you want to edit, and the specified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

- specified individual files
- the files in any selected or recently-searched folder
- open text documents
- the files listed in any results browser (such as a search results browser, an HTML syntax errors browser, or a compile errors browser)

- the files and folders contained in a file group
- the files in a defined web site folder

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. BBEdit 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.

Searching the files in a file group

If the files you are working with are all listed in a BBEdit file group, you can choose that group in order to search the files. To chose a file group, click the box next to that group's name, or double-click on its name, in the Sources drawer.

Searching the files on a web site

You can limit your multi-file search to the files for a specific web site folder as defined in the HTML Web Sites preference panel.

This menu lists all of the local site root and "Templates & Includes" folders that you have specified for the web sites in your HTML Web Sites preference panel. If you have not designated such a folder for a site, the site name appears dimmed in the menu and cannot be selected.

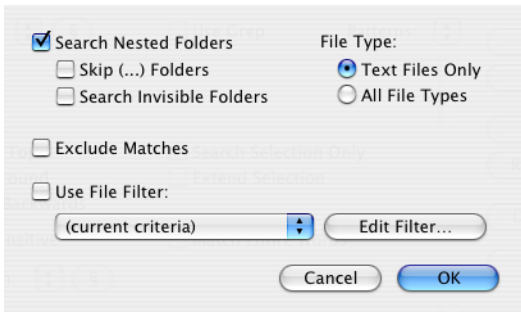
Note The Choose a Folder dialog will display any packages it encounters as folders (rather than just as single files, the way they appear in the Finder). This allows you to navigate their internal structure just as you would any other folder. Similarly, you can drag a package from the Finder into the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

Saved Search Sources

You can use the Saved Search Sources pop-up menu to store specific sets of search sources for later reuse. To save a set of search sources, choose Remember this Set from the pop-up menu and give the set a name in the resulting dialog. To select a saved set of search sources, choose that set's name from the pop-menu.

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.

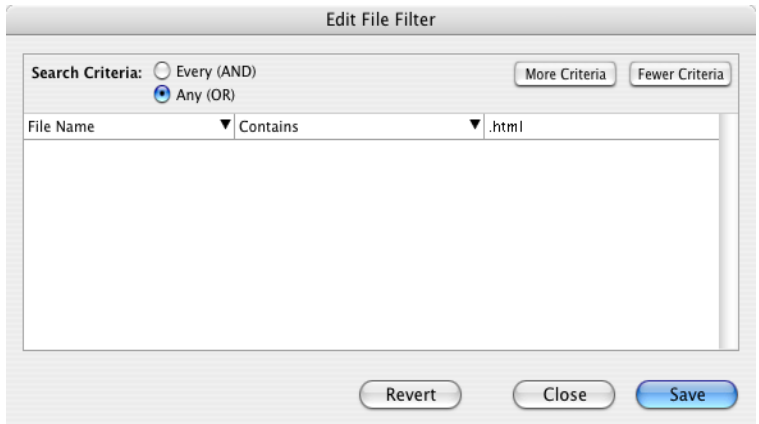
You can further restrict which files from the chosen sources will be searched by applying a file filter. See "File Filters" (below) for more details.

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 BBEdit and not some other program, or those that are HTML or 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

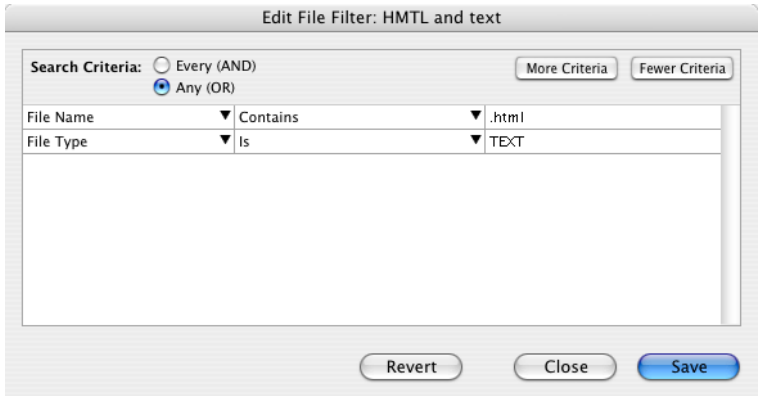
To define a new saved file filter, select New Filter from the pop-up menu. BBEdit will ask you for a filter name; then the Edit File Filter dialog, below, appears. You can also define new file filters in the File Filters preferences panel (see page 174).



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. BBEdit 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. BBEdit 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. BBEdit 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, BBEdit 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, BBEdit 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!

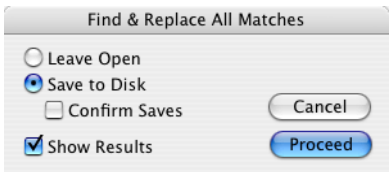
To do a multi-file search and replace, replacing all occurrences:

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.**

BBEdit 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, BBEdit 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 BBEdit 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.

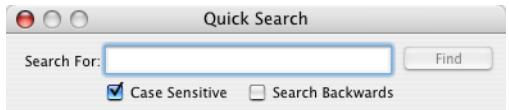
Quick Search

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, BBEdit 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. When you invoke this command, BBEdit will add the current search string to the Find dialog's pop-up menu of recently-used search strings.

Tip Hold down the Shift key to search in the opposite direction from the current search direction, just as for Find Again. You can also hold down the Option and Command keys as you double-click on a selection to search for the next occurrence of the selected text.

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. When you invoke this command, BBEdit will add the current search string to the Find dialog's pop-up menu of recently-used search strings.

Enter Replace String

Choose the Enter Replace 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 search string. When you invoke this command, BBEdit will add the current replace string to the Find dialog's pop-up menu of recently-used replace strings.

Enter Search Pattern

Choose the Enter Replace String command to enter the currently selected text into the Find & Replace dialog as the search pattern and set the Use Grep search option (without opening the dialog). This command does not perform a search, but only sets the contents of the search pattern and the specified search option. When you invoke this command, BBEdit will add the current search pattern to the Find dialog's pop-up menu of recently-used search strings.

Enter Replace Pattern

Choose the Enter Replace String command to enter the currently selected text into the Find & Replace dialog as the replace pattern and set the Use Grep search option (without opening the dialog). This command does not perform a search, but only sets the contents of the replace pattern and the specified search option. When you invoke this command, BBEdit will add the current replace pattern to the Find dialog's pop-up menu of recently-used replace strings.

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, BBEdit 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 Previous/Next Error

If an error browser is open, this command will open the listed error which came before or after the selected error. See Chapter 9 for more information on error browsers.

Go to Previous/Next Placeholder

When you apply a glossary item that contains multiple #INSERTION# cookies, the second and subsequent cookies are replaced with a placeholder string. You can use the Go To Previous Placeholder and Go to Next Placeholder commands to jump back and forth between these special strings from the keyboard. For example, you might use this command when filling in the parameters of a function call, or a series of tag attributes. For additional details, see "Selection and Insertion Placeholders" on page 249.

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 Definition

Looks up definitions for the selected word using ctags information if available. If there is no selection, BBEdit will attempt to determine the symbol name by inspection of the text around the insertion point, rather than requiring you to type a name. (See "Exuberant Ctags" in Chapter 14 for more details about working with ctags.)

Find in Reference

Performs a search for the selected symbol on the Apple Developer Connection web site. As for Find Definition, if there is no selection, BBEdit will attempt to determine the symbol name by inspection around the insertion point.

If for some reason you wish to modify the URL template which BBEdit uses to perform the lookup, you may do so by issuing the following 'defaults write' command in the Terminal:

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

In the template, use "%@" to indicate where the selected symbol name should be placed in the lookup string. For example, to make Find in Reference use Google instead, use:

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

CHAPTER

8

Searching with Grep

This chapter describes the Grep option in BBEdit'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, BBEdit's grep capabilities can make many editing tasks quicker and easier, whether you are modifying Web pages, extracting data from a file, or just rearranging 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 BBEdit 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 BBEdit. For this reason, you will often see regular expressions called "grep patterns," as BBEdit 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 BBEdit'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.

BBEdit-Talk

The BBEdit-Talk online mailing list covers a wide range of topics and questions about using BBEdit, 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
```

Tech Note BBEdit'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 BBEdit'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 BBEdit'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. BBEdit's grep syntax coloring helps make this clear.

Note When passing grep patterns to BBEdit 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 BBEdit, 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: BBEdit-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.

Note ^ and $ do not match the positions after and before soft line breaks.

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.

WARNING In versions of BBEdit prior to 6.5, the # character was a wildcard that matched any digit (0–9); however, this is no longer the case. If you have grep patterns written with an old version of BBEdit that use # for this purpose, you will need to change these patterns. The easiest way to do this is to use the \d character class, which has the exact same meaning that # used to—it matches any character from 0-9. Character classes are explained in the next section.

Other Positional Assertions

BBEdit'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 foo bar`
Will match: `foo bar`
Will not match: `foobar`

Search for: `\bJane\b`
Will match: `Jane's`
Will match: `Tell Jane about the monkey.`

Search for: `\Afoo`
Will match: `foobar`
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", BBEdit 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, BBEdit 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

BBEdit 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 BBEdit 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

IMPORTANT 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

IMPORTANT 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 BBEdit 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>BBEdit!</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 BBEdit 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 BBEdit'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 BBEdit 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

BBEdit'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 BBEdit. 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

The grep engine used in versions of BBEdit prior to 6.5 was unable to search text that contained null characters (ASCII value zero), but this limitation has since been removed. Here's one way to match a null:

```
\x{0}
```

Backreferences

The following charts explain the rules BBEdit 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

BBEdit now 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 BBEdit 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. BBEdit now 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

BBEdit’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 BBEdit, 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 parentheses
(?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: foobar

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, BBEdit 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, BBEdit’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: `abc` *or* `aBc`

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: `SATURDAY` *or* `Saturday` *or* `SUNday` (and so on)

Search for: `(?i:foo)(?-i:bar)`
Will match: `foobar` *or* `FOObar`
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:

```
aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa
```

it takes a long time before reporting failure. So long, in fact, that it will effectively "freeze" BBEdit. 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, BBEdit'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:

```
(aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa()
```

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.

CHAPTER

9 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 BBEdit browser.

In this chapter

Browser Overview

All BBEdit 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 BBEdit can open
Search results	File and line number of each match
Error results (or) general results	File, line number, and status message for each condition

You can open both files and folders from the list pane. When you double-click a folder name, BBEdit replaces the file list pane with the contents of the folder. When you double-click a file name, BBEdit opens the file in an editing window. If the file list pane also included a line number, BBEdit 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 error browsers, checkboxes allow you to hide or show all errors, warnings, or notes.

For results browsers, BBEdit can either show an error hierarchy (where all the errors 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 error encountered 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 Info buttons—should already be familiar to you, since they appear on BBEdit document windows by default. See "Window Anatomy" in Chapter 4 for an explanation of these standard BBEdit functions.

Text View Pane

When you click an item in the list pane, BBEdit displays its contents in the text view pane. If you click a folder, BBEdit lists the names of the files in the folder. If you click a file name, BBEdit 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."

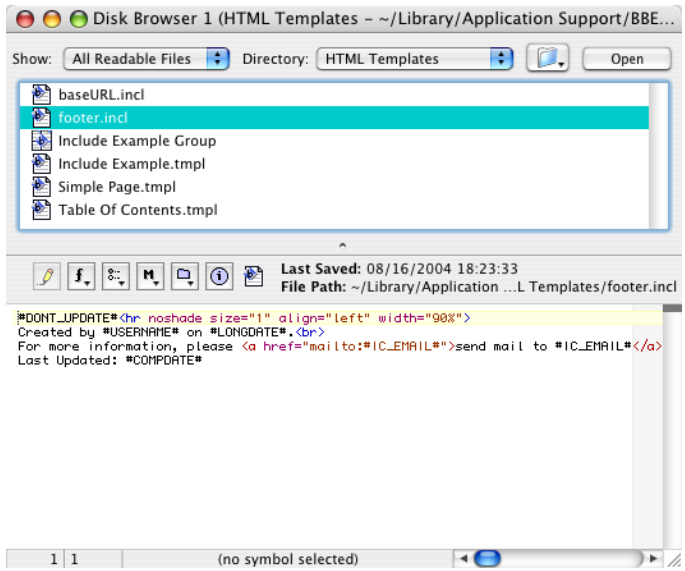
Splitter

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. BBEdit opens a new disk browser that starts in your home directory, but you can navigate to any desired location:



The name and path of the directory currently being viewed are displayed in the title bar of the window. The file list pane displays all the items in the current folder. Click on a file in the file list pane to display its contents in the text pane, or double-click to open the file into a text window. You can also click on a folder to display a listing of its contents in the text pane.

Note You can open a disk browser starting at any particular folder, by dragging that folder onto BBEdit's icon in the Dock or 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 BBEdit to list in the browser. You can select "All Readable Files," meaning all types of files BBEdit 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, QuickTime images, BBEdit file groups, and shell worksheets.

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).

Recently Used Folders

This pop-up menu allows you to quickly go back to any recently-visited folder. You can also choose the Other… option to get a folder selection dialog from which you can choose any folder on any currently mounted volume.

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, BBEdit displays the contents of that folder in the file list pane. If you click one or more files, BBEdit 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 instead of opening the file with BBEdit.

Using the List Pane in Disk Browsers

The list pane of a disk browser displays disks, files, and folders. When you are at the computer level, the list shows all mounted volumes.

When you click a folder or disk, BBEdit 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, BBEdit 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

Note When the list pane has input focus, the browser window's AppleScript "selection" property will return a list of the files currently selected. See "Getting and Setting Properties" on page 270 for further details.

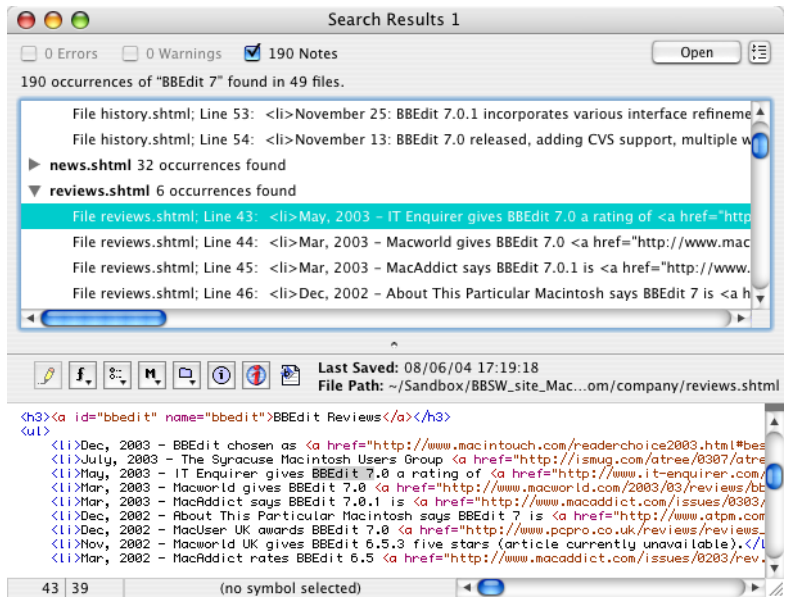
Using the Text Pane in Disk Browsers

When you select a folder or disk in the file list pane, BBEdit 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, BBEdit displays the contents of the file in the text pane if the file is of a type that BBEdit 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, BBEdit 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 BBEdit 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 BBEdit, 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.

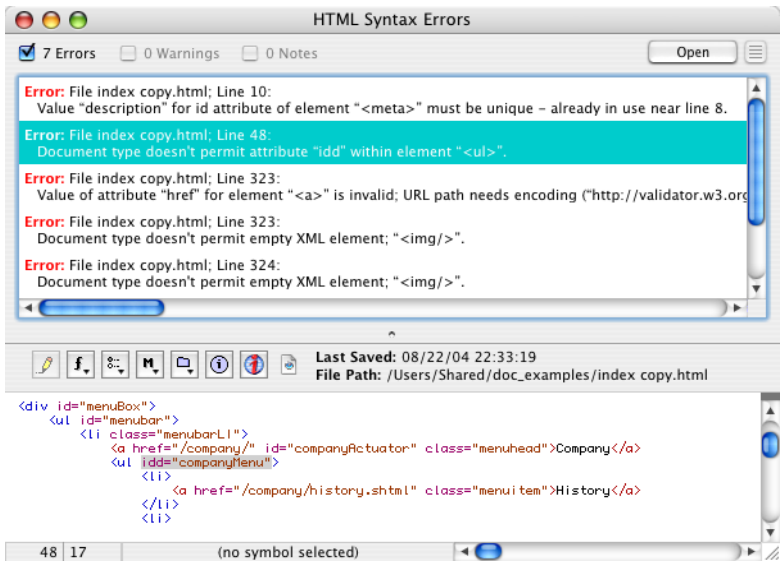
The Open button opens the selected items using BBEdit. To open the selected items using the Finder, hold down the Option key while clicking the Open button.

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.

Error Results Browsers

When you use the HTML syntax checker, link checker, or update tool, BBEdit will open an error results browser to display any errors generated by the command. BBEdit will also open an error results browser to list compilation errors generated by commands issued to CodeWarrior from the Compiler menu, or errors generated by Perl scripts.



Each entry in the list pane corresponds to an error, warning, or note. You can use the checkboxes for each type of item to suppress or display the associated results as desired.

If you click on a entry in the file list, BBEdit displays the location of the error. If you double-click an item in the file list, BBEdit opens the associated file into an editing window and selects the section of text related to the error.

CHAPTER

10 Preferences

You can use the Preferences command to customize much of BBEdit's behavior. You can decide which windows are open when you launch BBEdit, set the default options for windows, set the default options for searches, and so on. This chapter describes BBEdit's extensive preference options.

In this chapter

The Preferences Command

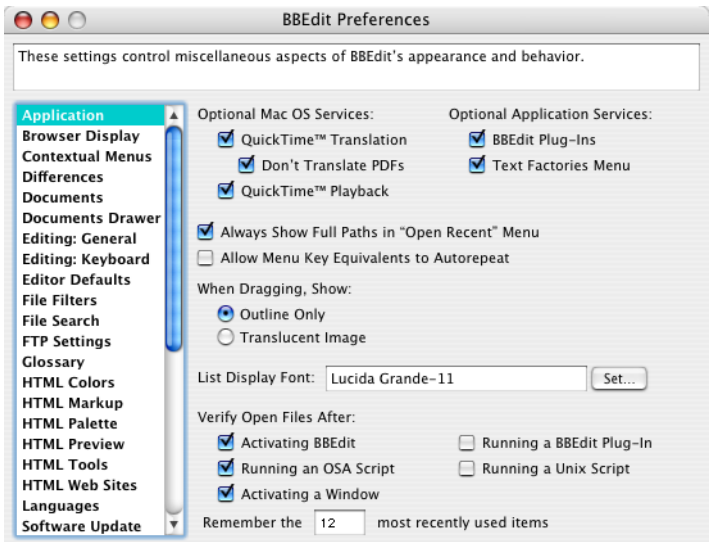
The Preferences window provides control over many aspects of BBEdit's behavior. You can decide which windows should open when you launch BBEdit, set the default display options for windows, set default options for editing behavior and searches, and so on.

BBEdit 8 employs standard OS services to store its preference settings, bringing improved durability and performance. BBEdit's use of these services also makes it possible for you to modify preference settings directly using appropriate "defaults write" commands (see page 199). However, the Preferences window remains the standard interface for making and changing preference settings.

IMPORTANT You should **not** arbitrarily remove your BBEdit preferences file, as doing so will require you to re-activate the application with your product serial number, in addition to re-selecting 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 BBEdit 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 preference panel.

The list along the left side of the window lets you select the preference panel to use.

BBEdit'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 preference panels in detail.

Application Preferences

The Application preferences control which application and system services BBEdit 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, BBEdit will ask QuickTime if it recognizes any files that BBEdit does not. If QuickTime recognizes the file as one it can display, BBEdit 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, BBEdit will not attempt to apply QuickTime translation to PDF files. This option is on by default.

QuickTime™ Playback

This checkbox controls whether BBEdit 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

This option controls whether BBEdit will display the Tools menu in its menu bar, which allows you to access BBEdit plug-ins (code modules that add functionality to BBEdit). Third-party plug-ins are kept in BBEdit's application support folders. See Chapter 2 to learn more about the organization of BBEdit's support folders, or Chapter 15 for information about plug-ins. Changes to this option will take effect the next time you launch BBEdit.

Text Factories Menu

This option controls whether BBEdit will display the Text Factories menu in its menu bar. Changes to this option will take effect the next time you launch BBEdit.

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 BBEdit'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 BBEdit, 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 BBEdit is checked, for instance, BBEdit will check the modification date of each open document every time you switch to BBEdit, alerting you if any changes have been made to the file while BBEdit was not the frontmost application. Similar functions are available to perform this check after running an AppleScript, a BBEdit 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 BBEdit's built-in browsers for search results, compile errors, disk contents, and so forth.

Results Lists

For search results browsers, you can have BBEdit 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 BBEdit 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 BBEdit 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 BBEdit 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, BBEdit 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 BBEdit rearranges the windows being compared when you click a difference item.

Differences Window Placement

The two Above/Beneath Compared Files radio buttons let you choose whether the differences window should be above or below the windows of the items 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 BBEdit should attempt to automatically resize and reposition windows during a Find Differences operation. The Hide Palettes option determines whether palettes will automatically be hidden during a Find Difference operation.

Documents Preferences

The Documents preferences control how BBEdit 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 BBEdit" 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 BBEdit warn you when you attempt to close a window with more than one document in it. This warning will not occur if:

- BBEdit is quitting
- Any of the documents in the window contain unsaved changes, since BBEdit 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 preference 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 BBEdit 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 BBEdit 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 BBEdit 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 BBEdit will not employ text smoothing.

Editing: Keyboard Preferences

The Editing: Keyboard preferences control BBEdit's response to the use of various special keys.

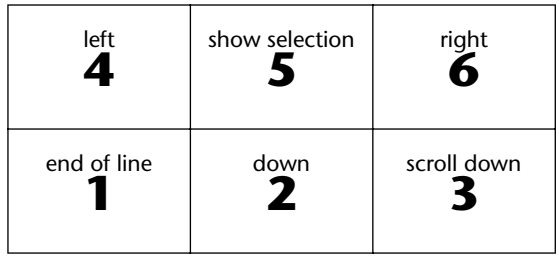
Enable Shift-Delete for Forward Delete

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 **7**	up **8**	scroll up **9**



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 BBEdit 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 BBEdit 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 sheet 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, BBEdit 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, BBEdit 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, BBEdit 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). BBEdit 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), BBEdit 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 BBEdit knows how to parse, BBEdit displays keywords and other language elements in color.

The languages that BBEdit 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 BBEdit can parse.

You can select the colors that BBEdit uses for syntax coloring in the Text Colors panel of the Preferences window.

Soft Wrap Text

When this option is selected, BBEdit 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 BBEdit 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 Filters Preferences

The File Filters preference panel lists all the file filters you have defined for multi-file searching. You can create, edit, rename, or delete filters using the buttons on the right side of this panel. For more information on creating and using file filters, see Chapter 7.

File Search Preferences

The File Search preferences control the way BBEdit 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, BBEdit looks for all the files that match the entered or selected text. Otherwise BBEdit stops looking as soon as it finds the first file that matches the selected name.

Skip (...) Folders

When this option is selected, BBEdit does not search folders whose names are enclosed in parentheses.

Search Folders

The Search Folders box displays a list of folders which BBEdit 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 BBEdit 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, BBEdit attempts to construct a file path using each of the directories shown in the Unix Search Paths list. If one resolves to a file, BBEdit 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 BBEdit 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, BBEdit 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 BBEdit. 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 BBEdit connect to this server via SFTP; if the option is unchecked, BBEdit 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.

Glossary Preferences

The Glossary preferences determine how BBEdit handles glossary entries.

Ignore Trailing CR

Click Ignore Trailing CR to have BBEdit strip off all white space from the end of a glossary entry when inserting it. This allows glossary entries to be inserted in the middle of a line, rather than as a block of text. You can use the #INLINE# cookie in a glossary entry to achieve the same effect, but only for that one entry.

Glossary Is Language Sensitive

Mark this checkbox to make BBEdit's glossary language-sensitive. If this feature is active, then each time you bring a window to the front, BBEdit automatically selects the first glossary set whose name maps to the same language as that window. For example, if the front window is an HTML document, BBEdit selects the first glossary set whose name ends in .html or .htm, or any of the other file suffixes mapped to HTML in the Languages preference panel. If there is no such entry, the active glossary set is unchanged.

HTML Colors Preferences

The HTML Colors preferences let you choose the appearance of the Web Safe Colors palette.

Color Palette Layout

You can choose from four layouts: 36 x 6 (a horizontal layout), 6 x 36 (a vertical layout), and two different VisiBone layouts, which organize the colors in a sort of circular layout that places similar colors close together. VisiBone 2 is a newer, more compact representation.

Color Swatch Size

You can choose the size of the swatches to be displayed in the palette. The default is Small.

Color Picker

The Color Picker option allows you to choose whether the color swatch buttons in various HTML dialogs use BBEdit's Web-safe pop-up menu (the factory default) or whether clicking the swatch brings up the standard system color picker dialog. Holding down the Option key on your keyboard while clicking the swatch reverses the sense of the preference.

HTML Markup Preferences

The HTML Markup preferences let you choose the format of tags generated by BBEdit's HTML tools.

HTML Tags

You can choose to generate uppercase or lowercase HTML tags, or to obtain this setting from a Dreamweaver HTML Source Profile.

You can choose a Dreamweaver HTML Source Profile to use by clicking Choose and selecting an appropriate file in the dialog, or by dragging & dropping such a file into the path box.

Quoting Tag Attributes

You may choose to always enclose tag attributes in quote marks or only to enclose attributes when the HTML standard would require it.

Quote Character

You may choose whether BBEdit should use single or double quotes to enclose tag attributes when generating new tag attributes. (BBEdit will always preserve the quote characters in use when editing any existing attributes.)

XML/HTML Markup Rules

If you are working with a document that does not contain a DOCTYPE specification or an XML declaration, you can specify whether BBEdit's HTML Tools should insert HTML or XML-style tags by selecting the appropriate radio button.

CSS Markup Formatting

The New Line Before Block Start checkbox controls how the CSS markup and formatting tools place the opening braces for block markup. If the checkbox is on, you get this style:

```
H1
{
    color: green;
}
```

If it is off, you get:

```
H1 {
    color: green;
}
```

The Put Simple Rules on One Line checkbox will format a single line of CSS like this:

```
H1 { color: green; }
```

The Allow Short Hex Color Notation checkbox controls whether hex color codes which can be expressed in a 3-digit collapsed form are inserted that way, or are inserted in the normal 6-digit format.

Close Current Tag

This option allows you to control how the Close Current Tag command inserts the appropriate closing tag.

- "Context Sensitive" places the closing tag according to various contextual clues, such as whether the opening tag is a block element or an inline element, and whether there are line breaks in a block element's contents (if the tag is a block element). This is the default option.
- "Before Insertion Point" always places the closing tag immediately before the insertion point, thus leaving the insertion point outside of the tag container.
- "After Insertion Point" always places the closing tag immediately after the insertion point. This leaves the insertion point inside the tag container.

HTML Palette Preferences

The HTML Palette preferences determine how the HTML tools floating palette is displayed.

Button Height

Choose normal or short buttons. Using short buttons allows the palette to require less space vertically.

Buttons on Main HTML Tools Palette

This list allows you to select which buttons are displayed on the palette. You can select a series of buttons by Shift-clicking their names in the display area, or a discontinuous group of buttons by Command-clicking each button name. After you make a selection, click Show or Hide to mark or unmark the selected buttons.

HTML Preview Preferences

The HTML Preview preferences tell BBEdit how you prefer to preview HTML documents.

When Previewing Files with Unsaved Changes

Choose to preview files with unsaved changes by saving the changes to a temporary file, by saving the current file just before previewing it, or by asking whether to save the file.

Note If you have set the Ask option, and choose Save when prompted, the file will be saved and then previewed; if you choose Don't Save, the preview operation will be cancelled and no changes will be made to the file.

Web Browsers Available for Preview

This list displays all the Web browsers known to BBEdit. Browsers are shown by name and version number, in the same form in which they will appear in the Preview With submenu of the Markup menu. Any browser which runs under the Classic environment will be further labeled "(Classic)."

The browser list includes each individual browser application that is available on your Mac. For example, if you have Netscape 4.7.5 and Netscape 6.2.1 on your hard disk, both applications will be listed in the preferences and available for previewing.

You can use the Add, Remove, and Change buttons respectively to add or delete a browser from the list or update an entry by choosing a different version of the application. (The Remove button does not delete a browser from your hard disk, but only removes its entry from the preview list.)

The Find All button finds all available Web browsers that BBEdit recognizes and adds them to the list if they are not already there. If using this button does not add a browser which you know is available, you can add it directly with the Add button. (Sometimes, the system may not properly advise BBEdit of every browser which is present.)

HTML Tools Preferences

The HTML Tools preferences set options for the HTML Tools.

HTML Updater

Mark the Preserve File Dates checkbox to have modification dates remain unchanged when updating HTML files using the Update Tool.

Syntax Checker Warnings

You can have the HTML Syntax Checker warn you when an HTML element is implicitly closed. Examples of these sorts of tags are <P> and <LI>. The closing tags for these elements are optional in some HTML specifications.

You can likewise have the Syntax Checker warn you about missing spaces before the closure for empty XML elements.

You can also configure the Syntax Checker to skip specifically-marked blocks which begin with the comment `<!-- #bbpragma ignore_errors="on" -->` and end with the comment `<!-- #bbpragma ignore_errors="off" -->`. Among other things, this option may be useful in preventing known warnings where you must include non-standard markup for compatibility with old browsers.

Link Checker Warnings

Choose the type of warnings to be issued by the HTML link checker.

Remote Links flags each offsite link so that it can be checked for validity manually.

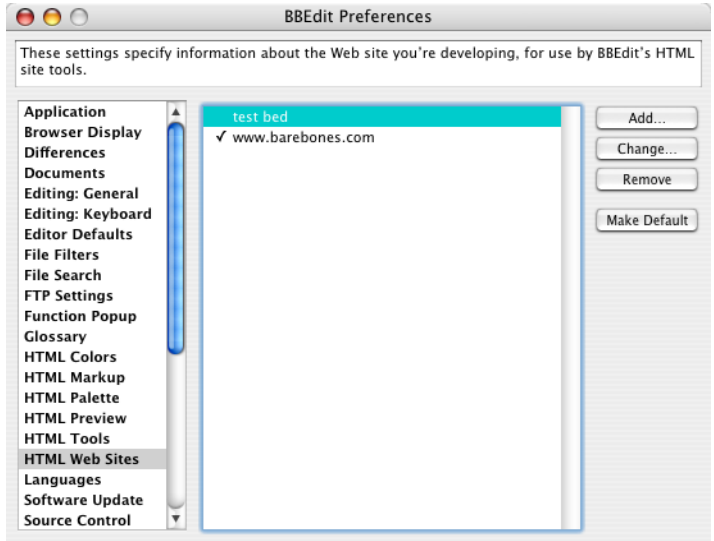
Incorrect Pathname Case flags situations where the name of a file as specified in an HTML document does not exactly match the name of the file on disk, which can cause problems when files are uploaded to servers with a case-sensitive file system.

Folder Aliases in URL Path warns you if any file paths contain aliases rather than real folders.

File Out of Server Scope controls whether the link checker will issue out of server scope warnings for links that fall outside of the server scope for the document being checked. In general you will want to leave this option on and make sure your web site preferences are properly configured. If, however, you often check transient content (such as help files or readme files) which does not merit adding a separate site configuration, you may want to turn this warning off.

HTML Web Sites Preferences

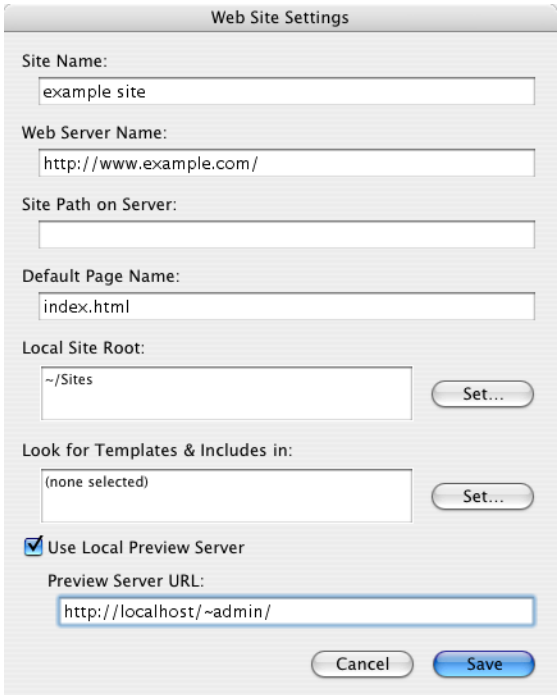
The HTML Web Sites preferences let you tell BBEdit about the Web server where your pages will be hosted.



You can define as many separate web sites as you like. The Add button brings up a Web Site Settings dialog (see below) for adding a new site to the list and specifying its properties. When a site is selected in the list, the Change button displays this same Web Site Settings dialog, allowing you to modify the existing settings for that site; the Remove button deletes the selected site from the list. The list must always include at least one site; the Remove button is disabled when only one site is listed, preventing you from deleting that last site. The check mark identifies one site in the list as the default site; you can change this setting by selecting another site and clicking the Make Default button.

All of the HTML Tools commands that generate or operate on links, such as Tag Maker and Check Links, will recognize and take account of the presence of multiple sites in the HTML Web Sites preference panel; see the descriptions of these tools in Chapter 11 for further details.

The Web Site Settings dialog displayed by the Add and Change buttons is shown below.



Site Name

Use this text field to enter the name by which the web site will be listed in the HTML Web Sites preference panel. BBEdit also uses this name in the Web Site pop-up menu of the New HTML Document dialog (see "Creating a New Document" in Chapter 11).

Web Server Name

Enter the URL of your web server here, such as "http://www.example.com/" in the figure. BBEdit uses this information to determine which links are on (local to) your server.

Site Path on Server

Enter the server path of your site's main page here. For example, if your web site is at "http://www.example.com/foo/bar/", you would enter "http://www.example.com/" for the Web Server Name (as noted above) and "foo/bar/" for the Site Path on Server.

Default Page Name

Specify the default name used by your server for the document that is sent to a web browser when a browser accesses a directory without specifying a file name. Most servers use “index.html”, “default.html”, “main.html”, or “home.html”.

Local Site Root

Use the Set button to select the folder that is the root of your web site on your local hard disk (or on a mounted server volume). You can also drag the folder’s icon onto the box to the left of the Set button.

Look for Templates and Include Files In

Use the Set button to select the folder that contains your HTML document templates and include files. You can also drag the folder’s icon onto the box to the left of the Set button.

Use Local Preview Server

If you have a web server running locally, you can preview HTML pages through it by activating this option, and entering the base URL for your preview server. (To start the built-in web server, turn on the “Personal Web Sharing” option in the Sharing panel of the System Preferences.)

Preview Server URL

When you are configuring a new web site, if the local site root folder is located within the “Sites” folder of your home folder (~/Sites/), BBEdit will create an appropriate local http URL and enter it in this field. Otherwise, you can specify the local http URL for your site root folder.

Note If your web site folders are not located within ~/Sites/, or if you want to use virtual domains, need to enable PHP, etc., you will need to modify the Apache config file. Such configurations are not directly supported through the HTML Web Site preferences.

Languages Preferences

The Languages preferences allow you to configure how BBEdit 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 BBEdit, 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.

Note The comment strings used for CSS are fixed and cannot be edited.

Suffix Mappings

By default, BBEdit 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 BBEdit 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 BBEdit.

The Check Automatically checkbox controls whether BBEdit 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 BBEdit 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 BBEdit you are using. This information is used to guide the future development of BBEdit; it is not personalized and will not be disclosed.

Source Control Preferences

The Source Control preferences panel allows you to configure settings for CVS and Perforce projects.

To add a new CVS project:

1. **Click Add CVS**

   The Configure CVS Repository dialog appears.

2. **Enter the project name, choose a root folder either via drag and drop, or by pressing the Choose button and selecting it via a standard file dialog, and set the "Use SSH" option if appropriate.**

3. **Click OK to save the project settings.**

To add a new Perforce project:

1. **Click Add P4**

   The Configure Perforce Repository dialog appears.

2. **Enter the project name, and values for each of the specified P4 environment variables as appropriate.**

3. **Click OK to save the project settings.**

To add a new Subversion project:

1. **Click Add Subversion**

   The Configure Subversion Working Copy dialog appears.

2. **Enter the project name and a username and passwor d as appropriate, and choose your working copy by clicking the folder button next to the Root display field.**

3. **Click OK to save the project settings.**

To change the configuration of an existing project:

1. **Select a project from the list.**

2. **Click Change.**

   The Configure CVS Repository, Configure Perforce Repository, or Configure Subversion Repository dialog will appear, as appropriate.

3. **Make the desired changes to the configuration settings and press "OK" to save the changes, or "Cancel" to dismiss the dialog without saving them.**

To delete an existing project:

1. **Select a project from the list.**
2. **Click Remove.**

To set the options which will be applied when performing diff operations:

1. **Click Diff Options.**
2. **Choose the desired options, and press "Save" to save them, or "Cancel" to dismiss the dialog without saving.**

You can override the client root inherited from BBEdit's environment for any P4 configuration by checking the "Override" option in the Configure Perforce Repository dialog and dragging your desired root folder into the path region below the checkbox.

Note Perforce passwords are stored in the current user's Keychain.

Spelling Preferences

The Spelling preferences control whether BBEdit uses the Mac OS X system spelling checker, or an external spelling checker.

Built-In

This option specifies that BBEdit should use the Mac OS X system spelling checker when you choose the Check Spelling command from the Text menu. You can set the highlight (underline) color that BBEdit uses to identify questioned words by clicking on the color button and choosing a color via the standard OS color pickers. Click the Reset to Factory Color button to restore the default setting. (You may find it useful to change the highlight color if you have defined a nonstandard color scheme via the Text Colors preferences.)

Word Services

When this option is selected, BBEdit 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 BBEdit does when you launch it, or when you click on BBEdit'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 BBEdit should not open any windows or perform any other actions.

New Text Document

This option specifies that BBEdit should open a new, empty text editing window.

New Disk Browser

This option specifies that BBEdit should open a disk browser starting at your home directory.

New FTP Browser

This option specifies that BBEdit should open an FTP browser.

Open

This option specifies that BBEdit should bring up the standard Open dialog, allowing you to select and open a file.

Open from FTP/SFTP Server

This option specifies that BBEdit 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 BBEdit 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 194.)

Activating Syntax Coloring

You can turn on syntax coloring globally by setting the Syntax Coloring option in the Editor Defaults preference panel. You can choose the colors which BBEdit uses for various content elements, such as keywords, string constants, etc., in the Text Colors preference panel.

Note For HTML content, you can turn on the Color Attributes Separately option to have BBEdit 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, BBEdit will color HTML attributes identically to their corresponding tags.

How to Change Colors

The color bars show the colors that BBEdit uses to display different interface and language elements. To change colors, click the color box. BBEdit 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, BBEdit 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, BBEdit 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 character set pop-up menus in various HTML tools dialogs (including the New HTML Document dialog)
- 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 BBEdit's behavior when it opens files.

Translate Line Breaks

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

If a File's Type Is Unknown

These settings tell BBEdit 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: BBEdit 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: BBEdit 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: BBEdit will inspect the file's name to see if it can figure out whether the file is text or not. BBEdit will first attempt to map the file name to the list of suffix-to-language mappings specified in the Languages preference 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 BBEdit'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, BBEdit 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, BBEdit 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 BBEdit 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 25.) 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 BBEdit 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, BBEdit 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, BBEdit honors state information that may be stored in a file. The following suboptions let you fine-tune which state information BBEdit honors.

Window Position

When this option is selected, BBEdit restores the window of the document to the same position as when the file was saved. Otherwise BBEdit opens the window in its default position.

Font Settings

When this option is selected, BBEdit restores the font information stored with a document. Otherwise it uses the default font settings.

Selection Range

When this option is selected, BBEdit 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. BBEdit now recognizes the Emacs variable block, and can optionally read and set certain variables.

If this option is selected, BBEdit 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, BBEdit restores the scroll bar position to the same position as when the file was closed. Otherwise BBEdit opens the file with the top of the file showing.

Option Settings

When selected, BBEdit 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 BBEdit's behavior when it saves files, including file backup settings.

Force New Line at End

When this option is selected, BBEdit 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 BBEdit 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, BBEdit 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 BBEdit will save a document with, if the document does not contain an encoding specification.

Make Backups Before Saving

Select this checkbox to tell BBEdit 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 BBEdit 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 BBEdit 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 BBEdit 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 BBEdit; it is no longer a property of a document's file. You can control how BBEdit 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. BBEdit now recognizes the Emacs variable block, and can optionally read and set certain variables.

If this option is selected, BBEdit 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 BBEdit prints your documents.

Default Font

To set the default printing font, click the Default Font button. BBEdit 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, BBEdit uses the document's font and tab settings when printing.

Frame Printing Area

When this option is selected, BBEdit draws a box along the edges of the printed text.

Print Page Headers

When this option is selected, BBEdit 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, BBEdit prints the full pathname of the file being printed in the header.

Print Line Numbers

When this option is selected, BBEdit prints line numbers along the left edge of the paper.

1-Inch Gutter

When this option is selected, BBEdit 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, BBEdit 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, BBEdit 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 BBEdit 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, BBEdit 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 BBEdit text windows.

All Windows:

The following options determine which controls and navigation items BBEdit should display by default in text windows and text panes.

Show Status Bar

Enable this option to have BBEdit 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 BBEdit 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 BBEdit 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 BBEdit 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 BBEdit 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/External Editors

The Coding Tools and External Editors options let you select which development environments you plan to use with BBEdit. Turning off the tools you do not plan to use will allow BBEdit to start up faster. However, in order for changes you make to any of these options to take effect, you must quit and relaunch BBEdit.

Supported environments include BBEdit's built-in HTML tools, a variety of programming packages, including Apple's Xcode, Metrowerks CodeWarrior, and Absoft Fortran, and the Dreamweaver visual HTML editor from Macromedia. BBEdit also integrates with any Unix scripting tool, such as Perl, Python, Ruby, or shells, as well as the CVS and Perforce source control systems.

In order to select any of these options, you must have the appropriate package(s) installed, e.g. the Apple Developer Tools for Xcode, or the p4 command line tool for Perforce.

IMPORTANT In order for CVS commands to be operative, you will need to first configure your system to allow repository access from the command line. The details of how to do this vary from one installation to another; consult your local CVS expert for specifics.

Default Shell

The Default Shell pop-up menu lists all Unix shells currently known and available on your system, allowing you to choose one as your preferred shell for use with shell worksheets. (Shell worksheets are discussed in detail in Chapter 14.)

Install Command Line Tools

The "Install Command Line Tools" button installs the current version of the "bbedit" and "bbdiff" tools for invoking BBEdit from the Unix command line. The first time you run BBEdit 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 preference 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 14 for further details on the "bbedit" and "bbdiff" command line tools.

Unix Scripting Preferences

The Unix Scripting preferences control how BBEdit handles files which are run as Unix filters or scripts.

Use UTF-8 for Unix Script I/O

Mark this checkbox to have BBEdit always send UTF-8 to Unix filters, and interpret Unix script output as UTF-8. This precludes the previous need for BBEdit 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 BBEdit.

Warn About Non-Unix Line Breaks Before Running

Mark this checkbox to have BBEdit post an alert when you attempt to run a filter or script which does not have Unix line breaks. Although BBEdit 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 BBEdit 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, BBEdit will pass complete environment parameters and control to Affrus, but will not attempt to retrieve output.

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 BBEdit 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, BBEdit 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 BBEdit 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 BBEdit'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 BBEdit 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

This option controls the order in which documents are listed on the Window menu, and in text window submenus. Select Name to list documents alphabetically by name, or Creation Order to list documents in the order they were opened in the current BBEdit session.

IMPORTANT This option also controls the order in which documents are displayed in the documents drawer, and on the navigation bar menu.

Window Stacking

These icons determine how BBEdit stacks windows: down and to the left, straight down, directly atop, or down and to the right. BBEdit stacks windows down and to the left by default.

Leave Room for Palettes

When this option is on, BBEdit 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, BBEdit will also leave room for any visible DragThing docks when creating or rearranging windows.

Leave Room for Finder

When these options are selected, BBEdit 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, BBEdit 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 BBEdit'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 BBEdit.

```
defaults write com.barebones.bbedit FTP:HideFTPMenuCommands -bool true
```

If you have applied this change, but want BBEdit to display the FTP/SFTP menu commands again, issue the following command in the Terminal, and then quit and relaunch BBEdit.

```
defaults write com.barebones.bbedit FTP:HideFTPMenuCommands -bool false
```

Add Temporary Files to the Open Recent Menu

Ordinarily, BBEdit 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.bbedit 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.bbedit RememberRecentTempFiles -bool false
```

CVS/Perforce/Subversion Tool Path Override

If you need to control which cvs, p4, or svn tool BBEdit should use, you can do so by issuing any or all of the following Terminal commands.

For CVS:

```
defaults write com.barebones.bbedit CVS:CVSToolPathOverride /path/to/
cvs/binary
```

For Perforce:

```
defaults write com.barebones.bbedit Perforce:PerforceToolPathOverride /
path/to/p4/binary
```

For Subversion:

```
defaults write com.barebones.bbedit
Subversion:SubversionToolPathOverride /path/to/svn/binary
```

CHAPTER

11 BBEdit HTML Tools

This chapter describes the use of BBEdit's HTML Tools, a powerful suite of utilities for creating and maintaining HTML documents and entire web sites.

In this chapter

Introduction to the HTML Tools

IMPORTANT Please be sure to read both this introduction and the next section, "Configuring the HTML Tools," before attempting to create Web pages using these tools.

Already the most powerful set of utilities ever created for the World Wide Web developer, the BBEdit HTML Tools are more powerful than ever. The HTML tools streamline the process of creating HTML documents, help you check for common usage errors, and speed up development time, without sacrificing flexibility or forcing you to work within the limits of visual editing tools.

Among the improvements made over time to the HTML Tools are that all of the tools support entry and editing of double-byte text, and they will also automatically recognize and insert tags in the correct format when used in XHTML documents.

Note Anyone preparing HTML files in UTF-8 format may want to save them without the byte-order mark, due to a bug in Microsoft Internet Explorer for the Macintosh.

The BBEdit HTML Tools and their documentation are written with the assumption that you already understand HTML. If you do not, we suggest one or more of the references listed below. None are published by or otherwise affiliated with Bare Bones Software, Inc., but other BBEdit users have found them useful for HTML usage and design issues.

Recommended Books

Learning Web Design, Jennifer Niederst. O'Reilly and Associates, 2001. ISBN: 0-59600-036-7

HTML for the World Wide Web with XHTML and CSS: Visual QuickStart Guide — 5th Edition, Elizabeth Castro. Peachpit Press, 2002. ISBN: 0-32113-007-3

Teach Yourself Web Publishing with HTML and XHTML in 21 Days — 3rd Edition, Laura Lemay, Denise Tyler, Rafe Colburn. Sams, 2001. ISBN: 0-67232-077-0

Cascading Style Sheets: The Definitive Guide, Eric A. Meyer. O'Reilly and Associates, 2000. ISBN: 1-56592-622-6

HTML & XHTML: The Definitive Guide — 4th Edition, Chuck Musciano, Bill Kennedy. O'Reilly and Associates, 2000. ISBN: 0-59600-026-X

HTML Stylesheet Sourcebook, Ian Graham. Wiley and Sons, 1997. ISBN 0-47119-664-9

Recommended Online Resources

HTML Help by The Web Design Group
`http://www.htmlhelp.com/`

Apple Internet Developer info
`http://developer.apple.com/internet/`

The Yale C/AIM WWW Style Manual
`http://info.med.yale.edu/caim/manual/contents.html`

The Bare Bones Guide to HTML by Kevin Werbach (no relation to Bare Bones Software)
`http://werbach.com/barebones/`

The W3 Consortium site
`http://www.w3.org/`

evolt.org — Browser Archive
`http://browsers.evolt.org/`

WebReference.com by Mecklermedia
`http://www.webreference.com/`

WebMonkey by Wired Digital
`http://www.webmonkey.com/`

DevEdge by Netscape
`http://developer.netscape.com/`

SGML Resources

HTML is an application of SGML and shares many concepts and characteristics with it. You do not need to know SGML to create Web pages, but you may find that studying it improves your understanding of some HTML issues.

What is SGML and why should I use it? by the SGML Project
`http://www.oasis-open.org/cover/exetwhat.html`

SGML on the Web by Softquad Inc.
`http://www.ncsa.uiuc.edu/SDG/Software/Mosaic/WebSGML.html`

What You Need

Before you start, make sure you have the following available:

- A modern Web browser for previewing your pages. We suggest having at least one version each of Microsoft Internet Explorer, Netscape Navigator or Communicator, Mozilla, and OmniWeb available, plus Safari (always available as a standard OS-provided application). These are the most widely used browsers but they often do not display pages the same way.
- A general-purpose FTP client such as Fetch, Interarchy, or Transmit. While BBEdit does have built-in FTP tools, such dedicated applications are, naturally, more powerful, and of course also allow you to upload things other than text files. You will find them useful in creating and managing your web site.
- Access to a Web server, either your own or someone else's, where you will publish your pages on the Web. (Your Internet provider is a good place to start and can help you find the answers to questions about obtaining your own domain name, setting up your own dedicated server, and so on.)

You will also want to be familiar with BBEdit's basic capabilities. The other chapters in this manual will help you learn more about editing and searching text using BBEdit.

Configuring the HTML Tools

Before using the HTML Tools, you should prepare your source documents and your copy of BBEdit. Although you can change your settings at any time, we suggest visiting the Preferences window before starting.

First, create a folder to contain working (draft) copies of your HTML documents. Put the HTML documents you are working on in this folder (either copying them from your web site or moving them from elsewhere on your hard drive); then upload them to your Web server when you are satisfied with them.

Next, launch BBEdit and open the Preferences window (by choosing Preferences from the BBEdit menu). For now we will only set the preferences for working with HTML. Refer to the following sections of this manual for further details:

- "HTML Colors Preferences" on page 177 to choose a layout for the Web Safe Colors palette
- "HTML Markup Preferences" on page 177 to choose options for formatting HTML tags and attributes

- "HTML Palette Preferences" on page 178 to select the appearance of the HTML Tools palette
- "HTML Tools Preferences" on page 179 to tweak the function of the HTML Tools
- "HTML Web Sites Preferences" on page 181 to tell BBEdit where your web site is (locally and on your server)
- "Languages Preferences" on page 183 to check that there are suitable file suffix mappings for all the types of documents you will be working with

Note In order to apply syntax coloring to stand-alone PHP files, you must use the Languages preference panel to set the suffix mapping for such files to be "PHP". If the mapping is set to "HTML", "XML", or "JSP", the PHP content must be enclosed by an appropriate tag (<SCRIPT language="php">...</SCRIPT> or <?php...?>) in order to have BBEdit apply syntax coloring.

Using the HTML Tools

There are two ways to use BBEdit's HTML Tools: via the HTML Tools floating palette (right) and through the Markup pull-down menu. These two methods are functionally equivalent in most respects.

Most beginning users find it easiest to use the HTML Tools through the palette. There are three basic types of buttons on the HTML Tools palette:

- Those you simply click to perform an action or bring up a settings dialog before performing an action—for example, New Document or Preview in BBEdit
- Those that provide pop-up menus containing related options—for example, Heading, Frames, and Utilities
- Those that contain both a clickable button and a pop-up list—for example, List, Table, and Preview

Buttons containing a "grip-strip" (a double vertical bar on the left side) may be used via drag and drop. For example, the Image button may be dragged into a document window to display the Image tool dialog. Subsequently, the specified <IMG> tag is placed at the point where the button was dropped.

Tip You can Command-click a pop-up menu to bypass the menu display; the tool will perform the last-used action again.

The second means of using the HTML Tools is from BBEdit's Markup menu. This allows you to make your own choice between the drag and drop convenience of palettes, and the less screen-intensive menus; either way, you will still be able to access all of the HTML Tools' capabilities.

Every HTML Tools function is available through an item on the Markup menu or one of its submenus. Key equivalents (if assigned) are displayed next to the menu item. (You can change or assign keyboard shortcuts to menu commands by choosing Set Menu Keys from the BBEdit menu.)

Many tool dialogs offer keyboard shortcuts for activating buttons. Hold the Command key down when a dialog is open to see these shortcuts.

Most of the HTML Tools commands apply to the document which you currently have in the foreground—either at the current insertion point, or on the current range, as appropriate. Some of the utility functions, however, can operate on many documents. The Tool Descriptions section provides more details on how each function works.

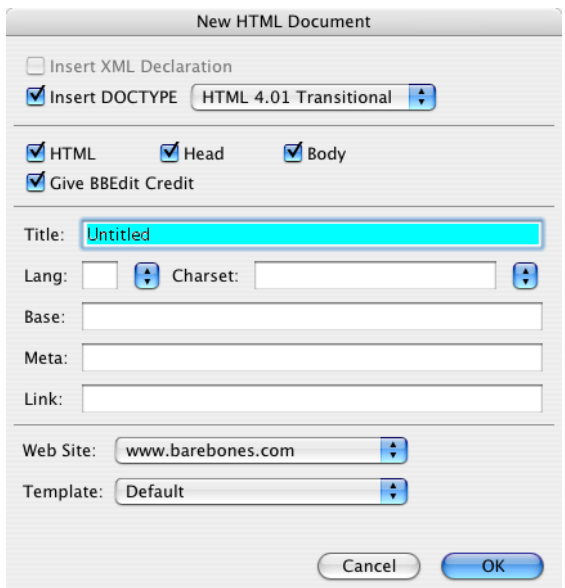
Creating a New Document

You can create an HTML document simply by taking any text file and adding HTML markup to it, but there's a better way. BBEdit includes a New Document command to create the basic skeleton of an HTML document for you.

To create a new HTML document, you can do either of the following:

- Chose New from the File menu and then choose HTML Document from the New submenu.
- Click the New Document button in the HTML Tools palette.

In either case, the following dialog appears:



Note If the frontmost document in BBEdit is editable, you can deselect the Create New Document option in the dialog. This will insert the document template around the current text contents of the window.

In many cases, you can simply specify a title for the document and click OK, ignoring the other options. However, we suggest that you fill out this dialog as completely as possible. The function of each field is described below.

Insert XML Declaration

Choose this checkbox to have BBEdit insert an XML declaration. If the DOCTYPE selected in the pop-up menu below is not an XML-based type (that is, is not an XHTML version), this checkbox will be disabled.

Note Although an XML declaration is required for XML documents whose character set is not UTF-8 or ISO 8859-1, some currently shipping Web browsers (including Internet Explorer for the Macintosh) cannot handle documents with XML declarations.

Insert DOCTYPE

Choose the type of this HTML document from the pop-up menu to have BBEdit insert an SGML prolog containing the desired document type. This information is largely ignored by browsers; however, HTML syntax checkers (such as the one built into BBEdit) use it to determine which constructs are legal according to the HTML standard you select. Available DOCTYPEs include:

- HTML 3.2
- HTML 4.01 (Transitional, Frameset, and Strict versions)
- XHTML 1.0 (Transitional, Frameset, and Strict versions)
- XHTML 1.1
- Compact HTML (CHTML) 2.0

Note Although HTML 4.0 has been superseded by HTML 4.01 and most users will want to use the latter, BBEdit's syntax checker continues to accept HTML 4.0 as well, for historical reasons.

HTML, Head, and Body

Every HTML document should contain <HTML>, <HEAD>, and <BODY> tags. Mark these checkboxes to have BBEdit insert these tags automatically.

Give BBEdit Credit

This option generates a <META NAME="generator" CONTENT="BBEdit 8"> tag in the document, indicating that you used BBEdit to create it.

Title

Enter the HTML title for the document (which can be different from the file name) here. This text will appear in the title bar of a browser's window when this document is opened.

Lang

This option indicates the language this document is written in. This information can be used by search engines and translation software to help Web users find pages in their own language.

Charset

This option indicates the character set used by the document. If you do not specify a character set, the character set chosen in the user's browser will be used.

Note You can choose which character sets appear in this pop-up menu by using the Text Encodings panel of the Preferences window.

Base

Enter the URL for this document's BASE tag. The BASE tag indicates the actual location of the document on a server, and all relative URLs specified in the document will be resolved by the browser relative to this location. No BASE tag is created if you leave this field blank.

Meta

Enter the META tag to be included at the top of the document here, if any. (META tags can be used for "client-pull" techniques, for indicating search keywords, and for a wide variety of other purposes.)

Link

If you want to use a LINK tag to specify a relationship between this document and other documents, an email address, style sheet, or other information about the document, enter the desired information in this field.

Note If you use a template to create the HTML document, the template must include the #META# and #LINK# placeholders to indicate the location at which this information should be inserted into the generated document.

Web Site

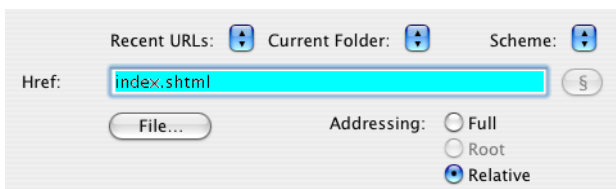
This pop-up menu displays the web sites defined in your HTML Web Sites preference panel, allowing you to switch between them for the purpose of choosing a template to base the new document on. The initial selection on opening the dialog is the default site designated in the HTML Web Sites preferences.

Template

This pop-up menu displays the templates specified in the "Templates & Includes" folder associated with the selected web site in the HTML Web Sites preferences. (The names of template files must end with ".tmpl" to be displayed in this menu.) Selecting a template other than Default will use the specified template to create a new document, potentially ignoring some or all of the settings specified in this dialog.

File Addressing

Many of the HTML tools require you to specify the pathname or URL of a file or folder, such as to identify a base address, style sheet, or hypertext link. Typically, tools will provide a File button you can click to locate the desired file via an Open dialog, together with a text field for displaying the resulting URL.



The URL can be expressed in any of three ways, selected by radio buttons below the text field:

- *Full addressing* specifies the complete URL, including the scheme ("http:"), the server's domain name, and the complete directory path leading to the file within that server.

- *Root addressing* specifies just the file's location within its host server.
- *Relative addressing* specifies the file's location relative to that of the HTML document referring to it.

For example, if the web site resides on a server named "www.example.com" in directory "foo/bar," and you are creating a document in that directory named "index.html" with a link to file "target.html" in subdirectory "flapdoodle", the full address would be

```
http://www.example.com/foo/bar/flapdoodle/target.html
```

the root address would be

```
/foo/bar/flapdoodle/target.html
```

and the relative address would be

```
flapdoodle/target.html
```

If you have designated a folder on your local disk as the site root for your web site in the HTML Web Sites preference panel, the HTML tools will recognize files within that folder and substitute the server domain and site path in its place. For instance, if your local site root for the web site above is the folder "HD/HTML Documents/Main Site", the tools will automatically convert the file path

```
HD/HTML Documents/Main Site/flapdoodle/target.html
```

to

```
http://www.example.com/foo/bar/flapdoodle/target.html
```

If you have defined multiple web sites in the HTML Web Sites preference panel, the HTML tools will recognize and convert file paths lying in any of their site root folders. This will work even when the source and target files are on different sites; the tools will convert the target file's pathname to a full URL within the target web site. When the source and target files are both on the same site, all three address styles (full, root, and relative) are available; when the files are on different sites, root addressing is disabled (its radio button is dimmed). For files residing in local site root folders on the same disk, relative addressing from one file to the other is always available.

Checking Syntax

You can use BBEdit's Check Syntax command (see page 232) to validate your HTML documents to the specification defined in their <!DOCTYPE> SGML prolog. BBEdit will apply HTML 4.0.1 Transitional rules when checking any document that does not contain a <!DOCTYPE> specification,

Note that an HTML document can display the way you expect it to in a browser and still contain invalid HTML. Browsers are designed to be lenient in the markup they accept, so you can get away with a certain amount of "sloppy" markup. However, producing well-formed (syntactically correct) HTML documents is the best way to assure that your document will display in some reasonable fashion in a wide variety of Web browsers, even those you have not tested the page in.

Syntax Checking Partial Documents

BBEdit can check the syntax of either complete or partial HTML documents. You may find the ability to check partial documents useful if you are preparing template sections for inclusion into other documents, whether this is done locally or via a server-side mechanism.

In order to check the syntax of a partial HTML document, the document must contain a balanced portion of the content tree. For example, you can check a partial document which contains a set of paragraphs or a table; you cannot check a partial document which contains an unclosed <BODY> or <DIV>.

Additionally, for partial documents which do not contain a DOCTYPE, you can specify one by means of a "#bbpragma doctype=" comment, which specifies what the root or parent element of the partial page's content is. For example, if your partial document consists of <BODY> content:

```
<!-- #bbpragma doctype="-//W3C//DTD HTML 4.01 Transitional//EN"
root_element="body" -->
```

In this comment, you must use either the public identifier text for the DOCTYPE you wish to check against (see above) or the "display name" for the document type used in the New HTML Document dialog.

Ignoring Sections of Documents

You can mark sections of HTML documents to have BBEdit ignore these sections when performing a syntax check. This can be useful for purposes such as checking documents which must from necessity incorporate non-standard markup to support old browsers, or which contain customized server constructs. To mark a section, enclose it with "#bbpragma ignore_errors=" directives, as follows:

```
<!-- #bbpragma ignore_errors="on" -->

ignored markup

<!-- #bbpragma ignore_errors="off" -->
```

Note that when you check a document containing ignored sections, BBEdit's syntax parser still runs through the markup contained in these sections; it simply does not report errors encountered there. You should thus be mindful of the following conditions:

- The presence of fragmentary tags or similarly malformed content in an ignored section can cause a syntax check to fail.
- An error may still be generated for an unclosed element which resides within an ignored section, if its lack of closure results in an error cascade which continues beyond that section.
- If you terminate a document inside of an ignored section, an error will be generated.

Previewing Pages

IMPORTANT BBEdit's Preview commands allow you to view your pages in one or more web browsers. You can display an automatically-updated page preview directly within BBEdit by choosing the Preview in BBEdit command, use the Preview in <Default Browser> command to use the current default browser, or choose a specific browser from the Preview With submenu. You can also preview the page in all running browsers or in a text-only format. (Any browser which runs in the Classic environment will be labeled "(Classic)".)

In addition to static previews, BBEdit also supports live local previewing of web pages through the standard web server built into Mac OS X. This capability enables you to easily preview pages which are built using server-side technologies, for example, DHTML or PHP.

To enable live previewing for a defined web site project, you must turn on the Use Local Preview Server option and enter an appropriate Site Path on Server in the HTML Web Sites preferences panel (see page 183). You must also turn on the "Personal Web Sharing" option in the Sharing panel of the System Preferences to start the built-in web server.

Once you have done so, whenever you preview a file from that web site project using any of the Preview In commands, BBEdit will have the selected browser load that file's corresponding page through the built-in web server.

Printing Previewed Pages

When a BBEdit preview window is frontmost, you can use the Page Setup and Print commands to print a copy of the displayed page.

Note Due to limitations of the Safari rendering engine which BBEdit employs, the format of the printed output may not exactly match the screen rendition.

Previewing from a File Vault-protected Home Folder

If your home folder is protected by FileVault, you will have to change the permissions on your home folder and your Sites folder (~/Sites/), before you can preview documents via "Personal Web Sharing". Your home folder should have permissions of at least (drwx--x--x), and your Sites directory should have permissions of at least (drwxr-xr-x).

To set these permissions using the Terminal application, you can issue the following commands:

```
~ username$ cd ~

~ username$ cd ..

~ username$ chmod 711 username

~ username$ cd ~

~ username$ chmod 755 Sites/
```

where the command text is shown in bold, and **username** is the short form of your user account name. You can also use any file utility which is capable of setting folder permissions, such as Super Get Info, to make these changes.

HTML Tool Descriptions

This section describes all of the HTML Tools as they appear in the hierarchical Markup menu. For a description of the tools as they appear on the palette, see the HTML Tool Palette Index, which appears after the tool descriptions.

Note Tools that create tags insert the tag at the insertion point unless otherwise specified. Such tools also place an end tag automatically where appropriate.

You should already be familiar with HTML before using BBEdit's HTML tools. The dialogs BBEdit displays will help you associate correct attributes with each tag, and provide shortcuts to help you enter information, but they do not (and cannot) tell what the final results of your markup will be. There is no substitute for knowing HTML.

Tag Maker

Tag Maker is BBEdit's context-sensitive HTML authoring tool. When you choose it from the menu or the palette, or invoke it by pressing Command-M, it provides a list of the tags that are valid in the current HTML context of the insertion point and then lets you insert and configure one. For example, if the insertion point is positioned inside the document's header section (delineated by the <HEAD> and </HEAD> tags), Tag Maker displays the following list of tags:

Choosing, for example, BASE inserts a <BASE> tag at the insertion point. Choosing TITLE inserts a <TITLE></TITLE> pair, with the insertion point positioned between the two tags for easy entry of a title.

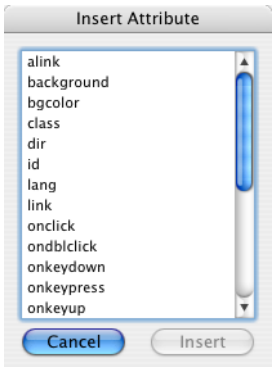
If you hold down the Option key while clicking the Insert button, Tag Maker automatically displays a follow-up dialog for editing the attributes of the new tag, if the tag takes attributes and BBEdit has a tag editing dialog for it.

When there is only one tag permitted in the current context, Tag Maker does not display the Insert Tag dialog, but simply inserts the appropriate tag at the insertion point. (Choose Undo from the Edit menu if this tag is not what you expected.) If there are no valid tags in the current context, Tag Maker sounds the system alert beep.

Tag Maker also works with CSS. With the insertion point in a CSS selector or declaration, Tag Maker will display a list of valid CSS 2.1 properties. Option-click while clicking the Insert button and Tag Maker will open a follow-up dialog for editing the property (for classes that BBEdit offers editing dialogs). Invoking Tag Maker when the insertion point is not within a CSS selector will create a new skeleton rule set.

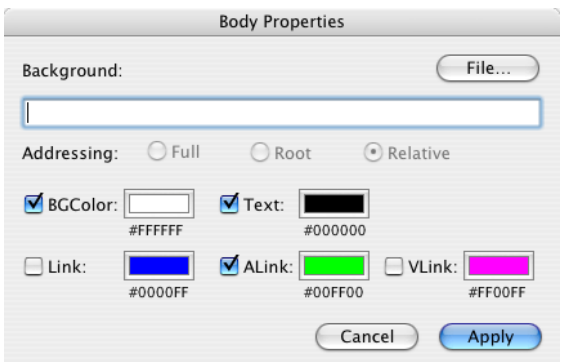
Tag Maker is also available through contextual menus. Simply Control-click at any point in your markup, and BBEdit will open a contextual menu that includes all valid attributes or properties for the context of the insertion point.

Once you have used Tag Maker to enter a tag, you can place the insertion point inside a tag and invoke Tag Maker again to show a list of available attributes for the tag. (Attributes that are already present in the tag are not shown in the list.) The list below, for instance, shows attributes of the <BODY> tag (under HTML 4.0 Transitional). Selecting an attribute from the list adds it to the tag and places the insertion point at the proper location for typing a value.



Edit Tag

Another way to add or edit the attributes of a tag is to place the cursor inside the tag and choose Edit Tag from the Markup menu or the HTML Tools palette. Choosing Edit Tag displays a dialog appropriate to editing the most common attributes used with the current tag. (The dialog shown is for a BODY tag; dialogs for other tags will look different.)



Edit Tag also works with CSS. Choose Edit Tag while the insertion point is within a selector's property or value, and BBEdit will display an appropriate dialog for editing many common properties.

Close Current Tag

The Close Current Tag command inserts a closing tag to match the nearest opening tag preceding it. If the closing tag is placed on a new line, it will use the same indent level as the opening tag. For instance, if the insertion point is preceded by a <P> (Paragraph) tag plus some text content, Close Current Tag will insert a matching </P> tag to close the paragraph.

Note If you frequently work with HTML documents, you may want to assign a key equivalent to this command using Set Menu Keys.

Balance Tags

When Balance Tags is chosen, BBEdit expands the selection to encompass the content of next outermost set of enclosing tags. The easiest way to understand how this works is to see it in action. Place the insertion point in an HTML document's <TITLE> element and choose Balance Tags. The title will be selected, since it lies between the tags <TITLE> and </TITLE>. Choose Balance Tags again, and BBEdit selects everything between <HEAD> and </HEAD>, the next set of enclosing tags outside the <TITLE> element. Choosing the command once more will select everything between <HTML> and </HTML>.

Use this command to quickly select an element for editing or just to check to see whether all your nested elements are formed correctly. If BBEdit sounds the system alert beep when you expect it to select text, it cannot find a matching set of tags around the selected text.

Document Type

Choosing Document Type brings up a dialog that allows you to select the desired document type (DTD) for the current document. If the document already contains an SGML <!DOCTYPE> declaration, this will be used as the default. You may choose any of BBEdit's supported document types or enter one of your own.

Character Set

Choosing Character Set brings up a dialog that allows you to enter a character set specification for the current document. To insert this specification, check the Meta Tag box, choose a character set from the pop-up menu of supplied standard types or type in a character set name, and click Apply. To remove any existing character set specification, deselect the Meta Tag box and click Apply.

Note You can choose which character sets appear in the pop-up menu by using the Text Encodings panel of the Preferences window.

Optionally, if the current document is an XML document, you can insert an XML Declaration (either alone or in addition to the Meta Tag) by checking the XML Declaration box and clicking Apply.

CSS Submenu

This submenu allows you to create, edit, and format Cascading Style Sheet markup. BBEdit has built-in support for CSS. When you are editing stand-alone CSS files or HTML files with embedded CSS, syntax coloring is available, and the Function pop-up menu in the status bar lists CSS selectors, as well as CSS files referenced by @import directives and <link> tags. Choose an external stylesheet from the Function pop-up and BBEdit will open that stylesheet for editing.

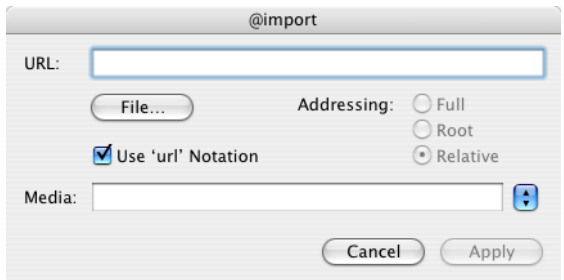
IMPORTANT BBEdit 8 supports all CSS 2.1 properties. Although there are no explicit dialog editors for many such properties, you can create or modify them using the Tag Maker command, or by using Insert Property from the contextual menu.

The CSS function parser supports the following syntax for laying a mark in the function menu:

```
/* bbmark string to appear in the menu */
```

@import

The @import directive instructs a Web browser to load an external style sheet. This dialog box allows you to select a file (or drag and drop one from the Finder) and choose whether to use the optional "url" notation for specifying the location of the style sheet. (Remember that @import must come before other CSS rules inside a <style> tag or in a stand-alone CSS document; otherwise it will not work.)



@media

The @media directive allows you to control which stylesheet should be used for different output media, e.g. screen versus printer. You can use this dialog to add media rules, or given an existing media rule, you can use the CSS editing dialogs, Edit Tag, or Tag Maker to edit the rulesets within.

For example:

```
@media print {
   body { font-size: 10pt; }
}

@media screen {
   body { font-size: 13px; }
}

@media screen, print {
   body { line-height: 1.2; }
}
```

Format

The CSS Format command will reformat your CSS markup for easier reading.

In stand-alone CSS files, if there is a selection range, only the selected text is formatted. If there is no selection range, the whole file will be formatted.

In HTML files with embedded CSS, if there is a selection range, only the selected text will be formatted. If there is no selection range, BBEdit will format all CSS in the <style></style> tag pair that encloses the insertion point. If the insertion point is outside a <style></style> tag pair, or if the selection range spans a <style></style> tag pair, the formatter will simply beep.

When formatting CSS embedded into HTML, BBEdit will indent the CSS based on the indent level of the opening <style> tag, plus one additional tab stop for better readability.

There are two preference settings in the HTML Markup preference panel that control how the CSS formatter and markup tools place the braces for block markup:

- The New Line Before Block Start checkbox controls the placement of opening braces. If the checkbox is on, you get this style:

  ```
  H1
  {
      color: green;
  }
  ```

  If it's off, you get:

  ```
  H1 {
      color: green;
  }
  ```

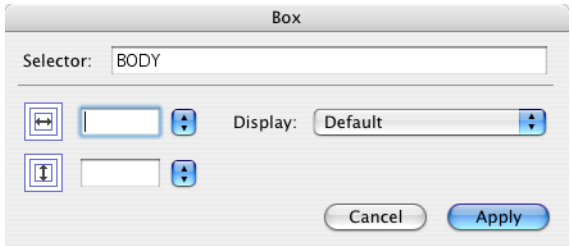
- The Put Simple Rules on One Line checkbox will format a single line of CSS like this:

  ```
  H1 { color: green; }
  ```

BBEdit's CSS markup tools (listed below) use the same rules for formatting as does the Format command.

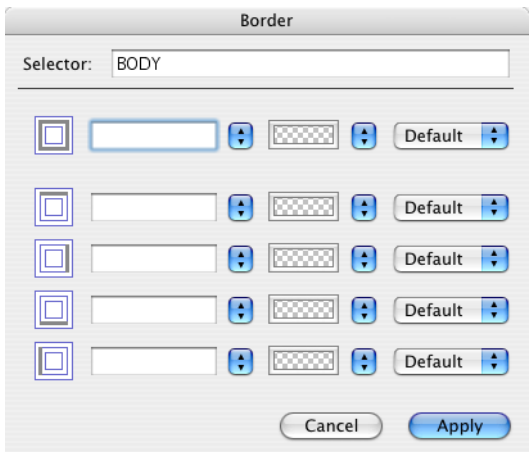
Box

The Box dialog allows you to specify a selector's width, height, and display properties.



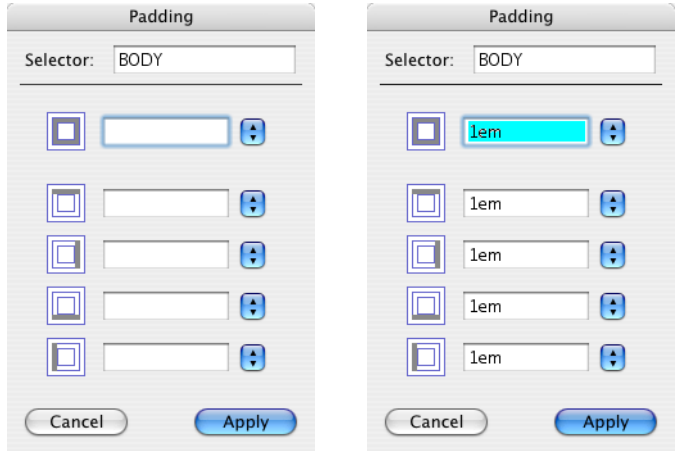
Border

The Border dialog allows you to edit the border-width, border-color, and border-style properties for a selector. The first row lets you specify values that apply to all four sides. The color buttons let you select colors from your preferred Web color palette (as specified in the HTML Colors preference panel); the pop-up menus next to them let you select colors by name. The icons on the left side of the dialog represent (from top to bottom), the entire border, top, right, bottom, and left.



Padding/Margins

These similar dialogs allow you to edit the padding and margin properties. In both cases, the icons on the left in the dialogs represent the entire box, top, right, bottom, and left, respectively.



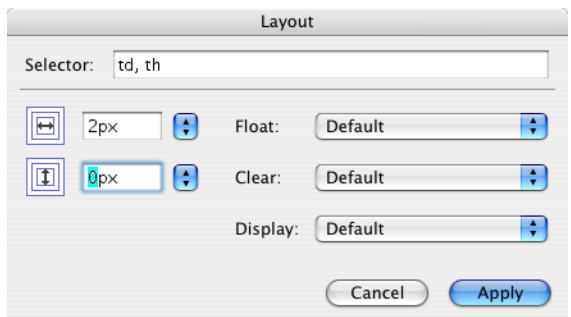
When you are working with the individual margin fields in these dialogs as opposed to the overall value, they behave the same way CSS value replication does:

- If right is missing, it takes on the value of top
- If bottom is missing, it takes on the value of top
- If left is missing, it takes on the value of right

so an empty field has special meaning - it means "replicate the related value". If you want to specify a value for any given side, you must enter it explicitly.

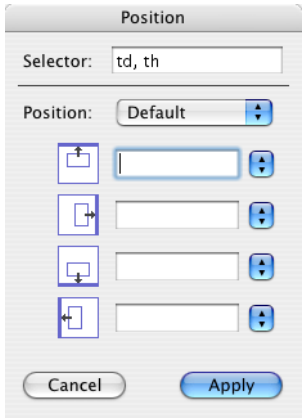
Layout

The Layout dialog allows you to edit the page layout properties of a selector.



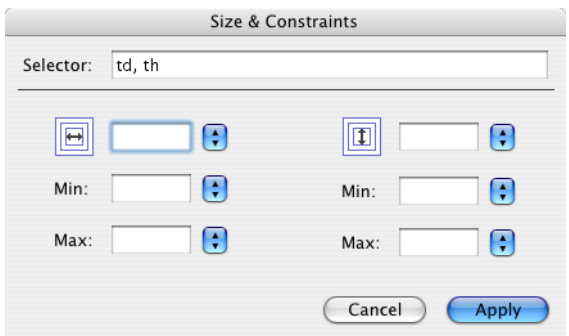
Position

The Position dialog allows you to edit the positioning properties of a selector.



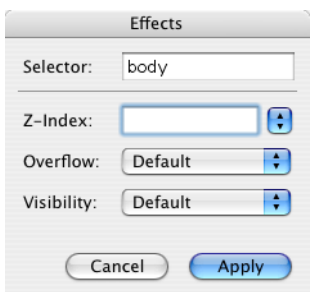
Size & Constraints

The Size & Constraints dialog allows you to edit the size and constraint properties of a selector.



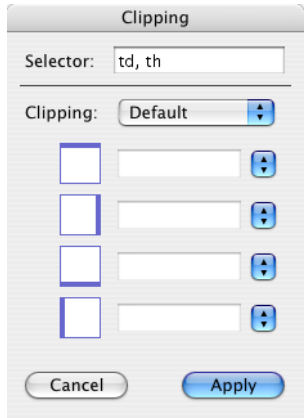
Effects

The Effects dialog allows you to edit the z-index, overflow, and visibility properties of a selector.



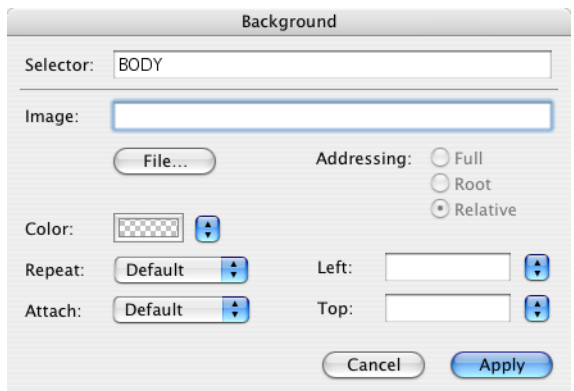
Clipping

The Clipping dialog allows you to edit the clipping properties of a selector.



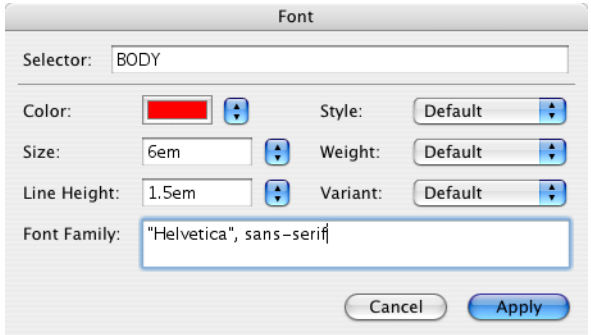
Background

The Background dialog allows you to edit background-image, background-color, background-repeat, background-position, and background-attachment properties. The Image field allows you to select an image file by clicking the File button, or by using drag and drop from the Finder.



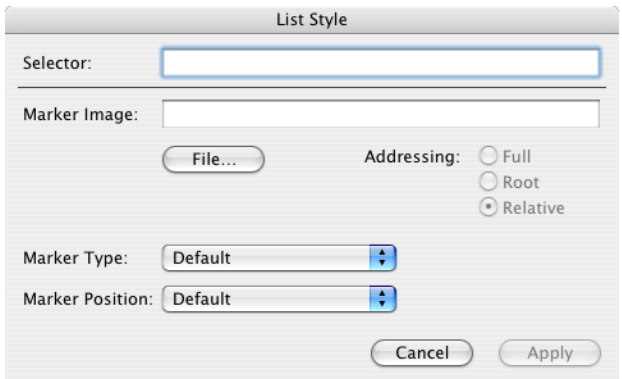
Font

The Font dialog allows you to edit values for the following font properties: color, font-size, line-height, font-family, font-style, font-weight, and font-variant. Note that BBEdit will parse the "font:" shortcut property, but never generates it; instead, BBEdit generates exploded values for font-style, font-variant, font-family, and font-weight.



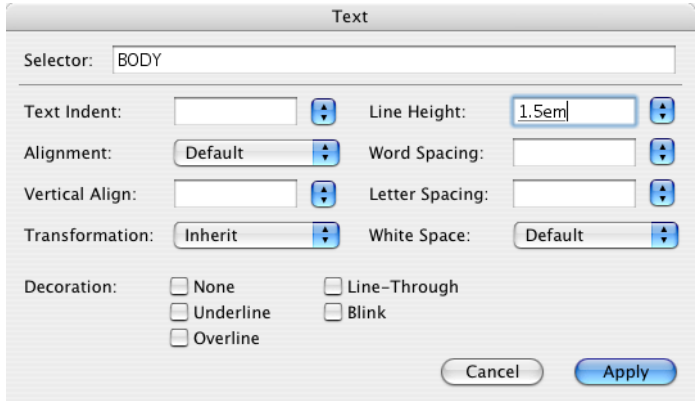
List

The List dialog allows you to edit the following list properties: marker image and addressing format, marker type, and marker position.



Text

The Text dialog allows you to edit the following text properties: text-decoration, text-indent, text-align, vertical-align, line-height, text-transform, word-spacing, letter-spacing, and white-space.



Body Properties

This submenu allows you to edit attributes of the BODY tag in the current document, including the background image, background color, text color, link color, active link color, and visited link color. You can specify the background image by typing or by clicking the File button. Link, background, and text colors are chosen from the 216-color "Web-safe palette" that appears when you click one of the color swatches. (Option-clicking a color swatch will bring up the system color picker, so you can select colors that are not in the Web-safe palette.)

Head Elements

This submenu allows you to add tags that are valid in the document's head. The tag is inserted at the current insertion point, even if the insertion point is not a contextually valid location for the tag.

Base

The BASE tag determines the default location of documents referenced in the current document. You will be prompted to enter the Href (the document or folder that all relative links in the document should be considered relative to). You can enter the Href by typing it into the supplied field or by clicking the Folder button to select a folder on your hard disk. A set of radio buttons offer the choice of full, root, or relative addressing. The Target field determines the frame in which hyperlinked documents will appear by default. Both fields are optional.

Link

The LINK tag tells the browser about a document related to the current document. The most common use for this tag is to point to an externally defined CSS stylesheet document. You can choose to indicate whether this link defines a REL (forward relation) or REV (reverse relation), the type of the relation (for example, a stylesheet), the media, the URL of the referenced file (including full, root, or relative addressing), and optional Type, Hreflang, Target, and Charset attributes.

Meta

META tags are used to define browser-specific or optional information that is not a part of the HTML specification. You can choose to create either a NAME or an HTTP-EQUIV variation of the META tag. The latter is frequently used for "client-pull" applications in browsers that support it, but more generically it makes most browsers behave as if the specified line was received as part of the HTTP protocol header. After choosing the type of META tag and the desired value of the NAME or HTTP-EQUIV field, enter a value for the tag's CONTENT attribute and optionally its SCHEME attribute. (You can choose from some common values for the NAME field by using the pop-up menu to the right of that field. If the HTTP-EQUIV value is "content-type", you can also choose from among some common values for the Content field by using the pop-up menu to its right.)

Script

This command begins a section of client-side script code (by default, JavaScript, although some browsers support other scripting languages). You can choose to execute a script contained in an external file by entering a URL in the Source field (click File to choose the file using an Open dialog). You can also enter values for the TYPE of script, the script LANGUAGE it is written in, and the character set or CHARSET of the script. Mark the DEFER checkbox to add a DEFER attribute.

Note You can choose which character sets appear in the Charset pop-up menu by using the Text Encodings panel of the Preferences window.

Noscript

This command begins a section of HTML to be displayed only if the Web browser does not support client-side scripting; often used to provide alternate content following a <SCRIPT> block. A matching </NOSCRIPT> tag is also inserted.

Style

This command begins a stylesheet declaration. You will be prompted to enter the type and title of the stylesheet, and optionally select the Media type it is to be used with.

Block Elements

This submenu lets you add HTML elements that behave as paragraphs or other types of text blocks. Since many block elements can be containers for other elements, most have an explicit or implicit ending tag (for example, </P> to close a paragraph) as well as an opening tag (for example, <P>). If text is selected when one of these commands is chosen, these opening and ending tags are placed before and after the selection.

Paragraph

This command begins a new paragraph element. You can specify alignment, ID, a class (for obtaining formatting cues from a stylesheet), and inline CSS style information. If a selection exists, each paragraph in the selection has <P> and </P> tags inserted around it. You can also control whether the opening and closing tags should be inserted on the same line as the content, or on separate lines, by setting the Place Tags on Separate Lines option appropriately.

Div
This command begins a new division. A division is a generic block of text containing one or more paragraphs (or other block elements) that all have some type of structural attribute in common. Use <DIV> when no predefined block type is appropriate. You can specify alignment, ID, a class (for obtaining formatting cues from a stylesheet), and inline CSS style information.

Horizontal Rule
This command inserts a <HR> tag. You can specify the alignment, the thickness (size) of the rule, its width, and whether it contains a three-dimensional "shade."

Heading
This command inserts a heading of any level and allows you to specify the alignment of the heading.

H1 through H6
These commands insert a heading of the specified level. (The alignment attribute used will match the last one chosen when using the Heading tool.)

Address
This command inserts an <ADDRESS> block. The formatting of this element is browser-dependent but it is usually used to indicate that a block of text is a postal address.

Blockquote
This command inserts a block quote—that is, several lines of text that have been quoted from another document. (Most browsers display this as indentation, leading many authors to use this tag to indent a section of text, although stylesheets are a more correct and flexible way to accomplish this.) You may optionally indicate the document being quoted, if it is available on the Web, using the Cite field.

Center
This command inserts a block centering tag. This tag, while permitted in current HTML specifications, is deprecated since it includes no information about the content being centered. It is generally considered better form to use stylesheets or P or DIV tag ALIGN attributes instead.

Deleted Text
This command inserts a block formatted to indicate that the enclosed text has been deleted (usually with a horizontal line through it—that is, "struck out"). You may optionally specify a citation (indicating a reference to another file) and a date and time.

Inserted Text
This command inserts a block formatted to indicate that the enclosed text has been inserted (usually by underlining the text). You may optionally specify a citation (indicating a reference to another file) and a date and time.

Noscript
This command begins a section of HTML to be displayed only if the Web browser does not support client-side scripting; it is often used to provide alternate content following a <SCRIPT> block. A matching </NOSCRIPT> tag is also inserted.

Preformatted

This command begins a section to be reproduced with line breaks as specified in the HTML document. (Normally, browsers convert carriage returns to white space for display, breaking lines only at <P> or
 tags.) Most browsers use a monospaced font for this type of block.

Lists

These commands add numbered or bulleted lists to your HTML documents. If text is selected, the selected text is converted to a list, with each line (terminated by a carriage return) becoming a list item.

List

This command inserts a list. You can choose the type of list (unordered, ordered, definition, menu, or directory) and also the type of marker for an unordered ("bulleted") list. You can also suggest a compact display format for the list. When converting existing text to a list, you can choose to ignore blank lines in the text being converted, to mark up only list items (and not insert the list header), and whether to indent the list items. When converting text to a definition list, DT (term) and DD (definition) tags are applied to alternating lines in the selection.

Unordered/Ordered/Definition/Menu/Directory

These commands convert the selected text to the indicated type of list, or insert a new list (as with the List command) using the options set in the last List dialog displayed.

List Items

This command converts selected text to list items (one line becomes one item), or inserts an <LI> tag if no text is selected.

Tables

The commands on this submenu all have to do with building HTML tables. HTML tables are frequently used for layout purposes as well as for the display of tabular data, although strictly speaking their use for layout should be avoided as much as possible. BBEdit fully supports HTML 4 tables, which are considerably more flexible than the HTML 3.2 tables many users may be familiar with. If some tags or attributes in this section are unfamiliar to you, we strongly encourage you to study the HTML 4 standard before attempting to use them.

Table

This command inserts <TABLE> and </TABLE> tags around the selected text. You can specify border, width, spacing, padding, frame, ruling, alignment, and background color. If there is no selection, you can have BBEdit generate a "shell" for the table (including all <TR> and <TD> elements inside the table for the number of rows and columns you specify) and optionally insert placeholder "label" text in each cell.

Row

This command inserts <TR> and </TR> tags in the document, before and after the selection if there is one. You can specify the desired horizontal and vertical cell alignment and a row background color. If horizontal alignment to a specific character is specified, you can also indicate the character that determines alignment and the character offset to the first alignment character in the line.

TD, TH

These commands inserts a table data cell element or a heading cell element, respectively. (Both have the same options, though most browsers render TH elements differently from TD elements.) You can specify the width and height of the cell, the number of rows or columns it should span, its vertical and horizontal alignment (including alignment to a character and the offset to the first such character), whether the text in the cell should be permitted to wrap, the background color of the cell, and the scope of the header information in this cell, if any. You can also specify the axes, an abbreviated version of the cell's content, and which header cells contain information about the current cell. Many of the less familiar and infrequently used attributes have use in certain applications such as speech accessibility. To provide maximum accessibility for tabular data, we suggest you consult the HTML 4 specification.

Caption

This tag specifies a caption for a table. You may also optionally specify the caption's vertical alignment.

Colgroup, Col

These tags are used to define column and column groups. Browsers that understand HTML 4 tables can, for example, be told to format a number of columns the same way, or to place rules between column groups, using this construction. The contents of a column group may be one or more <COL> elements (or none at all, if the SPAN attribute is used). You can specify the span of the column or group, its desired width, and its vertical and horizontal alignment. Cells within this column group may inherit some or all of these attributes depending on the attributes of the individual <TD> or <TH> elements.

THead, TFoot, TBody

These tags define an optional table section element, which can be used independently of the <TH> tag. (The latter indicates that particular cells should be displayed in a heading "style", which is usually displayed by browsers as boldface.) <TH> may be used anywhere in a table that a "heading look" is desired. In contrast, these three related tags define the logical divisions of a table. Browsers might hold the table's header or footer fixed on the screen while scrolling a lengthy body up and down, for instance. All three tags allow you to select vertical and horizontal alignment, which may be inherited by cells inside the element depending on the attributes of <TR>, <TD>, and <TH> tags.

Convert to Table

This command provides a quick way to convert tab-delimited or comma-delimited lines of text to tables. You must specify the delimiter to be used (either tabs or commas), and you can optionally have the entire first row of the table or the first cell of each row converted to <TH> rather than <TD> elements. If One Cell Per Line is marked, each cell will be placed on its own line in the resulting HTML; otherwise cells will be placed on a single line.

Forms

This submenu contains commands that help you build HTML forms, which are used for accepting user data for processing by a client-side script or a server-side CGI program (or other server-side technology, such as Active Server Pages).

Form

This tag defines a form. The Method can be either GET (encoding the form data in the URL) or POST (sending the form data separately after the HTTP transaction header). The Action should be the URL of the CGI program (or other server-side script, such as ASP). Enctype and Accept-Charset define the encoding type and character set for the transaction (usually, you will not need to use these fields). Use the On Submit and On Reset fields to enter the names of JavaScript handlers to be used for the Submit and Reset buttons, respectively. The Target field sets the frame to be used for the page returned by the CGI.

Button

This tag creates a form button. Choose a type (Submit, Reset, or Button), specify a name and value for the form element, and set optional attributes such as Disabled, Tab Index (the order in which the button will be reached by the Tab key), and Access Key (the key the user can press to activate the button in the browser). (The latter two options are HTML 4 features and may not work on all popular browsers.) You can also enter the names of JavaScript onFocus and onBlur handlers for the button.

Field Set, Legend

In HTML 4, you can group your form's fields and other controls into sets of related fields by using the FIELDSET container. Within the FIELDSET container, the LEGEND tag is used to define a title for the field set. Browsers differ in how they represent field sets visually, but some browsers may draw a rectangle around the related controls as in dialog boxes. In this case the Align attribute of the LEGEND tag can be used to set the alignment of the legend relative to the visual representation of the field set. (Browsers that do not support these tags will ignore them, and the contents of the LEGEND container will be displayed as any other text.)

Input

This tag defines an input field, which can be a text or password input, various types of buttons, and even files, images or hidden fields. Specify the name and the default value of these fields, and, if applicable, their size, maximum length, tab index, access key, and disabled or read-only attributes. (Disabled, Read Only, Tab Index, and Access Key are HTML 4 features and may not be supported by all popular browsers.) You may also specify handlers for the JavaScript onFocus, onBlur, onSelect, and onChange handlers.

Label

HTML 4 allows you to specify that text next to a control is a Label, and in browsers that understand the tag, clicking the label associated with a button activates the corresponding control. BBEdit lets you create a <LABEL> tag, specifying the name of the control it should be associated with, an optional keyboard equivalent to activate the control, and onFocus and onBlur JavaScript handlers.

Select

This tag defines a scrolling list or pop-up menu. Enter the name of the control, the number of items to display (leave the size blank for a pop-up menu rather than a scrolling list), and whether the list allows multiple items to be selected. Optionally mark the control as disabled and specify onFocus, onBlur, and onChange handlers.

Option Group

Using the <OPTIONGROUP> tag, you can create submenus in pop-up menus in browsers that support them. All <OPTION> tags within an <OPTIONGROUP> container are displayed as items cascading from the specified submenu label. (In browsers that do not understand <OPTIONGROUP>, users will see a simple straight list of all defined options.)

Option

This tag defines an option in a pop-up menu or a scrolling list. Enter the desired label and value for the option, and mark the Selected checkbox to make the option the default or initial choice.

Text Area

This tag defines a scrolling text area field for entering large amounts of data. You can specify the name of the file, its size in rows and columns, and optional HTML 4-only attributes such as Disabled, Read Only, Access Key, and Tab Index. You can also specify script handlers for onFocus, onBlur, onSelect, and onChange events.

Inline Elements

Inline elements are HTML elements that can appear as part of a paragraph, such as anchors, images, applets, client-side scripts, image maps, and more.

Anchor

This command inserts an HTML anchor (<A>) tag. Anchors can either be hyperlinks or be used as the target of hyperlinks to provided multiple targets on a single page. The anchor must have an associated URL in the HREF field to be a link; it must have a name in the Name field to be a target. The Target field is used to specify which frame the linked page should appear in.

When specifying a URL, you can choose a recently used URL from the Recent URLs pop-up menu, or choose another file from the same folder as the current document using the Current Folder pop-up menu. If you hold down the Option key, the Recent URLs pop-up menu will turn into a pop-up menu containing Internal Anchors in the current document. If you click the File button, you will be prompted to choose a file from an Open dialog. Normally, all types of files will be displayed here. However, if you select the Only Show HTML and Image Files checkbox at the bottom of the Anchor dialog, you will only see images and files whose names match those defined as HTML in BBEdit's Languages preference panel.

Alternatively, you can drag and drop a file into the HREF field in order to insert its path and name.

BBEdit displays a warning message when the URL does not exist on your computer or is outside the scope of your site; these warnings can be disabled using the checkboxes at the bottom of the dialog.

Image

This command inserts an <IMG> tag to display an image. As with the Anchor tag, you can select the Source from the Recent URLs or Current Folder pop-up menus, by typing a URL, or by clicking the File button. If the Canto Cumulus image cataloging tool is running on your computer, you can also click the Cumulus button to choose an image from Cumulus's catalog, or you can use drag and drop.

After choosing an image file, you can specify alternate text (which will appear in browsers that do not support images or for users who are surfing with image-loading turned off), enter the Size of the image, select the amount of horizontal and vertical Space for wrapping around the image, and choose the thickness of the border and the image's alignment. (Image height and width should be specified whenever possible to speed layout of the page in the browser; BBEdit will enter these values for you automatically when you choose an image file.) You can also mark the Is Map checkbox to use a server-side image map or the Use Map checkbox to use a client-side image map embedded in the HTML document.

Applet

This command inserts the <APPLET> tag for specifying a Java applet. You will need to specify the location the folder that contains your main Java class file (the codebase) as well as the name of the main class file. If the file is in a .ZIP or .JAR archive, you can specify its name here as well. If you will control the applet via a client-side script, enter a name for it. You should always enter the desired size for the applet's display area. You can also specify alignment and white space around the applet, along with ALT text to be displayed if the applet cannot be used.

Object

The <OBJECT> tag is a generic tag for including almost any type of data in a page, including images and Java applets. (It can also be used to insert ActiveX controls and data intended to be used by plug-ins.) However, it is an HTML 4 tag and may not be supported in all popular browsers. For this reason we suggest using <IMG> and <APPLET> for those types of objects and use <OBJECT> only for embedding other types of data, such as that used by plug-ins. For an example of this, see the Web Design Group's HTML Help reference page:

```
http://www.htmlhelp.com/reference/html40/special/object.html
```

The <OBJECT> tag, like the <IMG> and <APPLET> tags, allows you to reserve screen space in the browser window, recommend an amount of white space between the object and surrounding text, align the object, set its border, specify alt text to be displayed if the object cannot be displayed, and so forth. You will also need to specify at least the codebase and class ID of the object for ActiveX controls, and fill in the Data field for embedded objects such as Shockwave animations which will be handled by plug-ins. The Standby field can be used to tell browsers a text message to be displayed while the object is loading. For more information on the <OBJECT> tag, consult the HTML 4 specification.

Param

To pass parameters to a Java applet, ActiveX control, or plug-in, the <PARAM> tag can be used between the <OBJECT> and </OBJECT> (or <APPLET> and </APPLET>) tags. Each parameter to be passed to the object requires a separate <PARAM> tag. You must specify the name and value of each parameter; the actual parameter names and values required will vary depending on the object being embedded.

Script

This tag begins a section of client-side script code (by default, JavaScript, although some browsers support other scripting languages). You can choose to execute a script contained in an external file by entering a URL in the Source field (click File to choose the file using an Open dialog). You can also enter values for the TYPE of script, the script LANGUAGE it is written in, and the character set or CHARSET of the script. Mark the DEFER checkbox to add a DEFER attribute.

Note You can choose which character sets appear in the Charset pop-up menu by using the Text Encodings panel of the Preferences window.

Map

This tag embeds a client-side image map in the document. You must enter a name by which the map can be referenced in the Use Map attribute of the Image tag. Individual clickable areas within the image map are provided by the <AREA> tags inserted between the <MAP> and </MAP> tags.

Area

This tag defines a clickable area within a client-side image map. Each clickable area requires a separate <AREA> tag. You will need to specify the document to be loaded when the area is clicked (or mark the No HREF checkbox to cause clicks in the area to be ignored), along with its Target frame if the page is being used in a frameset. You can choose the desired map shape (rectangular, circular, polygonal, or the default URL) using the Shape pop-up menu and enter the desired coordinates of the shape in a comma-separated list in the Coords field. (For rectangles this is in the order left, top, right, bottom; for circles it is in the order X, Y, radius. For polygons this should be a comma-separated list of coordinates in X, Y form.) You can also set the tab index of the field for keyboard control on browsers that support it. JavaScript onFocus and onBlur handlers are also supported.

Convert to Client Side Map

This command converts the selected text, which should be a server-side image map file in NCSA server format, to a client-side image map. You must specify a name for the converted imagemap.

Break

This command enters a line break tag,
, into the document. If multiple lines are selected, a line break tag will be inserted after each.

Font

This tag selects the font, size, and/or color for the selected text. This tag is deprecated and should generally not be used; stylesheets are a more flexible and more content-oriented way of achieving this end.

Base Font

This tag selects the default font, size, and/or color for the text in this document. Like <FONT>, this tag is deprecated; it is considered better form to use stylesheets.

Bidirectional Override

This command inserts a <BDO> tag to note that the enclosed text is in a language that should be rendered in a different direction (either left-to-right or right-to-left) than the default text order for the document's primary language. You can specify the desired text order and the language, so that savvy browsers can switch fonts or script systems to display the text correctly.

Quotation

This command marks the selected text as a quotation. Use this only for short quotes within a paragraph; use <BLOCKQUOTE> for quotations consisting of a paragraph or more of text.

Span

This command marks the selection as belonging to a certain class of information—such as a book title—usually so that its text style can be retrieved from a stylesheet. (In contrast with <DIV>, which marks paragraph-level classes, <SPAN> marks character-level classes.) You will be prompted for an ID for this span, a class name (which should correspond to a stylesheet entry), and inline style information. All are optional.

The Span command can now create nested span elements. This means that in order to edit an existing span element (since they can be nested), you must place the insertion point within the open tag of the desired instance.

Subscript

This command marks the selected characters as a subscript (lowered below the baseline).

Superscript

This command marks the selected characters as a superscript (raised above the baseline).

Phrase Elements

Phrase elements are HTML tags that mark sentences or phrases within a block element (such as a paragraph) with certain content-related styles, such as emphasis, strong emphasis, citation, and so on. Indirectly this determines the displayed format of the enclosed text (although exactly what "emphasis" and so on mean is left up to the browser or the stylesheet).

Abbreviation

The enclosed text is an abbreviation.

Acronym

The enclosed text is an acronym.

Citation

The enclosed text is a citation of another document.

Computer Code

The enclosed text is computer source code.

Deleted Text
This command inserts a block formatted to indicate that the enclosed text has been deleted (usually with a horizontal line through it—that is, "struck out"). You may optionally specify a citation (indicating a reference to another file) and a date and time.

Defined Term
The enclosed text is term defined in a glossary.

Emphasis
The text should be displayed with visual emphasis (most browsers interpret this as italic text).

Inserted Text
This command inserts a block formatted to indicate that the enclosed text has been inserted (usually by underlining the text). You may optionally specify a citation (indicating a reference to another file) and a date and time.

Input Text (Kbd)
The enclosed text is text to be entered on a computer keyboard (used in instructions).

Sample Output
The enclosed text is sample output from a computer program (used in instructions).

Strong Emphasis
The text should be displayed with strong emphasis (most browsers interpret this as boldface).

Variable
The text is a placeholder in an instruction or tutorial, and should be replaced with an actual value of the appropriate type before actually performing the indicated operation.

Font Style Elements
Like Phrase Elements, Font Style Elements mark relatively short pieces of text within a block element. However, they are concerned more with the appearance of the text than its structural function in the document.

Big
This displays the enclosed text in a larger font than usual.

Small
This displays the enclosed text in a smaller font than usual.

Bold
This displays the enclosed text in boldface type.

Italic
This displays the enclosed text in italic type.

Strike-Through
This displays the enclosed text in a strike-through style.

Teletype Text
This displays the enclosed text in a monospaced font, as on a computer terminal or teletype.

Underline
This displays the enclosed text in an underlined style.

Frames
The commands in the Frames submenu help you design documents that use frames. The first document loaded by the browser contains at least one <FRAMESET> tag and one or more <FRAME> tags, which specify the number and sizes of the desired browser window subdivisions and indicate the URLs of the files to be loaded into each.

Frame Set
This defines a frame set, a series of one or more frames. You indicate whether the frame set divides the browser window vertically (ROWS) or horizontally (COLS), and then indicate the size of each frame in a comma-separated list, using * to tell the browser to use whatever space is left over from the other specified frames.

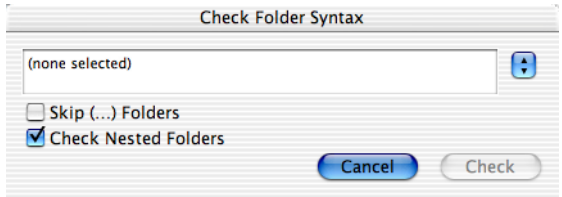
Frame sets can be nested. For example, if you want to create a framed Web page with three rows, with the middle row divided into two independent columns, you would first define a frame set consisting of three rows. Instead of defining the second row with a <FRAME> tag, however, you would open another <FRAMESET> tag there, this time to specify the two columns for the middle frame (which would then be specified by <FRAME> tags).

Frame
This defines a frame in a frame set document. You will need to specify the URL of the file to be displayed in this frame (either using the button, or by drag and drop). If the frame will be targeted by links in another frame, you will also need to give the frame a name. You can optionally specify a long description for the frame, choose whether the frame can be scrolled, and indicate whether the user should be able to resize the frame. You can also set margins and borders for the frame. (Borders are the visible lines between frames. Margins determine how far each frame's content appears from its border or from the window edge.)

No Frames
HTML markup included between the <NOFRAMES> and </NOFRAMES> tags is displayed by browsers that do not support frames. This is where you should include a link to a non-frame (or text-only) version of the site. Although many current browsers support frames, some users do not like the feature and intentionally disable it in their browser.

Check
The Check submenu contains utilities for finding errors in your HTML markup and document links. You can run syntax or link checks on the current document, a specified folder, or the current site (as defined in the HTML Web Sites preference panel). You can also perform a Balance Tags operation on the current document.

When you are checking a folder or a site using these tools, a dialog like the one below appears.



The pop-up menu to the right of the path box includes all of the web sites that you have specified in the HTML Web Sites preference panel; choosing any of these selects its designated site root folder for checking. The Other entry on the pop-up menu displays an Open dialog, allowing you to navigate to and choose any other desired folder. (You can also drag a folder from the Finder directly into the path box.) The Skip (…) Folders checkbox specifies that subfolders whose names are enclosed in parentheses should not be checked; Check Nested Folders indicates that nested folders should be included in the check.

If a Check Syntax or Check Links operation generates any errors or warnings, BBEdit will display an error results browser listing. For more details on the error results browser format, please refer to Chapter 9.

Syntax

This command invokes BBEdit's syntax checker, which validates your HTML document to the specification defined in the <!DOCTYPE> SGML prolog at the top of the document. Errors are displayed in an error results browser. Scroll through the list at the top of the window to see the errors that have been found; click to see the text that caused the error in the lower part of the window. Double-click an error message to open the file for editing.

Links

This command causes BBEdit to scan your document, or a folder of documents, looking for links and object references (such as images and Java applets) that cannot be resolved. Note that BBEdit only looks at pages on your site as defined in the HTML Web Sites preference panel, not at any links that go offsite. (BBEdit will, in fact, generate warnings for offsite links unless you disable the Remote Links warning in the HTML Tools preferences.)

Update

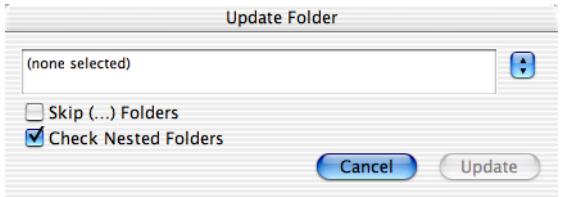
The Update submenu contains commands for updating IMG tags, includes, and placeholders in the current document, the selected folder, or the current site (as selected in the HTML Web Sites preference panel). BBEdit displays a results browser after the operation so you can see what was changed.

For more details on the results browser format that BBEdit displays, please refer to Chapter 9.

Choose the Document, Folder, or Site command from the Update submenu to update all includes and placeholders in the selected set of file(s). (Appendix C contains information regarding the use of placeholders.)

Choose the Document Images, Folder Images, or Site Images command from the Update submenu to update the HEIGHT and WIDTH attributes of image tags (and optionally to insert empty ALT attributes when missing) in the selected set of file(s).

When you are updating a folder or a site using either of these sets of tools, a dialog like the one below appears.



The pop-up menu to the right of the path box includes all of the web sites that you have specified in the HTML Web Sites preference panel; choosing any of these selects its designated site root folder for updating. The Other entry on the pop-up menu displays an Open dialog, allowing you to navigate to and choose any other desired folder. (You can also drag a folder from the Finder directly into the path box.) The Skip (…) Folders checkbox specifies that subfolders whose names are enclosed in parentheses should not be updated; Check Nested Folders indicates that nested folders should be included in the update.

Includes

The Includes submenu contains commands for inserting one-time include directives, "persistent" include directives, and placeholders. See Appendix C for more information.

Choosing Include or Persistent Include prompts you to choose a file using an Open dialog and then inserts the appropriate markup. Choosing Placeholders displays a scrolling list of available placeholders; selecting one and clicking Insert places it into the document.

Utilities

The Utilities submenu contains commands for automatically editing the current HTML document for ease of editing and for consistency.

Format

This command formats the current HTML document for easier editing. The appearance of the document in a browser is generally not affected (except in the case of Document Skeleton). You can choose from among the following:

- Hierarchical: Nested HTML structures are indented

Note This command may add white space, which can affect display in a browser

- Gentle Hierarchical: Same as Hierarchical, with less indenting
- Plain: Places each tag on a separate line with no indenting
- Compact: Absolute minimum white space

- Gentle Compact: A slightly more human-readable version of Compact
- Source Format Profile: Formats the document according to the Dreamweaver Source Format Profile selected in the HTML Markup preference panel (if any)
- Document Skeleton: A hierarchical view with all non-tag content removed
- Don't Reorganize: Allows normalizing of case, quote, and entity settings, as well as encoding entities within attributes, without otherwise changing the existing structure

You can also have the Format command operate on the whole document or only the selection, normalize the tags to uppercase or lowercase (according to your settings in the HTML Tools preferences), normalize quote marks around attributes, and encode special characters, or entities, found in attributes.

If you choose the "Format…" command (with the ellipsis), BBEdit displays a dialog allowing you to choose the formatting options. If you choose the Format command (without the ellipsis), BBEdit uses the previous options.

Optimize

This command reformats the document to use the absolute minimum of characters while remaining syntactically valid. You will have difficulty editing your document in this format (in fact, if you do not have Soft Wrap turned on in the text options, you might think most of your document has vanished, because the command strips out all line breaks), but rest assured that your document will appear the same in your browser as it always has. Use one of the Format commands discussed above to put your page back into an editable format if you need to make changes. This command also applies the various Cleaner tools automatically.

Translate

This command allows you to translate plain text to HTML or vice versa. The dialog that appears gives you flexible options for translating Macintosh characters to HTML entities, converting paragraphs to <P> tags and vice versa, removing HTML tags entirely, and so on. You can choose to have the results displayed in a new document window and to convert only the selection rather than the whole document. See "HTML Translation" later in this chapter for more information.

Remove Comments or Markup

This command removes all HTML comments or HTML tags, respectively, from the selection. Note that removing comments will not remove comment markers around client-side scripts like JavaScript, where they are required for proper functioning of the page on older browsers, but will remove the comment markers used by placeholders and indexes, making these items difficult to update in the future.

Comment, Uncomment

Comment adds comment markers (<!-- -->) around the selected text, regardless of whether it already contains tags or not, causing the browser to ignore that text. (It also converts any previously existing comment markers in the selection range to use ~~ in place of --, to prevent older browsers from misinterpreting the comment.) Uncomment removes all comment tags from the selection and converts any modified comments back to real HTML comments.

Normalize Tag Case, Make Tags Upper or Lower Case

These commands convert all HTML tags in the document to the desired case: upper, lower, or normalized (which is either upper or lower depending on your HTML Markup preference settings).

Misc

The Misc submenu contains a motley assortment of commands that simply did not seem to fit anywhere else in the command hierarchy, including commands related to using BBEdit with other HTML editors.

Dreamweaver

Choose Dreamweaver to open the current HTML document in Macromedia Dreamweaver (or to return to Dreamweaver if you opened a page in BBEdit from there).

Document Size

This command displays a report about the document's size and the amount of time it will require to download at various connection speeds.

Index Document

This command generates a list of links to all the <A> (anchor) tags in the current document that have a NAME attribute, providing a clickable index of all these anchors. If you have used the <A> tag to mark your main topics (for example, all <H2> headers) this produces an instant topic-level index of the current document. The index is placed after the <BODY> tag, unless the insertion point is in an already existing index, in which case the old index is replaced with the new one. (Do not remove the comment markers around the index if you want to be able to update it in the future.) Index Document also adds <A name> tags to all heading tags that do not already have them.

Index Folder

This command works like Index Document but produces an index of the HTML files within a specific folder. You can choose from five different styles of indexes, including one that provides full information about every referenced file. The best way to see what each format of index looks like is to experiment for yourself.

Index Site

This command works like Index Folder, except that it always creates an index of the HTML files within the root folder of the default web site, as set in the HTML Web Sites preference panel.

GoLive, Home Page, or PageMill Cleaner

These commands clean up the often gnarly HTML code created by these WYSIWYG editing tools. This will make the file easier to edit, smaller, and often slightly quicker to load into a browser. The GoLive cleaner removes font tags with no semantic value—that is, tags which contain nothing at all, or just white space. The PageMill cleaner removes the "NATURALSIZEFLAG" attribute that PageMill puts in <IMG> tags, and turns every two consecutive line break tags into paragraph (<P>) tags.

Tidy

The Tidy commands provide various options for reformatting and cleaning up your HTML documents. To perform these commands, BBEdit makes use of code from the HTML Tidy Library project.

`http://tidy.sourceforge.net/`

Clean Document

This command presents a dialog in which you may select various options for modifying the frontmost document's tags. The available options are:

- Remove Bogus Markup: This option controls whether BBEdit should strip out surplus presentational tags and attributes, replacing them by style rules and structural markup as appropriate. (This option works well on the HTML saved by Microsoft Office products.)
- Insert Blank Line before
: This option controls whether BBEdit should output a line break before each
 element.
- Discard Empty Paragraphs: This option controls whether BBEdit should discard empty paragraphs. If you do not set this option, BBEdit will replace each empty paragraph with a pair of
 elements as HTML4 precludes empty paragraphs.
- Discard Comments & Optional Tags: This option controls whether BBEdit should remove all comments and optional tags.
- Enclose Block Text: This option controls whether BBEdit should insert a <P> element to enclose any text it finds in any element that allows mixed content for HTML transitional but not HTML strict.
- Enclose Text: This option controls whether BBEdit should enclose any text it finds in the body element within a <P> element. This is useful when you want to take existing HTML and use it with a style sheet.
- Escape <CDATA>: This option controls whether BBEdit should convert `<![CDATA[]]>` sections to normal text.
- Give Tidy Credit: This option controls whether BBEdit should add a meta element to the document head to indicate that the document has been tidied. (BBEdit will not add such a meta element if one is already present.)

Reflow Document

This command presents a dialog in which you may select various options for reformatting the frontmost document's tag structure. The available options are:

- Input is XML: This option controls whether BBEdit should use the Tidy XML parser rather than the error-correcting HTML parser.
- Indent Block-Level Tags: This option controls whether BBEdit should indent block-level tags.
- Indent Attributes: This option controls whether BBEdit should begin each attribute on a new line.
- Wrap ASP: This option controls whether BBEdit should wrap text contained within ASP pseudo elements, which look like: `<% ... %>`.

- Wrap Attributes: This option controls whether BBEdit should wrap attribute values, for easier editing.
- Wrap JSTE: This option controls whether BBEdit should wrap text contained within JSTE pseudo elements, which look like: `<# ... #>`.
- Wrap PHP: This option controls whether BBEdit should line wrap text contained within PHP pseudo elements, which look like: `<?php ... ?>`.
- Wrap Script Literals: This option controls whether BBEdit should wrap string literals that appear in script attributes. BBEdit will wrap long script string literals by inserting a backslash character before the line break.
- Wrap Sections: This option controls whether BBEdit should line wrap text contained within `<![ ... ]>` section tags.
- Give Tidy Credit: This option controls whether BBEdit should add a meta element to the document head to indicate that the document has been tidied. (BBEdit will not add such a meta element if one is already present.)

Convert to XHTML

This option instructs BBEdit to convert the current contents of the frontmost document to XHTML. If no DOCTYPE or namespace was present, BBEdit will set them as appropriate. If a DOCTYPE or namespace was given, it will checked for consistency with the content of the document, and in the event of an inconsistency, the corrected values will appear in the output. Entities will be written using the named form, and the original case of tags and attributes will be preserved.

Convert to XML

This option instructs BBEdit to convert the current contents of the frontmost document to well-formed XML. Any entities not defined in XML 1.0 will be written as numeric entities to allow them to be parsed by a XML parser, and the original case of tags and attributes will be preserved.

Preview

The Preview commands provide various options for previewing your HTML documents in a web browser.

Preview in BBEdit

Choosing this command will open a live content preview window within BBEdit which is linked to the document that was frontmost when you chose the command. You can go back from the preview window to its corresponding source document by clicking on the document icon button in the preview window, or by choosing the Show Document command from the Markup menu.

The preview window uses the same content rendering engine as Safari, and is updated whenever you modify the document. Closing the document will also close the preview window. (You can of course have multiple preview windows open on multiple documents.)

The preview window will not automatically display changes made in any related files, such as images or linked CSS files. However, you can use the Refresh BBEdit Preview command (see below) to update the preview window's display of both the source document and all related files.

Refresh BBEdit Preview

This command works in conjunction with the Preview in BBEdit command. How it behaves depends on the situation in which it's invoked:

- If the front window is a BBEdit Preview window, its associated HTML file will be reloaded, together with any related files which were changed behind the preview window's back (e.g. images, linked CSS files).
- If the front window is a text document, and there exists a preview window previously created by a "Preview in BBEdit" command on that document, then the associated preview window will be reloaded.
- If the front window is a text document, and any preview windows are open, the frontmost preview window will be reloaded, even if it does not necessarily belong to the front document.

BBEdit Preview windows also contain a “Reload” button, which has the same effect as this command. Finally, whenever you save a CSS file, BBEdit will automatically refresh all open BBEdit Preview windows.

Preview in <Default Browser>

This command will display the name of the current default web browser, and choosing it will cause BBEdit to display the frontmost document in that browser.

You can choose a browser from the Preview With submenu, or use Preview by itself to use the last chosen browser. You can also preview the page in all running browsers or in a text-only format. (On Mac OS X, browsers running in the Classic environment are labeled “(Classic)”.)

Preview With

The Preview With command provides a submenu listing all installed web browsers and versions. The default browser, i.e. the one which BBEdit will use unless you choose otherwise, has a checkmark listed next to its name, while the names of any browsers that are currently running will be underlined. (You may add browsers which are not listed by using the HTML Preview Preferences panel.)

You can preview the frontmost document in any available browser by choosing it from the menu. Making such a choice will also cause BBEdit to use that browser as the default until another is selected.

Alternatively, you can choose the Preview as Text item to generate a text-only rendering within BBEdit, or the All Running Browsers item to preview the current document in all running browsers.

The HTML Tools Palette

The main HTML Tools Palette is the place from which you will probably access the HTML Tools most frequently. You can invoke the HTML Tools palette at any time by selecting it from the Palettes submenu in the Window menu. BBEdit remembers which palettes you had open when you quit, so if you open the HTML Tools palette, it will remain open until you close it again, even on subsequent uses of BBEdit.

You can choose the buttons to be displayed on HTML Tools palette in the HTML Palette preference panel.

HTML Tools Palette Tips

A list of all the tools that are available on the HTML Tools palette appears below. In most cases, their behavior corresponds obviously with the tool descriptions in the previous section. In the few cases where there are significant differences, these are noted.

Many of the tools also work with drag and drop. Those that do are marked with "grip-strips"—two vertical slashes at the left side of the button. You can drag these tools into your document to have them take effect wherever you drop them.

Some palette buttons are actually pop-up menus. Clicking a pop-up menu button while holding down the Command key invokes that menu's last-selected menu option.

Try pressing the Option key and watch how the palette changes. By holding down Option as you click, you can force many buttons that normally display a settings dialog to use their previous settings instead. The Preview button previews your document in a new browser window when you Option-click.

HTML Tools Palette

Tool	Menu-Based Equivalent
New Document	File > New > HTML Document
Tag Maker	Markup > Tag Maker
Edit Tag	Markup > Edit Tag
Close Current Tag	Markup > Close Current Tag
Balance Tags	Markup > Balance Tags
Document Type	Markup > Document Type
Character Set	Markup > Character Set
CSS	Markup > CSS submenu
Body Properties	Markup > Head Elements > Body Properties
Head Elements	Markup > Head Elements submenu
Anchor	Markup > Inline Elements > Anchor
Image	Markup > Inline Elements > Image
Break	Markup > Inline Elements > Break
Font	Markup > Inline Elements > Font
Paragraph	Markup > Block Elements > Paragraph
Div	Markup > Block Elements > Div
Heading	Markup > Block Elements submenu

Tool	Menu-Based Equivalent
Dreamweaver	Markup > Misc > Dreamweaver
Block Elements	Markup > Block Elements submenu
Inline Elements	Markup > Inline Elements submenu
List	Markup > Lists > List, Markup > Lists submenu
Table	Markup > Tables > Table, Markup > Tables submenu
Forms	Markup > Forms submenu
Phrase Elements	Markup > Phrase Elements submenu
Font Style Elements	Markup > Font Style Elements submenu
Frames	Markup > Frames submenu
Check Syntax	Markup > Check > Document Syntax
Check Links	Markup > Check > Document Links
Update Document	Markup > Update > Document
Check	Markup > Check submenu
Update	Markup > Update submenu
Includes	Markup > Includes submenu
Utilities	Markup > Utilities submenu
Misc	Markup > Misc submenu
Preview in BBEdit	Markup > Preview in BBEdit
Preview	Markup > Preview in submenu
Preview (all browsers)	Markup > Preview in > All Running Browsers

Other Palettes

In addition to the main HTML Tools palette, BBEdit incorporates a number of other palettes that may be useful to HTML authors.

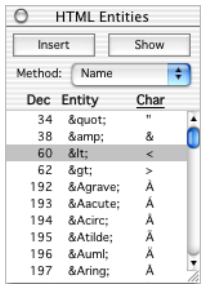
First, several of the pop-up menus on the HTML Tools palette can be used to open floating palettes for specific tools. At the bottom of the pop-up menus for CSS, Block Elements, Inline Elements, Table, Forms, Phrase Elements, Font Style Elements, and Utilities is a menu item that opens the specified menu as a palette. For example, choosing Inline Palette from the bottom of the Inline Elements pop-up menu opens a new palette containing a button for each of the items on that menu.

You probably will not need to open all these "subpalettes" at once, but they can be quite convenient to open temporarily when you are working on a particular type of element (form, table, and so on). In addition to being able to open them from the main HTML Tools palette, you can also access each palette from the Palettes submenu in the Window menu.

Two other palettes are also of particular interest to Web authors: the HTML Entities palette and the Web Safe Colors palette.

HTML Entities

In HTML, extended characters must be encoded as entities, since different computers define the extended ASCII characters differently. The HTML Entities palette lists these entities.



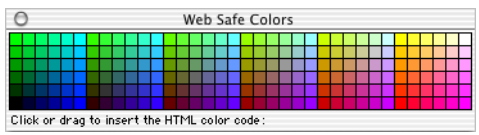
Entities can be inserted by name ("©" is the copyright symbol, ©) or number ("&169;" for ©) by choosing the desired method from the small pop-up menu at the top of the HTML Entities palette. (We suggest inserting entities by name, since they are more readable, unless browser compatibility requires use of the decimal versions.)

Double-click an entity name to insert it into the active document, or click once to select the desired entity and then click Insert.

The list of entities presented in the HTML Entities palette is now sortable by decimal value, name (case-insensitive, so "&ecaute;" and "É" are grouped together), or character (sorted by the character position after all diacriticals have been stripped, so that all "a"s are grouped, and so on). Click on a column label to set the sort order accordingly. (The default is to sort by decimal value.)

Web Safe Colors

The Web Safe Colors palette displays the 216 colors that display properly in Web browsers on all computers running in 256-color mode. The colors are based on a 6 x 6 x 6 color cube that adds red, green, and blue in 20% increments.



Use a color from this palette for Web page backgrounds and text to reduce the chance that the incorrect color (possibly dithered) will be used instead of a pure color.

Note There are four layouts available: horizontal, vertical, and VisiBone and VisiBone 2. You can choose a layout in the HTML Colors preference panel.

Click a color swatch to insert the color's RGB value, in HTML format (that is, "#RRGGBB") at the insertion point. You can also drag a color swatch into a document window to place the color value.

The Web Safe Color palette is CSS-aware, and will insert color values in CSS format—that is, unquoted and minimized—when appropriate.

HTML Translation

The following options are available in the Translate tool, available in the Utilities submenu of the Markup menu.

Remove Tags

When converting HTML to TEXT, the Translate tool will remove all HTML tags and comments.

Paragraphs

When converting TEXT to HTML, the Translate tool finds paragraphs in the same way the Paragraph tool does, and then adds opening and closing paragraph tags around them.

When converting HTML to TEXT, it makes sure there are line breaks around each paragraph in the resulting text.

HTML Entities

When converting TEXT to HTML, the Translate tool converts characters from their positions in the standard Macintosh character set into HTML entities, using either names or the code (in decimal or hexadecimal). You can specify whether the tool should ignore < and >. This is useful when translating text already marked up as HTML. You can also specify that all Unicode text should be converted to entities.

Templates

In addition to providing many facilities for creation and markup of individual documents, the HTML Tools also incorporate a Template facility, which can be used to quickly create (or revise) a set of HTML documents that share a common format, structure, or content. You can design a skeleton document, make a template from it, and then use that template over and over again to produce new pages ready to fill with content, or to insert into existing text documents to provide an uniform structure or appearance. Templates may also employ placeholders and include files (see Appendix C), adding even more power to this useful function.

Template Setup

A folder named HTML Templates, which contains some sample templates, is provided as part of the standard BBEdit installation. We suggest, in most cases, that you continue to use this as your templates folder. If you would prefer, however, you can set up your own template folder wherever you like. (See “Look for Templates and Include Files In” on page 183.) If you plan to maintain multiple sets of templates for different projects, you may find this option very useful.

Using a Template

A template is a simple text file that contains boilerplate text or HTML content that will form the foundation for the document you are creating. Template files must have the file name suffix ".tmpl" in order to be recognized.

When creating a template file, you can convert or reuse an existing document, or you can write one from scratch. Simply rename the file by adding the suffix ".tmpl" to it, and then move or copy it into your active HTML Templates folder.

Templates are always invoked using the New Document tool. Even if you want to impose a template on an existing document, you still do it through New Document. The Template option appears as a pop-up menu at the bottom left of the New Document dialog.

All template files in the Templates folder appear in this menu. (The "Default" setting is not a template per se, but rather a directive to create a blank HTML document framework containing whatever Title, Base, Meta, Link, SGML Prologue (and so on) values you specify. It is always available, regardless of the contents in your Templates folder.) Once you have specified the appropriate settings and chosen Create, BBEdit will open an new Untitled window containing the full text of the selected template file.

If you invoke the Document tool while working on an existing document, you will see a Create New Window checkbox next to the Template menu. Normally this option will be on, but if you deactivate it, the specified template (and other applicable Document values) will be inserted into the currently open document, so as to enclose the document's contents within the template's <BODY> tags. The contents of the document are inserted at the template's #BODYTEXT# placeholder, so each template to be used in this manner must contain this placeholder.

Regardless of whether you use a template to create a new document or to impose HTML markup on an existing document, the template file itself is never changed. Rather, its contents are simply copied into the document that the template has been used on.

Note Templates can make full use of placeholders and include files, which are fully documented in Appendix C.

CHAPTER

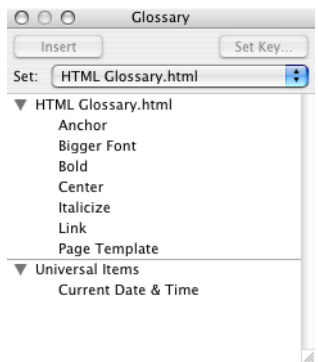
12 Using the Glossary

This chapter describes BBEdit's powerful Glossary command. The Glossary provides an easy way to store and access frequently used text of any sort, whether sections of program code, HTML markup, or just about anything else. The Glossary's language-sensitive set selection and its ability to perform keyword substitution, combined with an option to run scripts and insert their results, further extends its flexibility and usefulness.

In this chapter

The Glossary Command

Choosing the Glossary command from the Palettes submenu of the Window menu opens the Glossary palette, shown below. This window lists the contents of the active glossary set, plus the contents of the Universal Items glossary set and any glossary items contained directly within the Glossary folder (that is, which do not belong to a set). 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.

To create a glossary item, type or paste the desired text, or text and keywords, into a BBEdit document window and then save the text into an appropriate place within the Glossary folder.

The standard location of the Glossary folder is "~/Library/Application Support/BBEdit/Glossary/". (For more details, see "BBEdit's Application Support Folders" on page 12.) You can create multiple levels of subfolders inside the Glossary folder, to better organize different types of content. The first level of such subfolders appear in the Set pop-up menu of the Glossary window, allowing you to reveal only the group of glossary items you wish to work with at a given time. (Any glossary items that are not placed in a subfolder are always shown in the Glossary window, as are the contents of the Universal Items set.)

Language Sensitivity of the Glossary

If you have selected the Glossary Is Language Sensitive option in the Glossary preference panel, BBEdit will select a glossary set to use with the current document according to the following rules:

- If there is a glossary set whose name ends in the same suffix as the document, that set will be selected.
- If there is a glossary set with the same language mapping as the document (as determined from the mappings in the Languages preference panel), that set will be selected.
- If there are multiple glossary sets with the same language mapping (for example, "HTML Glossary.html" and "HTML Glossary.shtml"), the glossary set that appears first in the Set pop-up menu will be selected. (You can control the order in which glossary sets appear as described in the next section, "Manually Sorting the Glossary.")

These selection rules are intended to provide maximum flexibility while automatically doing the right thing as often as possible.

Manually Sorting the Glossary

By default, the Set pop-up menu displays glossary sets 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)Web template" would sort before "01)HTML Template". The first three characters of such names are not displayed in the menu. You can also insert a divider by including an empty folder whose name ends with the string "-***". (You can use anything you want for the rest of the name, to make it appear where you want it in the menu.) These conventions are the same as those used by the utilities FinderPop and OtherMenu.

Inserting Glossary Items

Double-clicking an item's name in the Glossary window inserts its contents at the insertion point or in place of the current selection. Alternatively, you can click the item's name once to select it and then click the Insert button, or drag the item directly to the desired location in the document window.

Note If you hold down the Option key, the Insert button changes to Edit, allowing you to open the file corresponding to the selected item so you can edit it; you can also do the same thing by Option-double-clicking the item directly.

The Insert Glossary Entry command in the Edit menu brings up a dialog window in which you can select a glossary item and insert it at the insertion point (or in place of the current selection) with the Insert button. You can move quickly to an item in this window by typing its first few letters from the keyboard, or by using Tab and Shift-Tab to navigate forward and backward through the list.

Another way to insert a glossary item is with the Auto-Complete Glossary command in the Edit menu. If there is a current selection, BBEdit will look for and insert a glossary item whose name begins with the selected text. (If there is more than one such item, BBEdit will display the Insert Glossary Entry dialog to allow you to choose one; if there are no matching glossary items, it will beep to signal an error.) If no text is currently selected, BBEdit will scan backward from the insertion point to the nearest non-alphanumeric character and match that text in the same manner. This allows you to quickly insert a glossary item "on the fly" by typing the first few letters of its name and choosing Auto-Complete Glossary from the menu.

IMPORTANT To make the most of "Insert Glossary Entry" and "Auto-Complete Glossary", we recommend that you assign key equivalents to these commands by using the Set Menu Keys command on the BBEdit menu.

Assigning Key Equivalents to Glossary Items

The Set Key button in the Glossary window lets you assign key equivalents for easy access to frequently used glossary items. To assign a key to a glossary item:

1 **Select the item in the Glossary window.**

2 **Click the Set Key button to display the Set Key dialog.**

3 **Type the key equivalent.**

You can use any combination of the Command, Shift, Option, and Control keys in the key equivalent, provided that it must use at least the Command or Control key to be valid. You can also use function keys, with or without additional modifiers.

4 **Click OK.**

Note If you try to assign a key equivalent that is already used elsewhere, BBEdit warns you that there is a conflict and asks you whether you want to reassign that key equivalent to the new item.

To remove a key equivalent from a glossary item:

1 **Select the item in the Glossary window.**

2 **Click the Set Key button to display the Set Key dialog.**

3 **Click Reset, then OK.**

Glossary Substitution Placeholders

When you insert a glossary item containing a placeholder into an editing window, BBEdit replaces the placeholder with appropriate substitution text. This is similar to the operation of BBEdit's HTML Templates and Update features. The following table shows the placeholders you can use in a glossary item:

Placeholder	Replaced by…
#BASENAME#	The name of the file stripped of its rightmost period-delimited portion. For example, if the file is named "test.html", the base name is "test", while if the file is named "test.foo.html", the base name is "test.foo".
#CLIPBOARD#	Contents of the current clipboard
#DATE#	Current date, formatted according to your Format settings in the International panel of the System Preferences
#FILE#	File name of the document into which the item is inserted
#FILE_EXTENSION#	The filename extension for the file (determined as the rightmost period-delimited portion of the filename, without the period). For example, whether the file is named "test.html" or "test.foo.html", the filename extension is "html".
#FUNCTION#	If the item is being inserted into a source file, the name of the current function
#GMTIME XXX#	The current GMT time formatted according to the parameters XXX (see "Time Formats" below)
#INDENT#	When used in a glossary item with multiple lines, causes every line after the first to be indented to the same whitespace level as the line in which the item was inserted (see the supplied WML glossary for examples)

Placeholder	Replaced by…
#INLINE#	Strips all trailing vertical white space from the item before insertion (see also the Glossary preference panel)
#INSERTION#	Marks the place where BBEdit will place the insertion point after inserting the item; if multiple #INSERTION# placeholders are used, the second and subsequent occurrences are replaced with a placeholder "<##>", which can be used with Go to Next/Previous Placeholder in the Search menu
#LOCALTIME XXX#	The current local time formatted according to the parameters XXX (see "Time Formats" below)
#NAME#	The long name of the active user account. (There is no intrinsic placeholder for the short name, but you can use #inline##system whoami# to obtain it.)
#SCRIPT filename#	Result of running the specified AppleScript
#SELECT#	Selected text
#SELSTART# and #SELEND#	Mark a range within the inserted material to be selected after the insertion. You can use multiple pairs of these placeholders within a single glossary item.
#SYSTEM shell_script#	Given the full path to a shell command or script, BBEdit will run that command or script and insert the result.
#TIME#	Current time, formatted according to your Format settings in the International panel of the System Preferences
#UUID#	A 128-bit UUID (universally unique identifier), formed by combining a value unique to the computer on which it was generated (usually the Ethernet hardware address) with a value representing the number of 100-nanosecond intervals since October 15, 1582

Placeholders are not case-sensitive. If you want to include a literal placeholder in a glossary item, escape the first # with a backslash, as in \#DATE#.

Selection and Insertion Placeholders

In BBEdit 8, you can use multiple #SELSTART#/#SELEND# pairs together with any number of #INSERTION# placeholders.

Example:

Suppose you have defined the following glossary item which contains an insertion placeholder:

```
typedef struct #SELECT#

{

#INSERTION#

} #SELECT#, *#SELECT#Ptr, **#SELECT#Handle;
```

If the selected text in your editing window is "MyStruct" and you insert this glossary item, BBEdit will insert the following in the editing window:

```
typedef struct MyStruct

{

|

} MyStruct, * MyStruct Ptr, ** MyStruct Handle;
```

(where the vertical bar marks the point at which the blinking insertion point will be placed).

Example:

Suppose you have defined the following glossary item which contains multiple pairs of selection placeholders:

```
MyFancyFunction(#selstart#arg1#selend#,
#selstart#arg2#selend#, #selstart#arg2#selend#);
```

When you insert this glossary item, BBEdit will place the following text in the editing window:

```
MyFancyFunction(arg1, <#arg2#>, <#arg3#>);
```

and the string "arg1" will be selected. You can then use the Go To Next/Previous Placeholder commands from the Search menu to hop to the other arguments and enter the desired values.

Temporary Placeholder Formats

In previous versions of BBEdit, the Glossary would generate temporary placeholders of the form "#•#" for items containing multiple instances of #INSERTION#. However, since the bullet character cannot be represented in all character encodings, BBEdit now generates temporary placeholders.

You should not rely on the format of these temporary placeholders; use the supported placeholders (#INSERTION#, #SELSTART#, #SELEND#) instead. If you have any existing glossary items which directly employed the old temporary placeholder format, you will need to modify them to use the supported placeholders.

Time Formats

The #GMTIME XXX# and #LOCALTIME XXX# placeholders offer you the option to insert the specified time value with flexible formatting.

In order to use these placeholders, you must substitute a time format using the same expansion options offered by the 'strftime' routine (see 'man strftime' for further details).

Examples:

`#LOCALTIME %r %z on %A#` produces: `06:50:13 PM -0400 on Monday`

`#GMTIME %r %z#` produces: `10:50:13 PM +0000`

Using Scripts with the Glossary

The placeholder #script filename# provides a powerful means to insert variable or conditional content by allowing access to any compiled AppleScript, or Unix shell script, from within a glossary item.

The script itself can either be located in the same folder as the glossary item that invokes it (in which case you need only specify its name, such as "MyDateScript") or you can supply a full pathname to a script on any mounted volume (such as "Hard Drive:My Project:Scripts:MyDateScript"). An instance of a placeholder referencing the latter would be

```
#script Hard Drive:My Project:Scripts:MyDateScript#
```

The script must return a text string (or a value that can be coerced to a string). This result string can itself contain additional glossary placeholders, which will be interpreted before the item is inserted in the current document.

WARNING Note that this makes it possible for one script to invoke another. Take care not to create a script execution loop, which would hang your system!

CHAPTER

13 Scripting BBEdit

BBEdit offers access to nearly all of its features and commands via AppleScript. This chapter provides a brief overview of AppleScript, discusses BBEdit's scripting model, and explains how you can use scripts within BBEdit.

An excellent way to learn how to script BBEdit is to look at the scripts others have written for it, or to turn on recording in your script editor while you perform actions in BBEdit. A number of example scripts are included in the standard distribution package. The BBEdit-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
```

IMPORTANT Regardless of whether you are new to scripting BBEdit or are familiar with scripting previous versions, we strongly recommend that you carefully review the sections "BBEdit 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 BBEdit. 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*. BBEdit 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 "BBEdit" to count text documents
```

If you are sending several commands to the same application, it is more convenient to write it this way:

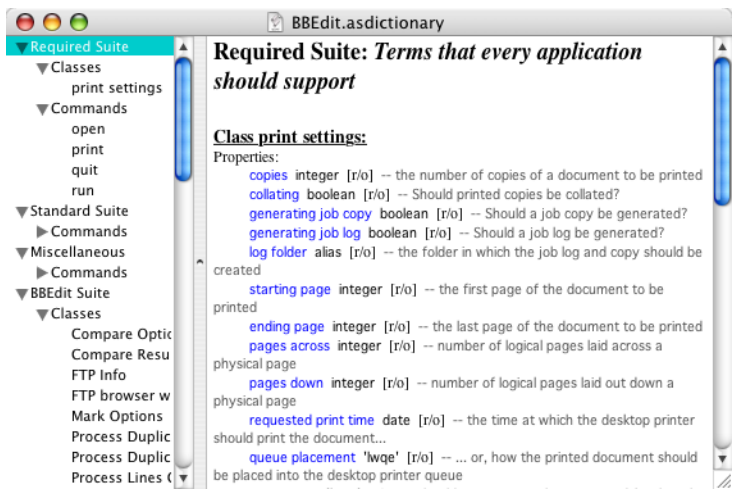
```
tell application "BBEdit"
    count text documents
    repeat with x from 1 to the result
        save text document x
    end repeat
end tell
```

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 BBEdit'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 BBEdit, for instance, classes are things like files, windows, the clipboard, browsers, and so on.

Suites

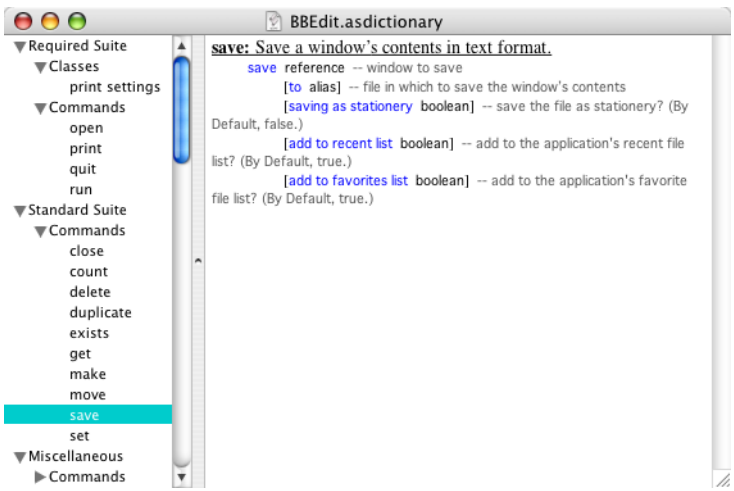
The first thing you will notice is that the events and classes are divided into *suites*. A suite is just a collection of related events and classes. Apple, for instance, has decreed that all applications should support particular events, which together are called the Required Suite. Another Apple-defined suite is the Standard Suite: if an application supports certain functions which Apple considers to be common, it 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, BBEdit has a Miscellaneous suite, a BBEdit Suite, a Text suite, and an HTML Scripting 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 BBEdit 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 BBEdit 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.)

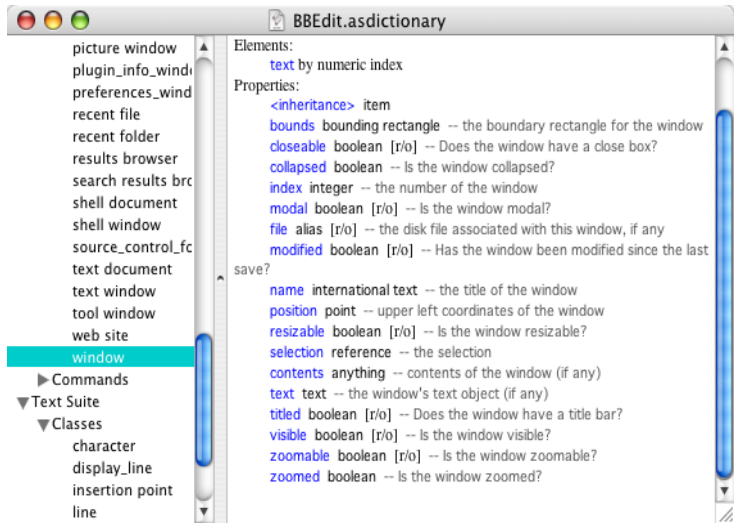
Anything in square brackets is optional. Most of the rest of the *save* event is optional, in fact. The basic event just saves the frontmost window to the same file from which it was opened. However, you can also optionally include the word *to* followed by a file reference. (You specify a file simply 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 BBEdit 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 BBEdit'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 BBEdit 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.

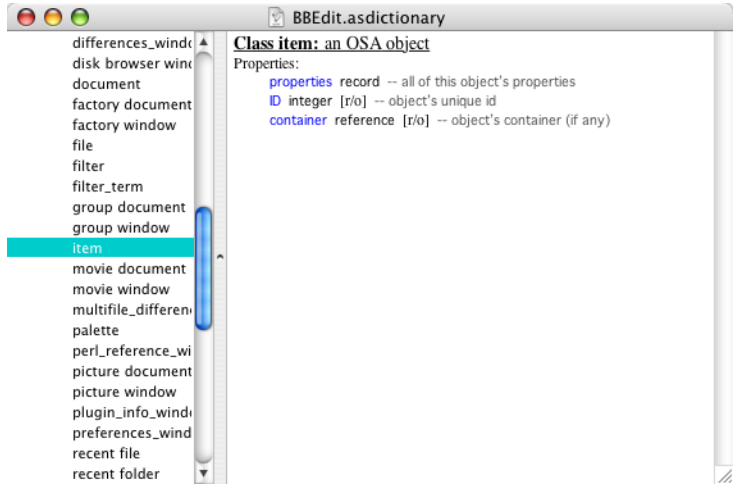
After a tag line that tells you about the class ("an open window") comes the plural form. AppleScript lets you refer to windows either singly or as a group, so it needs to know what the plural of every term is. For example, try this little script:

```
tell application "BBEdit" to count windows
```

The result of this script is the total number of window objects currently displayed by BBEdit.

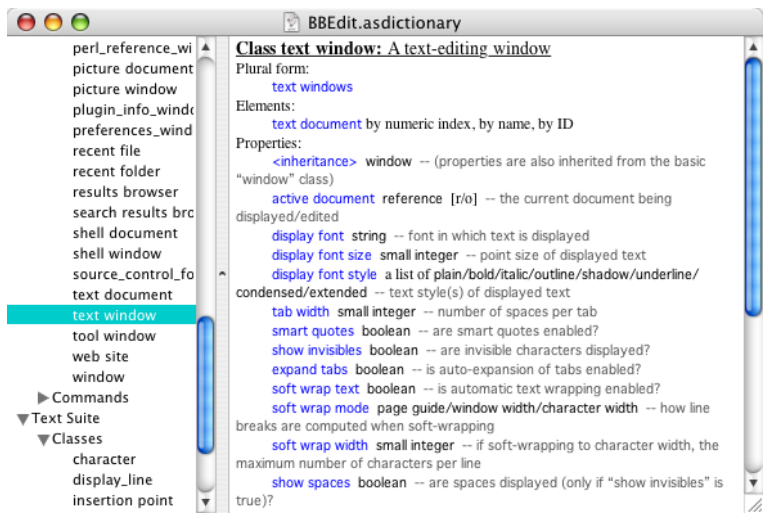
After the plural form comes a list of properties. Some objects do not have properties—for example, a string—but many applications do. An object's properties are merely a collection of data that describes that particular object. For example, as you look down the list of window properties, you will see 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 BBEdit is part of a hierarchy with the *item* class at the top, so every object in BBEdit "inherits" the *properties* property. This catch-all property can be handy for making exact duplicates of objects, among other uses.

You may realize that BBEdit has several kinds of windows; you can see their classes listed in the dictionary: clipboard window, differences window, disk browser window, file group 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 BBEdit, 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 "BBEdit" 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 "BBEdit" 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 BBEdit) 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. (BBEdit 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 BBEdit'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.

BBEdit-Talk, BBEdit-Scripting

`http://www.barebones.com/support/lists.html`
The discussion lists for BBEdit often cover BBEdit-specific scripting topics and are a good place to ask questions about BBEdit'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 BBEdit

BBEdit has been scriptable for years, and we have continually worked to refine its level of scripting support. In addition to providing extensive script access to its commands and data, BBEdit is both attachable *and* recordable.

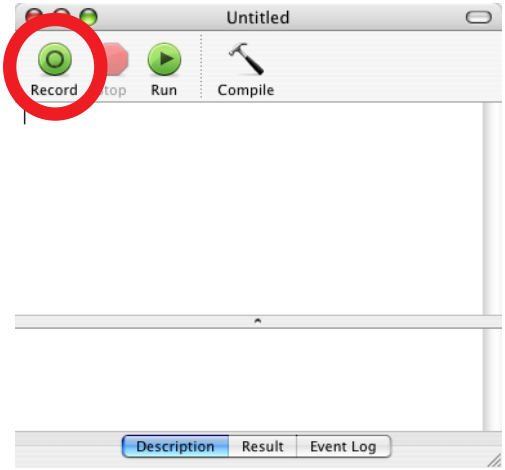
This section describes how you can create and employ AppleScripts within BBEdit via recording and BBEdit's various scripting facilities, while the following section covers BBEdit's scripting commands and other issues related to preparing scripts for use.

Recording Scripts in BBEdit

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 BBEdit 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 BBEdit and perform your task. Remember that the Script Editor is recording *everything* you do in every recordable application you are running, not just BBEdit. 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 BBEdit Scripts folder so that it shows up in BBEdit'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 BBEdit Scripts folder. Now try selecting it from the script menu in BBEdit.

The Scripts Menu

The Scripts menu (left) in BBEdit's menu bar contains several commands. It also lists all AppleScripts present in the Scripts folder within BBEdit'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, BBEdit launches it.

Open Scripting Dictionary
Choose this item to switch to your preferred AppleScript editor and open BBEdit's scripting dictionary for viewing. If the script editor is not running, BBEdit launches it.

Open Scripts Folder
Choose this item to open the Scripts folder which is located within BBEdit's application support folder. (See "Scripts" on page 14.)

Start Recording
Select this item to record all available actions that you perform within BBEdit (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, BBEdit 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 BBEdit open the script for editing in your preferred script editor, or hold down the Shift key when choosing a script item to have BBEdit reveal the script file in the Finder. If you choose a folder node rather than a script item, BBEdit 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

BBEdit lets you attach scripts to menu items. By this, we mean that you can write scripts that BBEdit automatically calls before or after performing a menu command. For example, if you want BBEdit'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 BBEdit 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 BBEdit menu items must be stored in the Menu Scripts folder of BBEdit'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 BBEdit's menus have icons rather than names. BBEdit uses the following names for its icon menus: "#!" [the 'Shebang' menu], "CVS", "Compiler", and "Scripts". Furthermore, the New With Stationery submenu is named "Stationery" for purposes of attachability.

When you choose a menu item with an attached script, BBEdit 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, BBEdit executes the script's run handler. The MenuSelect handler can return a boolean value to indicate whether BBEdit 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, BBEdit 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
```

BBEdit's Scripting Model

This section provides a high-level overview of BBEdit's scripting model that will, where appropriate, contrast the current scripting framework against older versions of BBEdit, and suggest how you can modify your existing scripts for compatibility.

IMPORTANT Because BBEdit'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 BBEdit's scripting dictionary, as well as for preparing and testing scripts.

```
http://www.latenightsw.com/
```

Script Compatibility

BBEdit's scripting model has changed significantly from that used in versions prior to 6.0. Consequently, scripts prepared for such versions may need to be revised in order to work properly. Further, with the addition of support for multiple documents per window in BBEdit 8, you may need to revise other existing scripts.

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 BBEdit.

Applying Commands toText

Since BBEdit supports opening multiple documents within a single text window, all scripting commands which operate on text must specifically target the text contents of a window, or a document within that window, rather than the window itself.

For example, you may use:

```
count lines of text of window 1
```

or:

```
count lines of active document of window 1
```

but not:

```
count lines of window 1
```

Documents vs. Windows

In old versions of BBEdit, 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 BBEdit:

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; the HTML tools palette, scripts list, and so on.
- *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.
- *group window*: provides a way to reference windows corresponding to open file groups. A group window does not present any properties beyond the basic *window* class, but provides a way to differentiate file group windows from other types of window.
- *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 BBEdit.

- *group document*: refers to a document corresponding to an open file group. A file group document does not present any properties beyond the basic *document* class, but provides a way to differentiate file group documents from other types of document.
- *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.

"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 BBEdit.

Getting and Setting Properties

One significant improvement in BBEdit'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 "BBEdit", 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 BBEdit to perform a whole series of search or search and replace operations.

Consider the scripting command below:

```
tell application "BBEdit"

find "BBEdit(.+)$" 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 "BBEdit"
    find "BBEdit" searching in document of text window 1
end tell
```

and

```
tell application "BBEdit"
    tell document of text window 1
        find "BBEdit"
    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 BBEdit, 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*, BBEdit 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 "BBEdit",
found text:"BBEdit"}
```

Scripting Single Replaces

To do a single find and replace via AppleScript, you can write:

```
tell application "BBEdit"

set result to (find "BBEdit" 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 "BBEdit"

set result to find "BBEdit(.+)$" 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 BBEdit.

Scripting Multi-File Searches

In BBEdit, 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

BBEdit has multiple clipboards. These are fully accessible via the scripting interface. Due to operating system constraints, most clipboard operations require BBEdit 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:"ProFont", display font size:9, style:{plain}}

```
clipboard 1 as text
```

- Returns "Files:WebSite:"

```
clipboard 1 as reference
```

- Returns clipboard 1 of application "BBEdit"

```
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.

Arranging Documents and Windows

BBEdit 8.1 includes several scripting enhancements which provide you greater control over window and document handling.

Opening Documents

The “open” command now supports additional options, which allow you to override your window handling preferences on a case by case basis:

```
open aFileList opening in <value>
```

As in previous releases, <value> may be a reference to an existing text window. However, you may instead specify “front_window”, “new_window”, or “separate_windows”, which have the following effect:

- front_window: All files in aFileList are opened in the frontmost text window. (If there is no text window open, BBEdit will create a new one.)
- new_window: All files in aFileList are opened into a new text window.
- separate_windows: Each file in aFileList is opened into its own text window.

Moving Documents

The "move"command can be used to move text documents between text windows. For example:

```
tell application "TextWrangler"
    if (count of text windows) > 0 then
        select text window 1
        repeat while (count of text windows) > 1
            set ct to count documents of text window 2
            repeat with i from 1 to ct
                move document 1 of text window 2 to text window 1
            end repeat
        end repeat
    else
        beep
    end if
end tell
```

Referencing Documents

Previously, documents were indexed inside of multi-document windows by their display order in the documents drawer. This meant that "document 1" of the application might not be the active document, which in turn required scripts to make special provisions to deal with the presence of multiple documents in a single window.

In order to handle this, BBEdit 8 introduced the "active document" property, which you could always use to specify the currently active document of a given text window. For example:

```
active document of text window 1 of application "BBEdit"
```

Although BBEdit still supports the "active document" property, this is no longer necessary. Instead, if a text window is frontmost:

```
document 1 of application "BBEdit"
```

```
document 1 of text window 1 of application "BBEdit"
```

```
active document of text window 1 of application "BBEdit"
```

now all refer to the same document. The side effect of this change is that if you wish to access documents within a text window by index, that index is:

a) not related to the visual ordering of documents in the documents drawer, and,

b) documents' indexes may change over time

This situation is effectively no different than handling documents which are contained in individual text windows, i.e. the index will change over time when you select different windows. If your script needs to keep a permanent references to a particular document, you should refer to that document by its id rather than its index.

Common AppleScript Pitfalls

Here are some things to watch out for when scripting BBEdit 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 BBEdit's powerful grep searching capability. BBEdit *also* uses the backslash as an escape character. If you want to search for an actual backslash in a document, you have to tell BBEdit 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 BBEdit. To pass one backslash to BBEdit from AppleScript, you must write two in AppleScript.

So to tell BBEdit 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 BBEdit. And BBEdit 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 BBEdit 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 BBEdit 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

14 Working with Development Tools

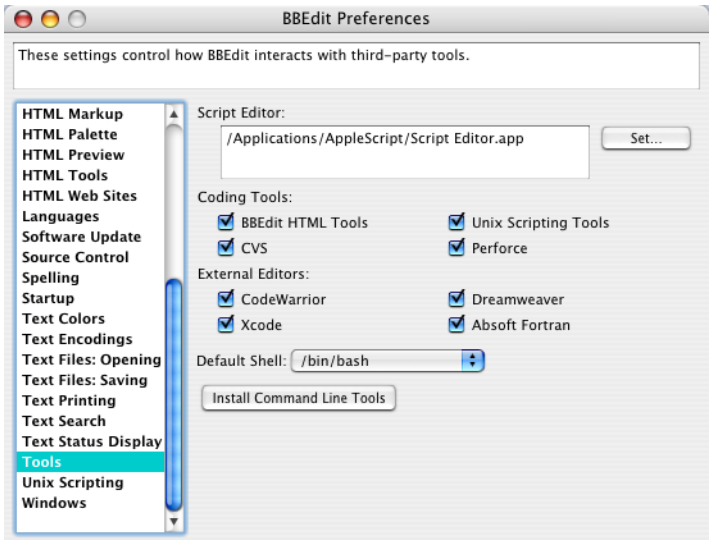
This chapter describes how to set up BBEdit to work with development environments. BBEdit offers a arsenal of capabilities in support of development tasks, beginning with syntax coloring and function browsing support for numerous languages, as well as support for Exuberant Ctags. You can use BBEdit as an external editor with Apple's Xcode, or Metrowerks' CodeWarrior. BBEdit also offers 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, and with the CVS and Perforce source control systems. Additionally, you can invoke BBEdit from the command line via optional tools, or employ shell worksheet windows to store and execute frequently used commands.

In this chapter

Configuring BBEdit for Development Environments

Before you can use BBEdit with a development environment, you need to let BBEdit know which development environments you plan to use.

Open the Preferences window (by choosing Preferences from the BBEdit menu). Select Tools from the list along the left side of the dialog. Click the check boxes to select the development tools you plan to use. (Note that some options will only be available if you have the appropriate programs installed. More information can be found in Chapter 10.)



If you made any changes in the default Tools options, click the Save button. Now, you must quit and relaunch BBEdit in order for these changes to take effect.

Syntax Coloring

Although it is not essential, you may want to turn on syntax coloring when you use BBEdit with a development environment. When syntax coloring is on, BBEdit displays keywords and other language elements in color. You can turn on syntax coloring by setting the Syntax Coloring option in the Editor Defaults preference panel.

Exuberant Ctags

In addition to its native function browsing capability, BBEdit now supports the use of Exuberant Ctags for navigating source code files. Complete information about Exuberant Ctags is available on the project web site.

```
http://ctags.sourceforge.net/
```

Source for the implementation used by BBEdit is available within the application package, or from our web site.

```
http://www.barebones.com/support/develop/ctags.shtml
```

Using ctags

If a tags file is found in the same directory as the front document, or in a parent of the front document's directory, you can employ the ctag information by selecting a word, and either choosing Find Definition from the Search menu, or by Control-clicking and accessing the Definitions submenu of the contextual menu.

If you choose Find Definition, BBEdit will use ctags information (if available) to find definitions of the selected word, and open a sheet from which you can choose the desired definitions to view. Select a definition to open it, or use the Show All button to open a search results browser showing all of the available definitions.

If you use the contextual menu, the Definitions submenu will contain a list of the available definitions. Select a definition to open it, or choose Show All to open a search results browser showing all of the available definitions.

Tag File Generation and Updating

Because individual work flows and setups vary widely, BBEdit does not attempt to generate or update ctags information; it only uses existing tags files. To generate your tags, we recommend you employ the following command:

```
ctags --excmd=number --tag-relative=no --fields=+a+m+n+S -R
`pwd`
```

By way of further explanation:

- BBEdit cannot process the search macros that a "standard" tags file has, so you must use `--excmd=number` to output line based tag data.
- BBEdit cannot locate files referenced by relative path, so you must use `--tag-relative=no` to output tag files with full paths.
- You can generate tags files for whatever fields you wish. BBEdit requires the line number (`--fields=+n`) and the signature (`--fields=+S`) to build the definitions menu. For our own source code, we also include class members (`+a`) and function implementations (`+m`) in addition to the required fields.
- BBEdit looks for tags files starting in the same folder as the current document, and crawls upwards from there. (Note that the /tmp portion of the Python script is included to allow you to merge tags files from different source trees.)
- If the ctags tool is invoked without a path, it appears to ignore the --tag-relative directive, so you should add the current directory's path as indicated (by adding `` `pwd` `` to the command).

In order to keep your tags up to date, we recommend incorporating a suitable script into your build system, or employing some other mechanism such as a cron task.

Note Refer to the Xcode documentation for information on creating build phases that run shell scripts. Similar solutions should be possible for CodeWarrior, and other IDEs.

For example, we use the below Python script to update the tags for our source code whenever a build is started in Xcode. Since ctags is efficient, this script only takes a few seconds to run, and will update the tags even if one or more files does not compile.

```
#!/usr/bin/python
import os
import sys

TAGS_TEMP_FILE = '/tmp/tags'

bbeditProjectDir = os.environ['SRCROOT']

bbeditTagsFile = os.path.join(bbeditProjectDir, 'tags')

sharedLibsDir = os.path.dirname(bbeditProjectDir)
sharedLibsDir = os.path.join(sharedLibsDir, "Shared Libs")

sharedLibsTagsFile = os.path.join(sharedLibsDir, 'tags')

ctagsExecutiblePath =
os.path.join(os.environ['TARGET_BUILD_DIR'], "ctags")

baseArgs = '--excmd=number --tag-relative=no --
fields=+a+m+n+S -f /tmp/tags -R'
appendArg = '--append'

os.chdir('/')

# create the BBEdit tags in '/tmp'
buildBBEditTagsCommand = ''''%s' %s '%s' ''' %
(ctagsExecutiblePath, baseArgs, bbeditProjectDir)
output = os.popen(buildBBEditTagsCommand).read()
print output

# append the Shared Libs tags to the same file
appendSharedLibsTagsCommand = ''''%s' %s %s '%s' ''' %
(ctagsExecutiblePath, appendArg, baseArgs, sharedLibsDir)
output = os.popen(appendSharedLibsTagsCommand).read()
print output

# move it where it goes
os.rename(TAGS_TEMP_FILE, bbeditTagsFile)

# create the Shared Libs tags in '/tmp'
buildSharedLibsTagsCommand = ''''%s' %s '%s' ''' %
(ctagsExecutiblePath, baseArgs, sharedLibsDir)
output = os.popen(buildSharedLibsTagsCommand).read()
print output

# move it where it goes
os.rename(TAGS_TEMP_FILE, sharedLibsTagsFile)
```

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. BBEdit uses the information in the Languages section of the Preferences window to determine whether a file is a source or header file.

Using Development Environments

Once you have set up BBEdit for the development environments you plan to use, the development tools communicate with BBEdit to report errors and changes in files. You use the commands in the appropriate menu (the Compiler menu or the Shebang menu) to send instructions to the development environment.

To switch to a development environment (or to launch it if it is not running), choose it from the appropriate menu. When you choose the Compile command (or any other command that causes files in the project to be recompiled), the development environment temporarily becomes the active application. If there are any compilation errors, BBEdit creates a Compile Errors browser that you can use to examine the errors.

BBEdit and the Unix Command Line

This section describes BBEdit's facilities for interacting with the Unix command line: shell worksheets for issuing commands *to* the Unix shell and the "bbedit" and "bbdiff" command line tools for invoking BBEdit *from* the command line.

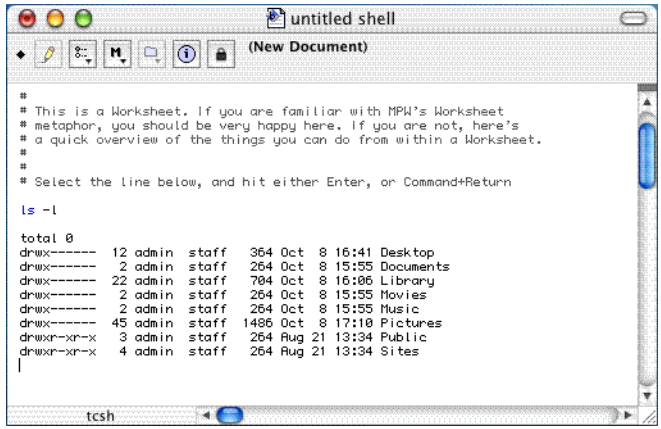
Shell Worksheets

BBEdit allows you to store and execute Unix command lines by means of a "shell worksheet." The New submenu of the File menu contains two commands for opening a shell worksheet. The first, "Shell Worksheet (*path*)", opens a worksheet for the default Unix shell that you have designated in the Tools preference panel (where *path* is the pathname of that shell). The second, simply called "Shell Worksheet", displays a dialog box for choosing the desired shell. Either command will open a worksheet window for typing commands to the specified shell.

Shell worksheets are stored in a private document format which is not text-based. This format has changed for BBEdit 8 in order to store auxiliary data in the worksheet file's data fork, thus ensuring that worksheets can safely be stored in version control systems that are not resource fork aware.

Using Worksheets

You can type, delete, and edit text in a worksheet window just as in an ordinary BBEdit document window. To invoke a Unix command, type the command, then press the Enter key or Command-Return, or click in the status area at the bottom-left of the worksheet window. (Keep in mind that Enter and Return are different keys; pressing Return by itself inserts a carriage return instead of executing a command.) You can execute more than one command at a time by selecting multiple lines and pressing Enter or Command-Return. The output will appear in the worksheet window below the line or lines containing the commands executed. Unlike a terminal, this does not have to be at the end of the document: you can type commands anywhere in the worksheet window, or place the insertion point back on a previously executed command to run it again.



Note If the selection range is non-empty, only the exact text selected will be executed; if there is just an insertion point, the entire line containing it will be executed (even if it is not at the end of the line).

The status area at the bottom-left of the worksheet window shows the name of the Unix process currently executing (or the name of the shell itself when no process is running). This can be useful for seeing what is going on when a process hangs or takes a long time to complete. You can kill the currently running process by typing Control-C or Command-Period in the worksheet window. Also, clicking in the status area sends the currently selected text (or the line containing the insertion point) as a command to the Unix shell.

Keep in mind that shell worksheets are not terminal windows. If you have ever used MPW, you will probably feel right at home using shell worksheets. If you are only familiar with terminal emulators, however, you will find that shell worksheets work quite differently. Command line editing gestures do not work, nor will any Unix commands that expect to be dealing with terminals. (For example, try running “emacs” in a shell worksheet.)

When you drag files or folders into a worksheet window, the behavior is different than when dragging these items into an ordinary document window. An unmodified drag of a single file or folder will insert the POSIX-style path of that item at the drop location. Additionally, rather than selecting the inserted text as in a normal editing window, the insertion point will be left at the end of the current line, so you can easily continue entering additional information or execute the line as a command.

Dragging multiple files and folders will now produce a set of paths for those items, with spaces for separators rather than carriage returns. This makes it easier to add arguments to a line for immediate execution as part of a command.

If you hold down the Command key while dragging, it will cause the file’s contents to be inserted (or a folder listing, if the item you are dragging is a folder).

The default working directory for new worksheet windows is the user’s home directory. This directory is also used as the search directory for any Open Selection or Open File by Name operations executed from within the worksheet. New shell windows are colored using the “Unix Shell Script” language.

New shell worksheets initially run in normal user mode as the currently logged-in user. This is indicated by the locked padlock icon on the rightmost button at the top of the worksheet window. You can switch to superuser mode by clicking the padlock button; you will be prompted to enter your administrator password. This is equivalent to executing the command

```
sudo -s
```

in a terminal window; the worksheet's status area will indicate this with a "sudo:" prefix to the current state, and the padlock icon will change to unlocked. Clicking the button again "locks" the padlock and returns to normal user mode.

WARNING If you are not familiar with Unix command line tools, we strongly urge you to obtain and read an introductory guide to using a Unix shell. Command line tools can be very useful, but if used incorrectly, they can render files, or even your entire system, unusable.

Default Worksheet Stationery

When creating a new worksheet window, BBEdit will look for a worksheet stationery file named "Default Worksheet Stationery". This file is located in the Stationery folder of BBEdit's application support folder. (See Chapter 2 for more information regarding BBEdit's application support folder.) If the default worksheet stationery exists, you will see the contents of this file in every new worksheet window you create.

BBEdit ships with a default worksheet stationery file that provides a small tutorial on using worksheet windows. When you grow tired of seeing this tutorial in every new worksheet, you can either remove the "Default Worksheet Stationery" file from the Stationery folder, or replace it with one of your own.

The "bbedit" Command Line Tool

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

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

```
bbedit 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 BBEdit open the file, or an FTP/SFTP browser to the folder.

To launch BBEdit without opening a file (or activate it, if it is already running), type

```
bbedit -l
```

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

```
ls -la | bbedit
```

If you just type

```
bbedit
```

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

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

```
bbedit [ -bchlpsuvVw --resume ] [ -e <encoding_name> ]
  [ -t <string> ] [ +<n> ] [ file (or) <S/FTP URL> ... ]
```

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

The "bbdiff" Command Line Tool

You can use the "bbdiff" command line tool to apply BBEdit's Find Differences command to a pair of files or folders specified on the Unix command line. The first time you run BBEdit after installation, it offers to install the "bbdiff" tool for you. If you choose not to do so, you can use the "Install Command Line Tools" button in the Tools preference panel to install the tool at a later time.

To invoke the Find Differences command from the command line, type

```
bbdiff file1 file2
```

or

```
bbdiff 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 "bbdiff" tool is

```
bbdiff [ --<options> ] [ FILE1 FILE2 | FOLDER1 FOLDER2 ]
```

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

Perl, Python, and Shell Scripting

BBEdit provides robust integration with Perl, Python, and Unix shell scripting languages.

Using Unix Scripts

BBEdit 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.

BBEdit'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 BBEdit 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. BBEdit 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, BBEdit will always send scripts to Perl, Python, and other script interpreters with Unix line endings.

Configuring Perl

BBEdit 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.

BBEdit 8 also supports the use of Affrus for debugging Perl scripts. Affrus is an integrated Perl development environment from Late Night Software.

`http://www.latenightsw.com/`

You can configure BBEdit to use Affrus in preference to the command line Perl debugger by setting the appropriate option in the Unix Scripting preferences panel (see page 197). If you chose to do this, BBEdit will pass complete environment parameters and control to Affrus, but will not attempt to retrieve output.

Configuring Python

BBEdit 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 BBEdit 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 BBEdit 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 BBEdit window. The output window is titled "Unix Script Output", and the file is created in the Unix Support folder in BBEdit'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 BBEdit's Logs folder (~/Library/Logs/BBEdit/).

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 BBEdit 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.

Show POD/Show Module Documentation

When the frontmost document is a Perl file and you invoke the Show POD command, BBEdit 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, the name of this command will change to Show Module Documentation, and if you invoke it, BBEdit will display the module documentation.

Filters and Scripts

Before you begin using Unix scripts with BBEdit, you should locate and familiarize yourself with the Unix Support folder, both of which reside in the BBEdit folder. (See Chapter 2 for details about the BBEdit 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 BBEdit'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 BBEdit. 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 BBEdit. Since BBEdit 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.

Working with CVS

BBEdit offers integrated support for CVS (the Concurrent Versions System source control package).

CVS Information

`http://www.cvshome.org/`

Quick Start for SSH-accessed Repositories

If you already have a CVS working copy from a repository that you access via SSH, BBEdit should now work "out of the box". To get started, you need only create a configuration in the Source Control preference panel for your CVS working copy by choosing the project folder, checking the "Use SSH" checkbox, and supplying your password (which BBEdit will store in the system keychain).

Other Configurations

If you use any other means than the above to access a CVS server, you will need to configure your local CVS setup and repository access from the command line, and perform an initial checkout.

How this configuration must be done varies from user to user and from organization to organization. The below article on Apple's web site provides a general overview of using CVS; for more details, you should consult your local CVS guru as necessary.

Mac OS X: Version Control with CVS

`http://developer.apple.com/internet/macosx/cvsoverview.html`

Once you have configured CVS, you must create a profile for each of your local CVS repositories for use with BBEdit via the Source Control preferences panel (see page 185). After doing so, you can proceed to work with files and folders in your local repository using the commands on BBEdit's CVS menu.

Performing Commit Operations

IMPORTANT All CVS commit operations now open a CVSEDITOR window to permit entry of checkin comments, in place of the text fields previously presented in the command dialogs. This lifts the previous 255 character restriction on checkin messages, and provides a better text composition environment.

CVS Menu Commands:

BBEdit's CVS menu contains the following commands.

Update to Head

This command replaces a file with the latest revision of that file from the CVS repository. To update the file in the front-most window, choose Update File from the CVS menu. To update any file on disk, hold down the Shift key while choosing Update File from the CVS menu, and choose a file from the Open dialog that appears.

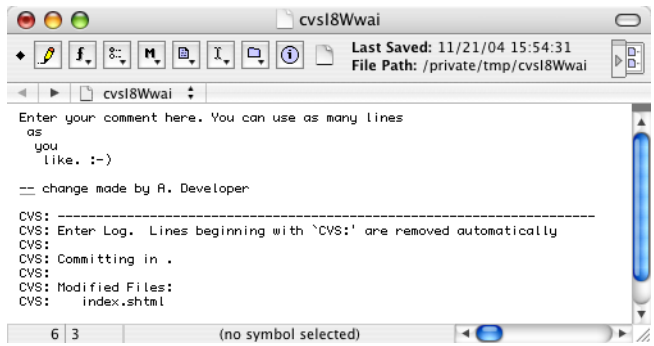
Update to Revision...

Choosing this command will bring up a dialog listing all available revisions for the current file. Select any of these revisions to perform an update against the current file.

Note To preserve the previous functionality of this command, if your file has local modifications, BBEdit will ask whether you want to merge those mods with the selected revision (as "cvs update" would normally do), or discard them and fetch the selected revision as-is (which BBEdit formerly did).

Commit...

Choosing this command will bring up a CVSEDITOR window (see below) to permit you to enter checkin comments, and then commits the current document into the repository. If the document has unsaved changes, it will be saved first. You cannot commit a file without having first added it to the repository using the Add command.



To enter a comment, type whatever you wish above the lines which are prefixed with "CVS:". Then, perform a Save (Command-S) and Close the window (Command-W); once you have done so, BBEdit will commit the document, and display the CVS log.

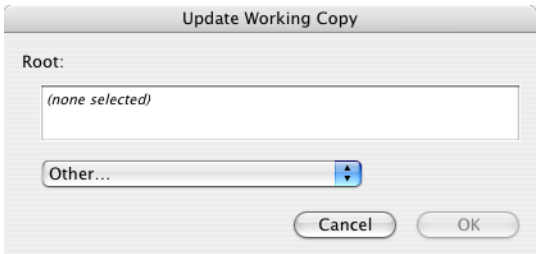
Note Although BBEdit lets you commit a file without a comment, it's considered good form to supply a comment that summarizes the changes you made to the file.

Commit Parent Folder...

When you choose this command, BBEdit will prompt you for a comment using the Commit Folder dialog, and then commit all of the files contained in the parent folder of the current document, just as if you had selected that folder and chosen the Commit Folder command. If you do not need to enter a comment, hold down the Shift key as you choose this command, and BBEdit will commit the folder's files without displaying a dialog box.

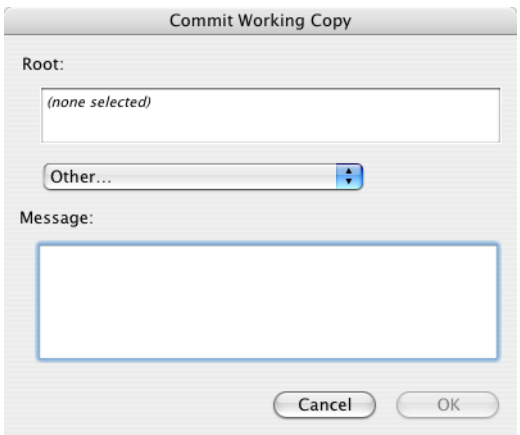
Update Working Copy...

When you choose this command, and select a folder in the Update Working Copy dialog, BBEdit will update all of the files contained in the folder to the latest available revisions. You can select any recently accessed folder from the pop-up menu, or choose the Other entry to bring up an Open dialog, allowing you to navigate to and choose any desired folder to update. You can also drag a folder from the Finder directly into the path box.



Commit Working Copy...

When you choose this command, BBEdit will bring up the Commit Folder dialog, which allows you to batch commit all the files within a specific folder that have uncommitted changes .

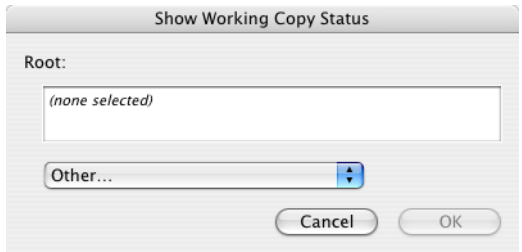


You can select any recently accessed folder from the pop-up menu, choose the Other entry to bring up an Open dialog, in which you can select any desired folder, or drag a folder from the Finder directly into the path box. If you enter a checkin comment in the Message field, that comment will be applied to each file that BBEdit commits.

Show Working Copy Status...

When you choose this command and select a folder in the Show Working Copy Status dialog, BBEdit will check whether any files contained within that folder have uncommitted changes. If there are any such files, BBEdit will display a results browser containing a list of them.

You can select any recently accessed folder from the pop-up menu, or choose the Other entry to bring up an Open dialog, allowing you to navigate to and choose any desired folder to commit. You can also drag a folder from the Finder directly into the path box.



Compare Revisions…

When working with a file that is in a CVS repository, you can use this command to access a dialog listing all available revisions of the file. When you choose a revision from the list, BBEdit will check that revision out into a temporary file, and automatically perform a Find Differences between it and the current file.

Compare Arbitrary Revisions…

When working with a file that is in a CVS repository, hold down the Shift key and Compare Revisions becomes Compare Arbitrary Revisions. You can use this command to bring up a dialog which contains two lists, each of which shows all available revisions of the file. Choose a revision from each list to have BBEdit check them out into temporary files, and automatically perform a Find Differences between them.

Compare with Base

When working with a file that is in a CVS repository, choosing this command will check out the base revision of this file as a temporary file, and compare that to the current document.

Compare with Head

When working with a file that is in a CVS repository, choosing this command will check out the head revision of this file into a temporary file, and compare that to the current document.

Show Annotation

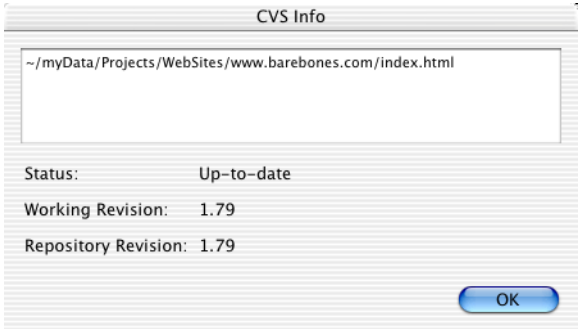
When working with a file that is in a CVS repository, choosing this command will perform a ‘cvs annotate’ on the file.

Show Revision History

When working with a file that is in a CVS repository, choosing this command will show the file’s revision history in a new display window.

Show CVS Status

When working with a file that is in a CVS repository, choosing this command will bring up the CVS Info dialog, which displays the local path to the file, and its current status and revision information.



Add

When you choose this command, BBEdit will add the current file to the appropriate CVS repository if it is not already present. Note that after using this command, you must also perform a Commit in order to store the file's contents to the repository.

Remove

When working with a file that is in a CVS repository, you can choose this command to remove the file from the repository. Note that after using this command, you must also perform a Commit Parent Folder (or a Commit Folder on the repository as a whole) before the file will actually be removed.

Go to Previous/Next Conflict

When you perform a CVS update on one or more files, if there are conflicts between your revisions and those in the repository, CVS will add markers in the affected file(s) at the appropriate points. If you update a single file and one or more conflicts result, BBEdit will automatically jump to the first conflict and you can use these commands to move to other instances (if any exist). If you update multiple files and conflicts result, you can choose either of these commands to move between instances of conflicts in a file. If there are no conflicts in the current file, BBEdit will beep when you select either command.

Export Project...

This command creates a copy of a project at the selected location. When you choose this command, BBEdit displays a dialog that lets you choose a CVS project and then displays a Save dialog that lets you choose the location in which to create a copy of it.

Open Log File

Opens the "CVS.log" file, or brings it to the front if it is already open. (This file is stored in the "Log files" folder within the "BBEdit Preferences" folder.)

Working with Perforce

BBEdit features integrated support for the Perforce source control system.

`http://www.perforce.com/`

In order to enable BBEdit's Perforce integration, you must have the "p4" command line tool installed, and must turn on the Perforce option in the Tools preference panel and quit and relaunch BBEdit. Next, you should configure your local Perforce repositories via the Source Control preferences panel (see page 185).

Once you have configured Perforce, you can work with files and folders in client spaces using the commands on BBEdit's P4 menu. You can apply many commands to the current document. You can also select one or more files from a results browser window, such as a search results browser, or the results browser for the Show Opened command, and apply commands.

If you have an open document which BBEdit does not recognize as being associated with any defined Source Control configuration, BBEdit will look for $P4CONFIG in its environment. If BBEdit finds such a value, and finds a config file in an appropriate directory, then it will honor that configuration.

Note In any such case, you must first open a document associated with the configuration, so that BBEdit knows where to set the working directory, before any commands on the Perforce menu, even non file-specific commands, will become enabled.

Perforce Menu Commands:

BBEdit's Perforce menu contains the following commands.

Edit

Performs a 'p4 edit' on the frontmost document, or the selected files.

Revert

Performs a 'p4 revert' on the frontmost document, or the selected files.

Revert & Sync To Head

Performs a 'p4 revert' followed by a 'p4 sync' on the frontmost document. or the selected files.

Sync To Head

Performs a 'p4 sync', passing the file path of the frontmost document, or of the selected files in a results browser, to the command.

Sync To Revision...

Displays a revision list for the frontmost document, and then syncs the file to the chosen revision

Sync All

Performs a 'p4 sync' with no additional arguments.

Submit...

Opens a change list editor window in BBEdit, and then submits the changes described within.

Show Opened
Displays a results browser showing all the files opened or added to the repository. You can select files in this browser, and apply many other commands to them.

Compare Revisions…
Displays a revision list for the frontmost document, and then compares the contents of that document to the revision you chose from the list.

Compare Arbitrary Revisions…
Displays two revision lists, and then compares the two revisions you choose to each other without changing the version you have checked out.

Compare with Base
Compares the contents of the frontmost document to the revision it is derived from.

Compare with Head
Compares the frontmost document to the head revision.

Show Annotation
Opens a window containing the results of a 'p4 annotate'.

Show Revision History
Shows the revision history for the current document. (Equivalent to a 'p4 filelog'.)

Add
Adds the frontmost document to the P4 repository. As with the command line tool, you must then perform a Submit to register this change with the server.

Delete
Deletes the frontmost document from the P4 repository. As with the command line tool, you must then perform a Submit to register this change with the server.

Revert & Delete
Performs a 'p4 revert' on the chosen document file prior to performing a deletion, allowing modified files to be discarded.

Go to Previous/Next Conflict
Searches the current document for the previous or next marker which indicates a sync conflict.

Open Log File
Opens the log file which contains the output from BBEdit's Perforce commands (~/Library/Logs/BBEdit/Perforce.log).

Working with Subversion

BBEdit now features integrated support for the Subversion source control system.

`http://subversion.tigris.org/`

Configuring Subversion

In order to enable BBEdit's Subversion integration, you must have Subversion 1.1 or later installed and available in the PATH for GUI applications (as controlled by ~/.MacOSX/environment.plist). You must also have a Subversion working copy which has been checked out via the command line.

If your repository access method is:

`file://`

> No additional steps are necessary, but for convenience, you might want to set up a Subversion configuration in the Source Control preference panel. (See the descriptions of the Show Working Copy Status, Commit Working Copy, and Update Working Copy commands which follow.)

`http://`
`https://`
`svn://`

> If access to the repository requires authentication, you will need to either:
>
> a) Let Subversion cache your credentials. At the present time, this will happen automatically unless you specifically prohibit it by passing "--no-auth-cache" on the command line, or setting that as the default behavior in your Subversion configuration files.
>
> b) Create a Subversion source control configuration in the Source Control preference panel, supply a username and password, and choose the location of your working copy. (The password will be stored in your system Keychain.)

Note If you employ option b), the username and password will be passed to the svn tool on the command line, and this information may be visible to other users of the machine. If you allow other users to log into your machine, this represents a security problem, and you should allow Subversion to cache credentials as outlined in (a).

`svn+ssh://`

> Create a Subversion source control configuration in the Source Control preferences, supply a username and password, and choose the location of your working copy.

Note If you have already configured an ssh-agent to work with GUI processes, it should also work here.

Command Line Integration

You can use the "bbdiff" command line tool as an external diff tool for Subversion. You can do so either directly, e.g.:

```
svn diff --diff-cmd bbdiff --extensions "--resume --wait" -rPREV
<FILE>
```

or by specifying the tool as the default "diff-cmd" in your Subversion config (~/.subversion/config) as appropriate.

Subversion Commands:

You can use most Subversion commands on either the active document, or the selection in a results browser or disk browser. Exceptions are as follows.

These commands are always available:

- Update Working Copy...
- Commit Working Copy...
- Show Working Copy Status...
- Open Log File

These commands require either an open document, or that you have selected a single item in a results browser or disk browser:

- Compare Revisions (all variants)
- View Annotation
- View Revision History
- Go to Previous Conflict
- Go to Next Conflict

Revert

Discards local changes to the active document's file, reverting it back to BASE. Has no effect if there are no local modifications to the file.

Update to Head

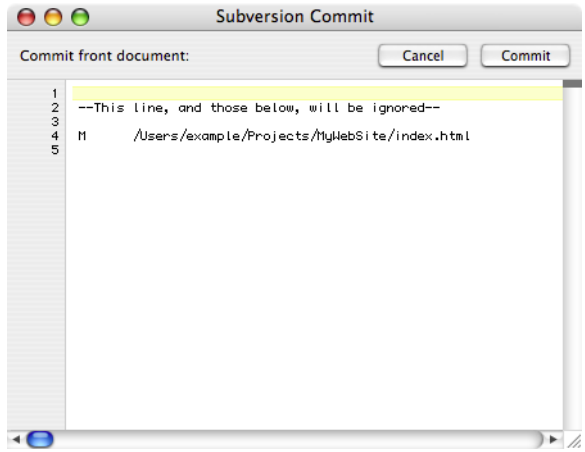
Updates the active document's file to the HEAD revision, merging local changes with the head revision, and marking conflicts as necessary. The first conflict in the file, if present, will be selected after the command successfully runs.

Revert & Update to Head

Discard local modifications before updating to HEAD.

Commit...

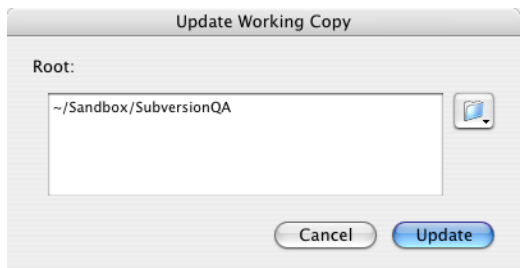
Commits the local file to the repository. BBEdit will display a Subversion commit window into which you can enter your commit message.



As shown above, this window will be prefilled with form text below the insertion point, consisting of a special separator line and a listing of the files which are about to be committed and their status. This separator line, and all lines below it, will be removed and will not appear as part of the commit message.

Update Working Copy...

Updates all the files contained in a working copy (folder tree) to the HEAD revision. BBEdit will display a dialog in which you can select the desired working copy, by either dragging and dropping a folder to the folder well, or by using the pop-up menu on the right which allows you to select any defined Subversion config, or any recently-accessed folder which is under Subversion's control.



Commit Working Copy...

Commits all changed files in a working copy to the Subversion repository. Uses a working copy selection dialog like that for the Update Working Copy command (above).

Show Working Copy Status...

Shows pending changes/status of the selected working copy. If you choose to show updates, BBEdit will query the server and also display files for which there are newer revisions on the server. Uses a working copy selection dialog like that for the Update Working Copy command (above).

Compare Revisions...

Displays a dialog which allows you to select any previous/future revision of the current file in the repository and perform a Find Differences between that revision and the local revision.

Compare Arbitrary Revisions...

Displays a dialog which allows you to choose any two revisions of the current file from the repository and perform a Find Differences on those revisions.

Compare with Base/Compare with Previous/Compare with Head

These are convenience commands which allow you to compare the local revision with BASE, PREV or HEAD revisions respectively.

Resolved

Marks the current file as resolved if it was previously marked as in conflict with updates received from the repository.

Show Annotation

Displays the content of the file line by line with author and revision information.

Show Revision History...

Brings up a sheet on which you can display options, and then displays the revision history of the local file. You can hold down the Option key when selecting this command to use the last settings.

Add

Schedules the current file, or all files selected in a browser, for addition to the repository. (You must perform a Commit to add the file(s) to the repository.) This command will not be available if the selected files are already part of the repository.

Delete

Schedules the current file, or all files selected in a browser, for deletion from the repository. (The local file(s) will be deleted immediately, but you must perform a Commit to permanently delete them from the repository.)

Revert & Delete

Reverts local changes to the current file, or all files selected in a browser, then schedules the file(s) for deletion from the repository. (The local file(s) will be deleted immediately, but you must perform a Commit to permanently delete them from the repository.)

Go to Previous /Next Conflict

Selects the previous or next conflict in the file, if any.

Open Log File

Opens BBEdit's Subversion log file (~/Library/Logs/BBEdit/Subversion.log).

Working with Metrowerks CodeWarrior

The following sections describe the range of commands which you can send to Metrowerks' CodeWarrior. For complete information on setting up and using BBEdit as an external editor with the CodeWarrior IDE, please refer to the appropriate section of your CodeWarrior user manual.

Note Due to the continual change in CodeWarrior project format, BBEdit does not currently offer support for browsing or performing Find Differences on such files.

Using the CodeWarrior Menu

This section describes each of the commands in the CodeWarrior menu (left).

CodeWarrior IDE

Choose this item to switch to CodeWarrior. If CodeWarrior is not running, BBEdit launches it.

Compile

This command tells CodeWarrior to compile the source file in the active window. The active window must belong to a project that is open in the development environment.

Check Syntax

This command tells the target development environment to check the syntax of the source file in the active window. The active window must belong to a project that is open in the development environment.

Show in Debugger

This command makes the selected text visible in the source-level debugger if the debugger is running.

Set Breakpoint

Use this command to make the selected text visible in the source-level debugger and to set a breakpoint. If you hold down the option key when you open the Compiler menu, this command becomes Set Breakpoint & Go, which sets the breakpoints and begins execution of the program.

If CodeWarrior's source-level debugger is not active, this command is not available.

Add

If the active editing window has been saved on disk, and it is not part of an open project, this command adds the file to the project.

Add & Compile

If the active editing window has been saved on disk, and it is not part of an open project, this command adds the file to the project and compiles it. This command is the same as using the Add command followed by the Compile command.

Precompile

Tells CodeWarrior to precompile the source file in the active window. BBEdit displays a standard file dialog that lets you specify the name of the precompiled header.

The active window must belong to a project that is open in CodeWarrior.

Preprocess

Tells CodeWarrior to preprocess the source file in the active window and to place the results in a new untitled editing window. The active window must belong to a project that is open in CodeWarrior.

Bring Up To Date

Tells CodeWarrior to check the project for modified files and to recompile and reload any files that have changed.

Make

Tells CodeWarrior to issue a make command to recompile and link all modified files in the open project.

Debug

Tells CodeWarrior to run the open project in the CodeWarrior debugger.

Run

Tells CodeWarrior to run the open project *without* using the source debugger. If choosing this command causes CodeWarrior to recompile files, compiler errors may not be reported to BBEdit. If that happens, use Make instead, and then select Run.

Working with Xcode

Xcode is Apple's native development environment for Mac OS X. If you have Xcode installed, you can configure it to use BBEdit as an external editor.

To enable this integration, first, turn on the "Xcode" option in the Tools panel of BBEdit's Preferences window. Then, switch to Xcode, open its Preferences window and select the File Types section, and expand the "file" hierarchy to reach the "sourcecode" node. From the popup-menu at the right, choose "External Editor", and select BBEdit from the list of available applications.

Once you have completed these steps, Xcode will automatically ask BBEdit to open any files you select, and BBEdit will notify Xcode whenever you save a file that it asked BBEdit to open.

CHAPTER

15 Language Modules & Plug-Ins

Language modules and plug-ins are code modules that you can install to enhance BBEdit's features. Language modules provide support for syntax coloring, and optionally, function browsing, for programming languages beyond those built in, while plug-ins provide specialized text processing and related features. Many developers have written language modules or plug-ins for BBEdit, which are available from various web sites (including our own).

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 BBEdit application support folder. If no such folder exists, you may create one.

To install a plug-in, drag and drop it directly onto the BBEdit application icon in the Finder. BBEdit 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 BBEdit 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 BBEdit application support folder available, one will be created. (See "BBEdit's Application Support Folders" on page 12.)

After installing a new language module or plug-in, you will need to quit and relaunch BBEdit 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 BBEdit application support folder, then quit and relaunch BBEdit.

Using Language Modules

Language modules are add-on items which provide syntax coloring and function browsing for programming languages that BBEdit 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 BBEdit's plug-in module interface. (See "Developer Information" on page 307.) Codeless language modules are text documents prepared in a specific plist format. (See below.)

After you install a language module and relaunch BBEdit, 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 183).

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. BBEdit 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 BBEdit 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 D of this manual.

Language Module Compatibility

IMPORTANT You will **not** be able to use any third-party language modules which do not support Unicode text. If BBEdit 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, BBEdit 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 BBEdit 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 BBEdit 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 BBEdit 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, BBEdit 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 BBEdit 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 BBEdit 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, BBEdit 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 BBEdit plug-in:

1. **Choose the Tool from the Tools palette.**
2. **Click Set Key.**
3. **BBEdit opens the Set Key dialog.**
4. **Click Reset.**

   BBEdit removes the key assignment from the plug-in.

Supplied Plug-Ins

The text processing plug-ins supplied with previous versions of BBEdit have been superseded by equivalent commands on the Text menu. (See "Text Menu Commands" on page 81.)

IMPORTANT You should **not** copy any of the factory-supplied plug-ins from previous versions to use with BBEdit 8, and BBEdit will warn you if it encounters any such items. (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 BBEdit 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 BBEdit 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" 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

The developer information overview page is:

```
http://www.barebones.com/support/develop/index.shtml
```

APPENDIX

A

Command Reference

This appendix provides a quick reference for key assignments and a comprehensive list of the commands that are available from BBEdit's user interface.

In this appendix

Keyboard Shortcuts for Commands

Many of BBEdit's commands have pre-defined keyboard shortcuts. BBEdit also lets you reassign the shortcuts for any menu command, glossary entry, plug-in, or script 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.)

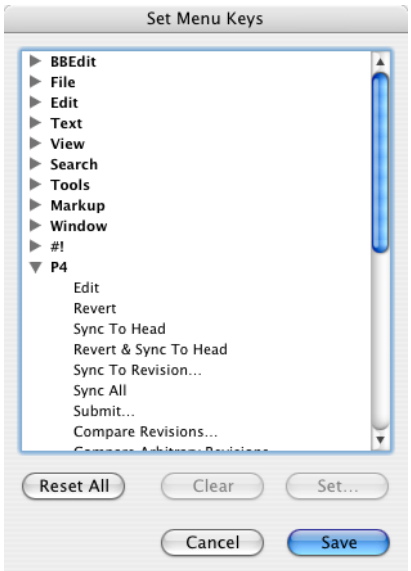
Many other BBEdit features can have keyboard shortcuts assigned as well. Here's how to set them:

Feature	Set Keys in...
Menu commands	Set Menu Keys on the BBEdit menu
Plug-ins	Plug-In Tools palette
Glossary items	Glossary palette
Scripts	Scripts palette
Unix filters and scripts	Unix Filters and Scripts palettes
Stationery	Stationery palette

To display any of BBEdit'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 BBEdit's menu commands, as well as items on the Text Options, Markers, and Line Breaks status bar pop-up menus, by choosing Set Menu Keys from the BBEdit 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. The Help, Home, End, Page Up and Page Down keys can be used 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.

If the Auto-Assign Shortcut Keys option in the Windows panel of the Preferences is turned on, the key combinations Command-0 through Command-9 will be reserved for file entries on the Window menu. There may also be other key combinations which the operating system preempts, such as Command-Tab.

Listing by Menu and Command Name

BBEdit Menu

About BBEdit	
Plug-In Info	
Preferences	Cmd-,
Set Menu Keys…	
Enter Serial Number…	
Services	*(submenu)*
Hide BBEdit	(*none or* Cmd-H)
Hide Others	
Show All	
Quit BBEdit	Cmd-Q

File

New	*(see next column)*
New With Stationery	*(submenu)*
Open…	Cmd-O
Open Hidden…	
Open from FTP/SFTP 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-Shift-W
Close All Windows	Cmd-Opt-W
Close Document	Cmd-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 FTPS/FTP Server…	Cmd-Shift-S
Save a Copy to FTP Server…	Cmd-Opt-Shift-S
Revert	
Hex Dump File	
Hex Dump Front Document	
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)	
HTML Document…	Cmd-Ctl-N
Text Window	Cmd-Shift-N
File Group	
Disk Browser	Cmd-Opt-N
FTP/SFTP Browser	
Shell Worksheet <default>	
Shell Worksheet…	
Text Factory	

Edit

Undo	Cmd-Z
Redo	Cmd-Shift-Z
Clear Undo History	Cmd-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	Cmd-Ctl-V
Clear	
Select All	Cmd-A
Select None	Cmd-Shift-A
Select Line	Cmd-L
Select Paragraph	Cmd-Opt-L
Insert	*(see below)*
Auto-Complete Glossary…	
Insert Glossary Entry…	
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

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 *(see next column)*

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-Shift-Opt-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
Go to Previous Error	Cmd-Opt–up arrow
Go to Next Error	Cmd-Opt–dn arrow
Go to Previous Placeholder	Ctl-Shift-'
Go to Next Placeholder	Ctl-'
Go to Function Start	
Go to Function End	
Go to Previous Function	
Go to Next Function	
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 Definition	Cmd-hyphen
Find in Reference	Cmd-Shift-hyphen

Tools *(Installed plug-ins)* (none by default)

Markup *(see following pages)*

View

Hide Status Bar	
Hide Navigation Bar	
Hide Navigation Bar	
Previous Document	Cmd-Opt-[
Next Document	Cmd-Opt-]
Open in Separate Window	
Get Info	
Reveal in Finder	
Open in Super Get Info	

Window

ASCII Table	
Minimize Window	
Minimize All Windows	(Opt)
Bring All to Front	
Palettes	*(see next page)*
Workspace	*(submenu)*
Arrange…	
Get Info	
Reveal in Finder	
Cycle Through Windows	Cmd-‘
Exchange With Next	
Synchro Scrolling	
(Open windows)	Cmd-1 *to* Cmd-0
Glossary	
Plug-In Tools	
Scripts	
Stationery	

Window -> Palettes

Windows
HTML Markup Tools
- Block
- CSS
- Entities
- Font Style
- Forms
- Inline
- Phrase
- Table
- Utilities
- Web Safe Colors

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…	
Show POD/Show Module Documentation	
Unix Filters	*(submenu)*
Unix Scripts	*(submenu)*

CVS

Update to Head	
Update to Head…	(Shift)
Update to Revision…	
Commit…	
Commit File…	(Shift)
Commit Parent Folder…	
Update Working Copy…	
Commit Working Copy…	
Show Working Copy Status…	(Shift)
Compare Revisions…	
Compare Arbitrary Revisions	
Compare with Base	
Compare with Head	
Show Annotation	
Show Annotation…	(Shift)
Show Revision History	
Show Revision History…	(Shift
Show CVS Status	
Add	
Remove	
Go to Previous Conflict	
Go to Next Conflict	
Export Project…	
Open Log File	

P4

Edit
Revert
Revert & Sync to Head
Sync to Revision…
Sync All
Submit…
Show Opened
Compare Revisions…
Compare Arbitrary Revisions…
Compare with Base…
Compare with Head
Show Annotation
Show Revision History…
Add
Revert & Delete
Open Log File
Go to Previous Conflict
Go to Next Conflict

Subversion

Revert
Update to Head
Revert & Update to Head (Shift)
Commit…
Update Working Copy…
Commit Working Copy…
Show Working Copy Status…
Compare Revisions…
Compare Arbitrary Revisions…
Compare with Base
Compare with Previous
Compare with Head
Resolved
Show Annotation
Show Revision History…
Add
Delete
Revert & Delete (Shift)
Go to Previous Conflict
Go to Next Conflict

Compiler

CodeWarrior IDE
Absoft Tools
Compile Cmd-K
Check Syntax Cmd-Y
Debug Cmd-I
Set Breakpoint Cmd-Shift-I
Set Breakpoint & Go Cmd-Opt-Shift-I
Add
Add & Compile
Precompile…
Preprocess
Bring Up To Date Cmd-U
Make
Debug Cmd-R
Run Cmd-Opt-R

Scripts

Open Script Editor
Open Scripting Dictionary
Open Scripts Folder
Start/Stop Recording
(Installed scripts)

Text Factories

Open Text Factories Folder
(Installed text factories)

Markup

Tag Maker…	Cmd-M
Edit Tag…	Cmd-Opt-M
Close Current Tag	
Balance Tags	Cmd-Opt-B
Document Type…	
Character Set…	
CSS	*(see below)*
Body Properties…	
Head Elements	*(see below)*
Block Elements	*(see next column)*
Lists	*(see next column)*
Tables	*(see page 316)*
Forms	*(see page 316)*
Inline	*(see page 316)*
Phrase Elements	*(see page 316)*
Font Style Elements	*(see page 316)*
Frames	*(see page 316)*
Check	*(see page 316)*
Update	*(see page 317)*
Includes	*(see page 317)*
Utilities	*(see page 317)*
Misc	*(see page 317)*
Preview in BBEdit	
Preview in <Default Browser>	Cmd-Ctl-P
Preview With	*(see page 317)*

Markup --> CSS

@import…
@media…
Box…
  Padding…
  Border…
  Margins…
Layout…
  Position…
  Size & Constraints…
  Clipping…
  Effects…
Background…
Font…
List Style…
Text…
Format

Markup --> Head Elements

Base…
Link…
Meta…
Script…
Noscript
Style…
Format

Markup --> Block Elements

Paragraph…	
Paragraph	(Opt)
Div…	
Horizontal Rule…	
Horizontal Rule	(Opt)
Heading…	Cmd-Ctl-H

H1
H2
H3
H4
H5
H6
Address
Blockquote…
Center
Deleted Text…
Inserted Text…
Noscript
Preformatted

Markup --> Lists

List…	Cmd-Ctl-L

Unordered
Ordered
Definition
Menu
Directory
List Item
List Items

Markup --> Tables

Table…	Cmd-Ctl-T
Row…	
Row	(Opt)
TD…	
TD	(Opt)
TH…	
TH	(Opt)
Caption	
Colgroup…	
Col…	
THead…	
TFoot…	
TBody…	
Convert to Table…	

Markup --> Forms

Form…
Button…
Field Set
  Legend…
Input…
Label…
Select…
  Option Group…
    Option…
Text Area…

Markup --> Inline

Anchor…	Cmd-Ctl-A
Image…	Cmd-Ctl-I
Applet…	
Object…	
Param…	
Script…	
Map…	
Area…	
Convert to Client Side Map…	
Break…	Cmd-Ctl-B
Break	
Font…	Cmd-Ctl-F
Base Font…	
Bidirectional Override…	
Quotation	
Span…	
Subscript	
Superscript	

Markup --> Phrase Elements

Abbreviation
Acronym
Citation
Computer Code
Deleted Text…
Defined Term
Emphasis
Inserted Text…
Input Text (Kbd)
Sample Output
Strong Emphasis
Variable

Markup --> Font Style Elements

Big
Small
Bold
Italic
Strike-Through
Teletype Text
Underline

Markup --> Frames

Frame Set…
Frame…
No Frames

Markup --> Check

Document Syntax	Cmd-Ctl-Y
Document Links	Cmd-Ctl-K
Folder Syntax…	
Folder Links…	
Site Syntax…	
Site Syntax	(Opt)
Site Links…	
Site Links	(Opt)

Markup --> Update

Document	Cmd-Ctl-U
Folder…	
Site…	
Site	(Opt)
Document Images…	
Document Images	(Opt)
Folder Images…	
Site Images…	
Site Images	(Opt)

Markup --> Includes

Persistent Include…
Include…
Placeholders…

Markup --> Utilities

Format…	Cmd-Opt-Shift-F
Format	
Optimize	
Translate…	Cmd-Opt-T
Remove Comments	
Remove Markup	
Comment	
Uncomment	
Normalize Tag Case	
Make Tags Upper Case	
Make Tags Lower Case	

Markup --> Misc

Dreamweaver	
Document Size	
Index Document	
Index Folder…	
Index Site…	Cmd-Ctl-X
GoLive Cleaner	
HomePage Cleaner	
PageMill Cleaner	

Markup --> Preview With

Preview as Text
in All Running Browsers
(Installed browsers)

Listing by Default Key Equivalent

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 File by Name
Cmd-E	Search: Enter Search String
Cmd-F	Search: Find…
Cmd-G	Search: Find Again
Cmd-H	Search: Find Selection *or* BBEdit: Hide BBEdit
Cmd-I	CodeWarrior: Debug
Cmd-J	Search: Go to Line…
Cmd-K	CodeWarrior: Compile
Cmd-L	Edit: Select Line
Cmd-M	Markup: Tag Maker…
Cmd-N	File: New: Text Document
Cmd-O	File: Open…
Cmd-P	File: Print…
Cmd-Q	BBEdit: Quit BBEdit
Cmd-R	CodeWarrior: Debug
Cmd-S	File: Save
Cmd-T	Search: Replace & Find Again
Cmd-U	CodeWarrior: Bring Up To Date
Cmd-V	Edit: Paste
Cmd-W	File: Close Document
Cmd-X	Edit: Cut
Cmd-Y	CodeWarrior: Check Syntax
Cmd-Z	Edit: Undo

Key	Command
Cmd-’	Text: Rewrap Quoted Text…
Cmd--	Search: Find Definition
Cmd-;	Text: Find Next Misspelled Word
Cmd-[	Text: Shift Left
Cmd-]	Text: Shift Right
Cmd-`	Window: Cycle Through Windows
Cmd-/	Zoom Window
Cmd-=	Search: Replace
Cmd-\	Text: Hard Wrap…
Cmd–left arrow	Search: Apply to New
Cmd–right arrow	Search: Apply to Old
Cmd-Ctl-A	Markup: Inline: Anchor…
Cmd-Ctl-B	Markup: Inline: Break…
Cmd-Ctl-F	Markup: Inline: Font…
Cmd-Ctl-H	Markup: Block Elements: Heading…
Cmd-Ctl-I	Markup: Inline: Image…
Cmd-Ctl-K	Markup: Check: Document Links
Cmd-Ctl-L	Markup: Lists: List…
Cmd-Ctl-N	File: New: HTML Document…
Cmd-Ctl-P	Markup: Preview
Cmd-Ctl-T	Markup: Tables: Table…
Cmd-Ctl-U	Markup: Update: Document
Cmd-Ctl-X	Markup: Misc: Index Site…
Cmd-Ctl-Y	Markup: Check: Document Syntax
Cmd-Ctl-Z	Edit: Clear Undo History
Cmd-Ctl-/	Window: Send To Back
Cmd-Opt-B	Markup: Check: Balance Tags
Cmd-Opt-D	File: Reveal Selection
Cmd-Opt-E	Search: Enter Replace String
Cmd-Opt-F	Search: Quick Search

Key	Command
Cmd-Opt-J	Search: Go to Line
Cmd-Opt-L	Edit: Select Paragraph
Cmd-Opt-M	Markup: Edit Tag…
Cmd-Opt-N	File: New: Disk Browser
Cmd-Opt-P	File: Print All
Cmd-Opt-R	CodeWarrior: Run
Cmd-Opt-S	File: Save All
Cmd-Opt-T	Markup: Utilities: Translate…
Cmd-Opt-W	File: Close All Windows
Cmd-Opt-'	Text: Rewrap Quoted Text
Cmd-Opt-;	Edit: Find All Misspelled Words
Cmd-Opt-`	Text: Exchange Words
Cmd-Opt-[	View: Previous Document
Cmd-Opt-]	View: Next Document
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-E	Search: Enter Replace Pattern
Cmd-Opt-Shift-F	Markup: Utilities: Format…
Cmd-Opt-Shift-I	CodeWarrior: Set Breakpoint & Go
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
Cmd-Shift-E	Search: Enter Search Pattern
Cmd-Shift-G	Search: Find Again (reverse)
Cmd-Shift-H	Search: Find Selection (reverse)
Cmd-Shift-I	CodeWarrior: Set Breakpoint

Key	Command
Cmd-Shift-J	Search: Go to Center Line
Cmd-Shift-N	File: New: Text Window
Cmd-Shift-O	File: Open from FTP/SFTP 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--	Search: Find in Reference
Cmd-Shift-;	Text: Show Spelling Panel
Cmd-Shift-[	Text: Shift Left One Space
Cmd-Shift-]	Text: Shift Right One Space
Ctl-‘	Search: Go to Next Placeholder
Ctl-Shift-‘	Search: Go to Previous Placeholder
Ctl-Tab	Switch to Header/Source File
Ctl-[	Edit: Previous Clipboard
Ctl-]	Edit: Next Clipboard

APPENDIX

B

Editing Shortcuts

In BBEdit you can perform many editing functions (including word selection or deletion) directly from the keyboard. Chapter 4 contains complete details on BBEdit'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	–none–

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, BBEdit 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 up one line)
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 BBEdit, 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.

APPENDIX

C

Placeholders and Include Files

This appendix lists the placeholder tokens used by BBEdit templates and include files, and describes the use and capabilities of include files.

In this appendix

Placeholders

Placeholders are processed under the following circumstances:

- When a new HTML document is created from a template, the placeholders in the template are replaced with their current values. (The new document receives the substituted text; the original template file is not modified.)
- When the Update Document command (part of the HTML Tools) is invoked, any placeholders in the documents being updated are replaced with their current values. (Since the placeholders are replaced, subsequent updates *do not* update the substituted text.) Although this command is part of the HTML Tools, it can be used in any document whenever you want to use placeholders.
- When a file is included in another file using the #bbinclude directive (or a related directive), any placeholders in the included file are replaced with their current values before the text is included. (The include file itself is not changed, only the included text is substituted.) All of the above methods of invoking placeholders can also invoke included files, which can have placeholders of their own.

Note The placeholders described in this chapter are only available for use with the HTML Tools' Update command. They cannot be used with BBEdit's Glossary command, nor can Glossary placeholders be used in include or template files.

BBEdit supports the following placeholders. Placeholders are not case-sensitive.

Placeholder	Replaced By…
#ABBREVDATE#	Abbreviated date—e.g., Sun, Aug 15, 2004
#BASE#	The BASE tag as entered using the New HTML Document command
#BASENAME#	The name of the file stripped of its rightmost period-delimited portion. For example, if the file is named “test.html”, the base name is “test”, while if the file is named “test.foo.html”, the base name is “test.foo”.
#BASE_URL#	The value of the BASE URL specified in an HTML document’s header (useful if you want to refer to the document’s location on the server)
#BODYTEXT#	When invoking a template using the New HTML Document command, and the Create New Window checkbox is not marked, the current contents of the frontmost window (if any) will replace this placeholder
#CHARSET#	The character set specified in the New HTML Document command
#COMPDATE#	Compact Date format—e.g., 15-Aug-04
#CREATIONDATE#	The creation date of the current file—e.g., 15-Aug-04
#CREATIONTIME#	The creation time of the current file, formatted according to your Format settings in the International panel of the System Preferences
#DIRPATH#	The path on the server as specified in the HTML Web Site panel of the Preferences window. Strips any leading slash from the path string
#DOCSIZE#	The size of the current document plus included images in bytes
#DOCTITLE#	The title of the current document as extracted from the <TITLE> tag
#DONT_UPDATE#	Marks a document so that the HTML Update tool will ignore it during processing
#FILE_EXTENSION#	The filename extension for the file (determined as the rightmost period-delimited portion of the filename, without the period). For example, whether the file is named “test.html” or “test.foo.html”, the filename extension is "html".
#FILENAME#	The file name of the current file
#GENERATOR#	Generator name used for “Give BBEdit Credit” in New HTML Document function (e.g., “BBEdit 8”)
#GMTIME XXX#	The current GMT time formatted according to the parameters XXX (see “Time Formats” below)
#LANGUAGE#	The language specified in the New HTML Document command, in format (space)lang=”en”

Placeholder	Replaced By…
#LINK#	The LINK tag as entered using the New HTML Document command
#LOCALPATH#	The full local path to the current file
#LOCALTIME XXX#	The current local time formatted according to the parameters XXX (see “Time Formats” below)
#LONGDATE#	Long Date format—e.g., Sunday, August 15, 2004
#MACHINE#	The machine name as specified in the Sharing section of the System Preferences.
#META#	Any META tag entered using the New HTML Document command
#MODIFIEDDATE#	Modification date of the current file—e.g., 15-Aug-04
#MODIFIEDTIME#	Modification time of the current file, in the format specified in the International section of the System Preferences
#MONTHDAYNUM#	Numeric value of the day of the month
#MONTHNUM#	Numeric value of the current month
#PATH#	Path to access your documents from the Web server, as specified in your HTML Web Site preferences
#PREFIX#	As #DIRPATH# but does not strip the leading slash of the path
#REAL_URL#	The real URL for the current document in its current location
#RELATIVE#	The relative path from the current file back up to the Local Server Root (inserts a path of the form “../../” to tell the browser to “back up” to the site’s root directory)
#ROOT#	Path to the Local Site Root, as specified in your HTML Web Site preferences
#ROOTPATH#	The file’s path relative to the Local Server Root specified in the HTML Web Site preferences to the current file
#SERVER#	URL of your Web server, as specified in your HTML Web Site preferences.
#SHORTDATE#	Short Date. Day, month, year—e.g., 08/15/04
#SHORTUSERNAME#	Returns the login (short) name instead of the full user name
#TIME#	Current time, according to your Format settings in the International panel of the System Preferences
#TITLE#	Title of the current document as entered using the New HTML Document command
#USERNAME#	The owner name (for the currently logged in user)

Placeholder	Replaced By…
#YEARNUM#	The current year—e.g., 2004

Time Formats

The #GMTIME XXX# and #LOCALTIME XXX# placeholders offer you the option to insert the specified time value with flexible formatting.

In order to use these placeholders, you must substitute a time format using the same expansion options offered by the 'strftime' routine (see 'man strftime' for further details).

Examples:

#LOCALTIME %r %z on %A#

produces:

06:50:13 PM -0400 on Monday

#GMTIME %r %z#

produces:

10:50:13 PM +0000

Using the #RELATIVE# Placeholder

When dealing with large web sites that have multiple content folders, it is often useful to specify relative rather than absolute paths for linking documents. The #RELATIVE# placeholder allows you to easily generate relative references in templates and include files by providing a virtual path that uses the ".." construction to "back up" the hierarchy to the root directory of the site.

To use this placeholder, write your links as if they were all relative to the top of your web site, including #RELATIVE# as the first "directory" in the path. For example, consider that you have the following file structure, where each page includes a file which references the separate GIF image.

```
My_Web_Site:
    Folder1:
        File1.html
    Folder2:
        File2.html
        File3.html
    Folder3:
        Folder4:
            Folder5:
                File4.html
    Graphics:
        Buttons:
            my_footer_button.gif
```

If you write a relative link as follows:

```
<img src="#relative#Graphics/Buttons/my_footer_button.gif"
alt="">
```

and then run the Update command, the following links will be generated.

In File1.html,

```
../Graphics/Buttons/my_footer_button.gif
```

In File2.html,

```
../Graphics/Buttons/my_footer_button.gif
```

In File3.html,

```
../Graphics/Buttons/my_footer_button.gif
```

In File4.html,

```
../../../Graphics/Buttons/my_footer_button.gif
```

Include Files

An include file, or just an "include," is a special form of placeholder whose substitution happens to be the contents of another file. If you have used C or certain other programming languages, you may already be familiar with the concept. Using includes, you can reuse standard bits of text content or HTML markup in several templates or glossary entries without having to revise all of those individual files whenever you revise the included text.

Include File Locations

BBEdit looks for include files first in the same directory as the document containing the directive, then in the same directory as the document into which the processed document is being inserted, and finally in the HTML Templates folder specified in your preferences.

Simple Includes

A simple include takes the following form:

```
#bbinclude "filename"
```

where `filename` is the full name of the file whose contents you wish to include. When such an include is present in a template or glossary entry, it is replaced with the contents of the specified file when the template is used to build a new document, or when the glossary entry is inserted. (The original template or glossary file is not changed.)

Imagine that you have ten different templates, each of which contains your name, address, phone number, email address, and a copyright statement with the current year in them. Rather than pasting this info into all ten templates, you can create a file named "address.html", put it in your Templates folder, and include this statement:

```
#bbinclude "address.html"
```

in each of the templates, at the appropriate point. Later, when the new year arrives, or you move, you only have to update one file, not all ten templates. (You could use the #YEARNUM# placeholder for the year and only need to update the include file when you move!)

Headers and footers are probably the most common uses for include files, but any template or glossary entry may use as many include statements as you wish. Included files themselves may also use #bbinclude directives, up to 16 levels deep.

Persistent Includes

Simple includes are appropriate for use situations where you want the inclusion to happen only once. Once the file has been included, however, it cannot be changed in any automated fashion. Since the #bbinclude directive is replaced by the included text, the Update tool cannot tell the included text is any different from any other text.

Includes become even more powerful, however, when you can update existing files to incorporate revised include text at a later date. For example, suppose you generate several dozen HTML documents using a template that uses an #bbinclude directive to insert a standard footer containing your email address. Later, you change your email address. After you change it in the footer document, only *new* HTML files you create from the template will include your new address. What you would really like to be able to do is update all the files you have *already* created to include the revised footer.

Since this capability is needed primarily in web site maintenance, BBEdit lets you embed the include directive in an HTML comment. An "end bbinclude" comment is also required. The included text is inserted *between* the two comment markers, but the comments themselves remain in place. The comments are not shown in the browser. This is known as a *persistent* include.

A persistent include looks like this:

```
<!-- #bbinclude "filename" -->
<!-- end bbinclude -->
```

The first time a persistent include is processed, it is handled much like a simple include. However, since the include directives remain in place, and because they mark the beginning and end of the inserted text, the Update tool can "rip out" the obsolete included text and replace it with the updated file. Using persistent includes and the Update Folder or Update Site commands, you can easily make these sorts of changes to entire sites in moments.

IMPORTANT Any changes you have made to the included text after its initial inclusion will be discarded when the persistent include is updated, even if you have not changed the include file.

Inline versus Block Includes

By default, BBEdit places included content in the document as a block, to ensure that it does not occupy the same line as the include directives. However, if you wish to override this behavior and have BBEdit place the included content inline, you may do so by adding the special option `#bbincludeoptions#="inline=true"` to the include directive.

```
<!-- #bbinclude "filename" #bbincludeoptions#="inline=true" -->
<!-- end bbinclude -->
```

Include Files with Variables

Include files can be extended even further through the use of variables, which provide a means of inserting arbitrary text when the included file is processed, so that not all instances of the included file are exactly the same. Variables are essentially placeholders that you make up yourself. Some possible uses are to insert names, taglines, alt strings for images, or file names (for files other than the current document) into documents.

Note A variable name consists of a string of alphanumeric characters, enclosed in number signs (the '#' character). Spaces are not allowed in variable names, but underscores may be used to represent word breaks.

Variables can be placed anywhere in an include file, just like placeholders. When you include that file in a document, you specify the variable names and values with it. Consider an include file named "footer.html", which contains the following

```
<HR>
<IMG SRC="#MY_GRAPHIC#" ALT="#MY_ALT_DESC#">
<H1>#MY_TITLE#</H1>
<BIG>This document copyright 1998-2004 by Sid Zookim.</BIG>
```

In your document, the Include reference would look like this:

```
…
<BODY>
…
<!-- #bbinclude "footer.html"
#MY_GRAPHIC#="test1.gif"
#MY_ALT_DESC#="a test image"
#MY_TITLE#="A Test Title"
-->
<!-- end bbinclude -->
…
</BODY>
…
```

Note that the values of placeholders are specified *inside* the HTML comment of a persistent include, using a #PLACEHOLDER#="Value" syntax. The quote marks around the value are mandatory; if you need to include a quote mark in the actual value, escape it with a backslash.

Including AppleScripts

BBEdit allows included files to be compiled AppleScript scripts. The script should contain an "on include" handler which is passed two parameters: a reference to the file from which the script is being called, and a record containing one field for each variable passed in the #bbinclude directive. Scripts can of course also retrieve information from BBEdit, other scriptable applications, or the system. The handler's return value is inserted into the file that included it.

Given the HTML document below:

```
<html>
<head>
    <title>Include Test</title>
    <meta name="generator" content="BBEdit 8">
</head>
<body>

<!-- #bbinclude "foo.script" #x#="3" #author#="JEK"-->

<!-- end bbinclude -->

</body>
</html>
```

The following script inserts three lines: the first containing the file's path, the second containing the parameter "x" passed to it in the #bbinclude directive, and the third containing the parameter "author."

```
on include(f, vars)
   set s to f as text
   set s to "File Path:  " & s & return & return as text
   set s to s & "x:  " & x of vars & return & return as text
   set s to s & "Name:  " & author of vars & return as text
   return s
end include
```

The resulting document might look like this:

```
<html>
<head>
    <title>Include Test</title>
    <meta name="generator" content="BBEdit 8.0">
</head>
<body>

<!-- #bbinclude "foo.script" #x#="3" #author#="JEK"-->
File Path:  Boot:Desktop Folder:incl_test.html

x:  3

Name:  JEK
<!-- end bbinclude -->

</body>
</html>
```

Including Perl or Python Scripts

BBEdit lets you include scripts written in Perl, Python, or any other Unix scripting language. The complete path name of the file being processed is passed to the script as its first argument, and any variables in the include statement are passed as additional arguments. All these can be retrieved via @ARGV in your Perl script.

Any text sent to STDOUT by the script will be taken as the value of the #bbinclude operation and inserted into the HTML file. If an error occurs while running the script, the STDERR output, if any, will be inserted into the file along with the STDOUT, and a single line indicating the error will be added to the error browser.

For example, enter this directive in your HTML file:

```
<!-- #bbinclude "foo.pl" #length#="2" #width#="3" -->
<!-- end bbinclude -->
```

Then use this source code for "foo.pl", and save it in the same folder as the HTML file, or in the "Templates and Includes" folder specified in the HTML Web Sites panel of your BBEdit preferences:

```
#!/usr/bin/perl -w
my $file = shift @ARGV;
my %args = @ARGV;
my $area = $args{"length"} * $args{"width"};
print "Filename: $file\n";
print "Area: $area\n";
```

When you run the Update command, BBEdit will place the file name in the script's variable $file and the "length" and "width" variables in the associative array (hash) %args.

After the update, the BBEdit file will look like this:

```
<!-- #bbinclude "foo.pl" #length#="2" #width#="3" -->
Filename: Mac HD:Desktop Folder:sample.html
Area: 6
<!-- end bbinclude -->
```

In addition, BBEdit will pass information about the current HTML Tools settings to the script in the following environment variables:

```
BBEditServerURL
BBEditServerPath
BBEditDefaultFileName
BBEditTemplateDirectory
BBEditRootDirectory
BBEditLowercaseTags
BBEditLowercaseAttributes
BBEditAlwaysQuoteAttributes
```

To access these in your Perl code, use the %ENV environment variable hash. For example, this line of Perl will print the Web server name specified in your BBEdit HTML Web Site preferences:

```
print $ENV{BBEditServerURL};
```

Here's an example Python include script.

```
#!/usr/local/bin/python
import os
import string
import sys

print "Hello Python World!"
print "==================="
print "File being updated: ", sys.argv[1]
print

userVariables = {}
for i in range(2, len(sys.argv), 2):
   userVariables[sys.argv[i]] = sys.argv[i+1];

print
print "Dumping the user variables passed to the script"
print "==============================================="
print
keys = userVariables.keys();
keys.sort()
for k in keys:
   print "%-30s %s" % (k, userVariables[k])

print
print "Dumping the environment variables set by BBEdit"
print "==============================================="
print
for k, v in os.environ.items():
   if (string.find(k, 'BBEdit') == 0):
      print "%-30s %s" % (k, os.environ[k])
```

Other Include Notes

IMPORTANT Some old versions of BBEdit supported the use of "#include" as an alternative to "#bbinclude". However, this syntax made it difficult to mix BBEdit includes and Microsoft Active Server Page (ASP) directives, so it is no longer supported. If you have existing documents which use this syntax, simply change "#include" and "end include" to "#bbinclude" and "end bbinclude" to use them with BBEdit 8.

APPENDIX

D 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`

In this appendix

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 BBEdit 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, BBEdit 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 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.	
BBLMIsSourceKind	**Boolean**
If this key is present and its value is 'true', files with this suffix are considered 'source' files.	
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.	
Language Features	**Dictionary**
This dictionary is simply a collection of key/value pairs that define the language elements that the module supports.	
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.

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.

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.

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.

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

A

B

C

D

E

F

G

H

I

J

K

L

M

N

O

P

Q

R

S

T

U

V

W

X

Y

Z