

Table of Contents

Table of Contents	1
Overview.....	1
SlickEdit 2008 (v13)	1
Version 13.0.0.....	2
Version 13.0.1.....	11
SlickEdit 2007 (v12)	19
Version 12.0.0.....	20
Version 12.0.1.....	24
Version 12.0.2.....	28
Version 12.0.3.....	33
SlickEdit v11	35
Version 11.0.0.....	35
Version 11.0.1.....	37
Version 11.0.2.....	37
SlickEdit v10.....	37
Version 10.0.0.....	37
Version 10.0.1.....	39
Version 10.0.2.....	40
Version 10.0.3.....	40
SlickEdit v9.....	40
Version 9.0.0.....	40
Version 9.0.1, 9.0.2, 9.0.3.....	42
Version 9.0.4.....	43

Overview

This document describes the features and enhancements made in the past several releases of SlickEdit. For more documentation, see the Help system that is installed with the product. For more information about SlickEdit and SlickEdit products, please visit our Web site at www.slickedit.com.

SlickEdit 2008 (v13)

SlickEdit 2008 (v13.0.0) was released in Spring, 2008.

SlickEdit 2008 introduces a new approach for our options dialogs. All of the options have been gathered together into an options hierarchy that is available from **Tools > Options**. The top level of the hierarchy is composed of categories for settings, like **Appearance**, **Keyboard**, **Editing**, and **Languages**. Each of these categories groups together related options. For example, the Appearance group contains settings for scroll bars, highlighting, fonts, colors, toolbars, etc. Some of options screens are the same as they were in the previous version. For example, the Fonts screen, which used to be on the main menu at **Tools > Options > Fonts**, is now located in the hierarchy at **Appearance > Fonts**. We have changed other options to aggregate them into property sheets. This gives us greater flexibility to group things into a logical category without running out of space on a given dialog.

The new Options dialog offers a search facility that will help you located options you are familiar with. In most cases, the option name has not been changed. So, if you're trying to locate **Click past end of line**, you can find it by entering "Click" in the search field. This will prune the options tree down to just those items that contain a match. By clicking on

SlickEdit® Version History

the matched categories you will quickly find that **Click past end of line** is located in **Editing > General**. Click the **Clear** button to restore the hierarchy to its full presentation.

The items that were formerly located in **Tools > Options > File Extension Setup** are now located in the options hierarchy under **Languages**. We've tried to make our settings language-centric rather than extension-centric as before. Each language has a mode name and a corresponding list of associated extensions. The Languages category contains a Language Manager, to add and delete languages, and a File Extension Manager to associate extensions with Languages. See "Language Manager" in the **Help > Index** for more information on managing languages and extensions.

The **Languages** category contains a list of languages broken down by type. For example, you will find C/C++ under **Application Languages**. Each language contains a set of options, like Indent, Word Wrap, Comments, etc. Most of these correspond to the tabs on the former Extension Options dialog. Here, again, the search capability will help you locate familiar options that may have moved.

Other important notes:

- SlickEdit now uses a new licensing and activation scheme based on FLEXnet™. See "licensing" in the **Help > Index** for more information.
- The **-sc** invocation option now creates a versioned subdirectory in the specified directory, so your old configuration will be preserved.
- The VSLICKCONFIG environment variable is no longer supported. You need to use SLICKEDITCONFIG instead. SlickEdit creates a versioned subdirectory under the specified directory, just as it does for the **-sc** option or the default configuration directory.
- The new version, SlickEdit 2008, will not include the C++ Refactoring feature. The Quick Refactorings, which use Context Tagging® to perform their changes, are being retained. We will continue to support this feature pursuant to the Maintenance and Support Terms and Conditions until the next major release.
- Support for the Windows NT platform is being discontinued in this version.
- Due to compatibility issues with .NET 2.0, the .NET debugger integration has been removed entirely from SlickEdit 2008. If you have a .NET SDK project that would normally support .NET debugging, **Debug > Start** will simply tell you that the feature is no longer available. The menu item **Debug > Attach to Running Process using .NET** has been removed.
- (Mac only) Until Apple includes the latest X11 in their software updates, the best place to go for the ongoing fixes being made to X for the Mac is <http://trac.macosforge.org/projects/xquartz>. Look for the section called "Latest Release" located near the top.

Version 13.0.0

New Features in this Version

- **Message List** - Message List is a feature that shows output messages from processes running in SlickEdit, such as build warnings and errors. Messages are displayed in the Message List tool window, docked in the bottom tab group of the editor by default. The tool window shows messages in tabular format, with columns for the message type (warning, error, etc.), source file and line number associated with the message, a description, the creator of the message (Build, Java Live Errors, etc.), and the date. Messages can be filtered and sorted. You can clear the tool window of all messages, or clear only messages with a certain creator or type. Messages are associated with a location in the code. When the cursor is on a line with an error or warning, the corresponding message in the

SlickEdit® Version History

Message List is highlighted. You can also navigate to the message in the source code from within the Message List by double-clicking on it (or right-click on a selected message and click **Go to code location**).

- **New Options Dialog** - One of the main advantages of SlickEdit is its vast configurability. To make options and settings more accessible, a new Options dialog is available. The Options dialog combines all of the individual options dialogs formerly used to configure SlickEdit, including dialogs previously on the **Tools > Options** menu. The options have been sorted into new categories which are presented as nodes in a tree. Click on the nodes to view individual options, which are displayed in embedded forms or in tabular format. The new Options dialog provides the following features:
 - Navigation buttons to jump to previously viewed options pages.
 - An incremental search function to find options that contain specific text.
 - A way to mark frequently used options pages as "favorites" for quicker access.
 - An Options History feature to see which options have been changed and when.

To access the Options dialog, from the main menu, click **Tools > Options**, or use the **config** command on the SlickEdit command line.

- **Enhancements to Installed Language Support** - Language-specific options are now accessed from the **Languages** node of the new Options dialog (**Tools > Options > Languages**). The options are categorized by language type and language. For example, to access C/C++ or Java options, expand the **Application Languages** node. Under each language, the options are further categorized by type (General, Advanced, Indent, etc.). The option categories are the same for each language while the particular settings are unique to each language. Some languages do not support all option categories or options, and some include additional options that are not available in other languages.

Installed language support has also been enhanced to add a layer of indirection between physical file extensions and language associations, providing more flexibility with extension mapping. Previously, file extensions could only be mapped to their native language, for example, the "m" extension could not mean anything other than Objective-C. Now you can remap extensions to a completely different language very easily, without hiding or removing support for the native language. Languages and file extensions are now managed through the **Language Manager** and **File Extension Manager** screens of the Options dialog (**Tools > Options > Languages**).

- **Adaptive Formatting** - Many development teams set standards for code formatting styles. These standards often vary from project to project or between languages. In this environment, you can lose valuable time in having to change configurations, set/unset options, or run beautifiers from file to file just so you can meet the team's requirements. Adaptive Formatting addresses these situations by scanning a file for the formatting styles in use, and automatically matching those settings for the current editing session. This provides seamless integration of new code with existing code, making it easier to read - not only for you, but for the next person who needs to edit the file.

Adaptive Formatting is enabled by default and recognizes indentation and tab style settings, parentheses padding, and begin/end style settings. It also recognizes case settings, such as keyword casing for case-insensitive languages, and tag, attribute, and value casing for HTML-based languages. To configure what settings are recognized or to disable the feature, use the language-specific Adaptive Formatting options page (**Tools > Options > Languages > [LanguageCategory] > [Language] > Adaptive Formatting**).

- **Quick Brace/Unbrace** - Quick Brace makes it easy to convert single line statements into a brace-enclosed blocks so you can add new lines without having to manually position the cursor and type extra keystrokes. Quick Brace attempts to honor your brace style and indent settings. To use Quick Brace, position the cursor where you would normally type the open brace, and type the open brace. SlickEdit automatically moves the child statement to the next line (if the statement was contained on one line), indents it according to your indent preferences, and inserts the closing brace according to your brace style preferences. Unbrace does the opposite of Quick Brace, removing the braces from a brace-enclosed block that contains a single line statement and moving the statement to the

SlickEdit® Version History

preceding line that contains the parent statement (unless it is just too long). To use Unbrace, simply delete the opening brace.

This feature set is on by default and can be enabled/disabled on a language-specific basis through the Formatting Options screen (**Tools > Options > Languages > [LanguageCategory] > [Language] > [Language] Formatting Options**).

- **Perl Regular Expressions** - SlickEdit now supports the Perl regular expression syntax and has enhanced the Brief, SlickEdit, and UNIX syntaxes to support various features found in the Perl syntax. New capabilities include:
 - **Word anchor matching** - Match a word boundary or anything except at a word boundary.
 - **Extended escape sequence** - Match all characters as literals until reaching a specified point.
 - **Case-sensitive matching** - Match the casing of text or ignore case.
 - **Character case modification on replace** - Convert matched characters to upper- or lowercase.
- **Makefile Import** - You can now create a project by importing a makefile. This feature is a time-saver when you have a project that was not created in SlickEdit that contains a makefile. Simply import the makefile, and SlickEdit automatically parses the targets, finds all referenced source files, and adds them to a new SlickEdit project. To import a makefile, from the main menu, click **Project > Open Other Workspace > Makefile**. When you import a makefile, SlickEdit creates a new workspace and adds the project to it.

The new project automatically imports all files that are referenced by the makefile. All of the make targets are also added and made available for execution from the main menu under **Build > Execute Makefile Target**. For makefiles that contain invocations of other makefiles, the other makefiles can be optionally added to the workspace as separate projects, or all their files added into one project.

- **Cursor on Symbol Shows All Uses in File** - SlickEdit can highlight all occurrences of the current symbol under the cursor. This makes it easy to see, at a glance, all uses of a symbol in a file. This option can be set on a language-specific basis. To enable it, from the main menu, click **Tools > Options > Languages**, expand your language category and click the language, then select **Context Tagging®**. On the options screen, select **Highlight matching symbols under cursor**.
- **Enhanced Auto-Generation of Doc Comments** - By default, SlickEdit automatically expands doc comment skeletons when you type the start characters. Now, using aliases, you can modify the template that gets expanded. To modify a doc comment template, from the main menu, click **Tools > Options**, expand **Languages**, your language category, and language, and select **Comments**. In the **Doc comments** section, select the type of skeleton you want to modify from the **For start characters** drop-down list, then click the **Edit expansion** button. The Doc Comment Editor dialog opens with the specified start characters already selected in the alias list, so you can start editing the comment template in the edit window.

In addition to Comment enhancements, SlickEdit has added several new escape sequences that you can use in your doc comment templates by using the **Insert Escape Sequence** list on the Doc Comment Editor. These include the ability to insert local function param names, types, and return types.

- **Full SCIM Support** - SlickEdit now provides full SCIM support for Linux users. First, set the VSLICK_XIM environment variable to **scim** and set the XMODIFIERS environment variable to **@scim (Macro > Set Macro Variable)**. Then, from the main menu, click **Tools > Options > Keyboard > Redefine Common Keys**, and set the **Use Ctrl+Space for input method editor** option to **True**. Now, you can press Ctrl+Space at any time to activate the SCIM input method editor.
- **Slick-C® Debugger** - The Slick-C Debugger helps you locate bugs and resolve issues in your Slick-C code. When the debugger is activated (**Macro > Start Slick-C® Debugger** or **slickc_debug_start** command), a debugger instance of SlickEdit opens and attaches to the running instance. Use the Debug menu items to perform debug operations.

SlickEdit® Version History

- **URLs Treated as Links** - SlickEdit now treats URLs in editor windows as hyperlinks, making them easy to identify and open from within your code. A string is interpreted as a URL if it begins with one of the following protocols (including trailing slashes):
 - http://
 - https://
 - ftp://
 - file://

URLs are underlined. You can navigate to a link by hovering over it with the mouse and using **Ctrl+Click** (or **Command+Click** on the Mac). To configure recognized URI schemes, from the main menu, click **Tools > Options**, then expand **Network & Internet Options** in the tree and select **URI Schemes**.

- **Enhanced URL Mapping** - SlickEdit lets you map URLs to different locations. You can now specify a default lookup directory that contains all of your DTDs and namespace schema files. Every mapping doesn't need to be explicitly configured. You can also create mappings for namespace URIs as well as DTD files. To configure URL Mapping, from the main menu, click **Tools > Options**, then expand **Network & Internet Options** in the tree and select **URL Mappings**.
- **Auto Symbol Translation** - Auto Symbol Translation automatically converts a character or sequence of characters to the appropriate entity reference, saving you from having to repeatedly guess at the correct entity or look up reference charts. This feature works automatically as you type, so you don't need to press a special key or key sequence to trigger the translation. For example, type `>>`, and SlickEdit automatically converts the `>>` sequence to **>**; - the entity reference for the right angle bracket (`>`). Typing `&&` translates to **&**; - the entity reference for the ampersand symbol (`&`).

Auto Symbol Translation is based on the Alias mechanism in SlickEdit - the character or character sequence that you type to trigger translation is actually a special kind of alias, henceforth called a "symbol alias". Regular aliases are expanded by pressing a key sequence or, in some cases, when you type a space. Symbol aliases are expanded as soon as you finish typing the characters in the alias. Because of this, you need to make sure a symbol alias does not begin with the another symbol alias. For example, you cannot have an alias named `"|"` and another one named `"|>"`.

SlickEdit comes with some predefined symbol aliases. You can view these, customize them, and create your own by using the Symbol Translation Editor dialog. The first time Auto Symbol Translation is triggered, a prompt appears that describes the feature and lets you open the Symbol Translation Editor.

- **New Clipboards Tool Window** - A new Clipboards tool window is available for previewing and managing your clipboards in SlickEdit. The tool window shows a list of your recently used clipboards (up to 50 by default), and has a Preview area that lets you see the entire contents of a clipboard including color-coding based on the source of the clipboard text. Display of the Clipboards tool window can be toggled on/off by clicking **View > Toolbars > Clipboards**, pressing **Ctrl+Shift+V**, or by using the **toggle_clipboards** command. To display the tool window on demand, use the **activate_clipboards** command.

Note that the **list_clipboards** command also now invokes the Clipboards window, which displays the same information presented by the modal Select Text to Paste dialog. If you prefer to use this dialog, use the **old_list_clipboards** command.

- **New Language-Specific File Options** - Most of the options regarding loading and saving files in SlickEdit are global, such as **Expand tabs to spaces**. But sometimes you may need to work with files that have different file load and save options from the defaults. For example, when working with makefiles, tabs should never be expanded to spaces. Now, load and save file options such as this are available on a language-specific basis. To access them, from the main menu, click **Tools > Options**. Expand **Languages** in the tree, then select your language category and language, and click **File Options**. New language-specific file options also include **Load as Binary**, **Save as Binary**, and **Strip trailing spaces**.

SlickEdit® Version History

Additional Enhancements and Changes

- **Beautify, Refactor, and Debug**
 - SlickEdit now ships the Mingw version of GDB 6.7.1 on Windows to eliminate the dependency on the Cygwin libraries.
 - Alt-hotkeys have been added for the buttons on the Refactoring Results dialog that appears after a refactoring operation.
 - The Debug tool window formerly called "Classes" has been renamed as the "Loaded Classes" tool window.
 - A **Contents** menu item has been added to the right-click context menus of all Debug tool windows that contains options for saving, copying, and printing the information.
 - The **debug_send_command** now outputs to a file or edit window so results can be copied.
 - Fixed a bug that caused the debug Watch tool window to display incorrect values.
 - Fixed a bug that caused comments to get moved when beautifying if/else statements.
 - Fixed a bug that caused C# methods that come after an attribute to be indented improperly when beautifying.
 - Fixed a bug in Organize/Auto Imports where imports for inaccessible package-private Classes could be inserted.
- **Context Tagging®**
 - Tagging operations run off current identifier - Previously, Context Tagging® relied on word characters to find identifiers in the code being analyzed. Now, Context Tagging operations are based on the current identifier. If you are creating color coding support for a new language, you can adjust ID start and follow characters by using the **Identifiers** options on the **Tokens tab** of the language-specific Color Coding options screen (**Tools > Options > Languages > [LanguageCategory] > [Language] > Color Coding**).
 - A new Context Tagging option is available for all languages: **Go to Definition ignores forward class declarations**. This option is on by default. When on, Go to Definition will filter out forward class declarations and only show you the actual class definitions. Note that when this option is off, turning on **Go to Definition navigates to symbol definition (proc)** will still navigate directly to the class definition, as it did before. However, **Go to Definition navigates to symbol declaration (proto)** will not navigate directly to forward class declarations (because you never want to actually do that by default). The option also appears on the Select a Tag dialog on first use, like the other Go to Definition options. See "go to definition" in the **Help > Index** for more information about the feature.
 - Go to Definition has been enhanced with better centering.
 - The **grep_tag** command now supports wildcards and Perl regular expressions.
 - List Members is now supported for PHP. Local variables in PHP (those that are not explicitly declared and the function parameters) are now added to the tag files.
 - Fixed a problem that caused the codehelp_complete command to duplicate the identifier.
 - Fixed a bug that caused the Preview window to fail to update during Auto List Parameters.
 - Fixed a bug that caused Auto Tag to fail when "Other directory" was selected.
 - Fixed a bug that prevented the C/C++ Preprocessing dialog from forcing buffers to update on **Apply**.
 - Fixed a bug that caused SlickEdit to hang when resizing the References tool window.
 - Fixed a bug that caused the Reference tool window, when in AutoHide mode, to freeze during a symbol search.

SlickEdit® Version History

- Fixed a problem with type expression result caching.
- Fixed a problem that resulted in incorrect tagging of ASP code.
- Fixed a bug that prevented tagging of numbers in Bourne shell functions.
- Fixed a bug in **_makefile_GetTargetList()** that resulted in all targets not being identified.
- Fixed a bug that prevented generated imports from screening out public/protected imports.
- Fixed a bug in COBOL that prevented symbol info when the word "set" appeared in Line 1.
- Fixed a bug that caused Assembly tagging for NPASM to miss @macro, @equ, struct, ends.
- Fixed a bug that caused single-click in the Symbols tool window to highlight the wrong reference in the editor window.
- Fixed a bug that caused the PHP parser to improperly handle **define()** constants.
- Fixed bug where the Preview window would automatically activate from the Find Symbol tool window even if it was not active.
- **Find and Replace**
 - Wildcard regular expression syntax now supports #, which matches any single digit (0-9).
 - The **gui_goto_col** command has been added to the Search menu (**Search > Go to Column**). This makes it possible for you to record a macro to position the cursor on a particular column (which can also be done by clicking on the status line).
 - A new command has been added, **find_current**, to complement **find_next** and **find_prev**. Use this command to jump to the currently selected reference.
 - The Color Coding Search Options dialog (accessed by clicking the **Color** button on the Find and Replace tool window) has been changed to show the "modern " names for the sets of symbols. Therefore, the labels for Symbol 1-4 are now Punctuation, Lib symbols, Operators, and User-defined.
 - Fixed a bug that caused a Slick-C stack when pressing Enter on the SlickEdit command line when Search output was active.
 - Fixed a bug in Brief regular expressions where \om \ol tokens in lowercase were not handled correctly when searching or using the Regex Evaluator.
 - Fixed a bug that caused a Slick-C stack when using Find and Replace in a Visual Source Safe project where none of the files were checked out.
- **General**
 - Two enhancements have been made to the Key Bindings options (**Tools > Options > Keyboard > Key Bindings**). First, a new button is available, located next to the **Search by key sequence** box at the top, that prompts for a list of mouse events that can have bindings, including WheelUp and WheelDn. If you want to add a modifier to the event (Ctrl, Alt, Shift, Command), hold down the modifier key(s) when clicking on the mouse event button, and that modifier will automatically prefix the events in the list. Second, Key Binding options have been enhanced to better handle those SlickEdit commands that work on a range of keys. For example, in CUA emulation, the **alt_bookmark** command is by default bound to Ctrl+0 through Ctrl+9. Ctrl+0 creates a bookmark named "0", Ctrl+1 creates a bookmark named "1", etc. Formerly, Key Binding options did not display any command bindings based on a range of keys, and therefore the command appeared as unbound in the list. Now, these bindings are shown in the list, with the ranges for these commands shown as "Ctrl+0 -> Ctrl+9". These bindings also now appear when you use the **Export** and **Save Chart** features. When unbinding one of these commands, SlickEdit displays a message asking if you want to remove the entire range from the command. There is no way to remove a single key from within the range; however, you can rebind a key to a different command.

SlickEdit® Version History

- SlickEdit now supports DocBook as a language. DocBook is a DTD set that includes both SGML and XML versions which are used to create technical documentation. Support has also been added for XHTML.
- The Emulation mode is now transferred when upgrading (no more prompt).
- The Fundamental language mode has been renamed as "Plain Text" mode.
- A Binary language mode has been added for viewing non-text files in hex mode.
- Delete Code Block has been enhanced. Now when you use **delete_line** or **cut_line** and are prompted with the Delete Code Block dialog, you can just press the key you used to initiate the cut_line to select just to delete the line. This makes it much easier to chain together cut_line's even if you have this feature turned on, since you can use cut_line, get the dialog, use cut_line again to specify to just delete the line, and then continue using cut_line to delete subsequent lines.
- The **edit_associated_file** command has been added to the **Document** menu and the right-click context menu (**Document > Edit Associated File**). It has also been enhanced to support WPF (XAML) files, and the finding of files in different directories.
- In Ada, the **ada_enter** command now splits lines of syntax-expanded statements.
- SlickEdit now detects UTF-8 files without signatures.
- Bookmarks, breakpoints, and annotations are now retained when beautifying, refactoring, auto-reloading, multi-file diffing, and reverting code, and SlickEdit attempts to relocate the markers.
- A Delete button has been added to the Bookmarks tool window to delete a selected bookmark.
- A new bookmark command is available: **bookmark_stack** (also on the main menu at **Search > Bookmark Stack**). This displays your list of pushed bookmarks and allows you to navigate to a pushed bookmark deeper in the stack. It also allows you to delete pushed bookmarks. Additionally, the Go to Bookmark dialog has been enhanced. When the Go to Bookmark dialog is displayed and you select a bookmark to jump to, SlickEdit pushes a bookmark at the current location.
- A new tool window option is available: **ESC dismisses floating window** (View > Toolbars > Customize, Toolbars tab). When checked, pressing Esc on a floating tool window (not a docked or auto-hide window) will dismiss the tool window as if it were a dialog. When unchecked, pressing Esc on a floating tool window puts focus back to the active MDI child. Note that this option applies to floating tool windows only (not toolbars). Note that pressing Esc on a docked tool window puts focus back to the active MDI child. Pressing Esc on an auto-hide tool window will hide the tool window.
- Color Coding now provides the capability to import lexer definitions from any VLX file. Use the **Import** button on the language-specific Color Coding options screen (**Document > [Language] Options**).
- The **gui_draw** command (Draw Lines dialog) has been enhanced to include an ASCII line drawing option.
- A new configuration variable, `def_save_config_on_autosave`, has been added for saving the SlickEdit config if necessary on the autosave timer. The default value is set to off. Use **Macro > Set Macro Variable** to enable the option.
- DIFFzilla has been enhanced so that Shift+Enter, Ctrl+Enter, and Ctrl+Shift+Enter work properly in diff mode.
- The Visual Studio emulation has been renamed as the "Visual Studio default" emulation. This change is because Visual Studio has multiple sets of key bindings. They prompt for the kind of development you will be doing and assign a set based on that. If you select "General Development", you are assigned their "default" emulation, which is the emulation supported by SlickEdit.
- Fixed a bug that, when switching out of Full Screen mode, unmaximized windows that were maximized and cleared the contents of certain tool windows.

SlickEdit® Version History

- Fixed a bug in embedded language coloring support that prevented the scriptlet end tag from being detected in HTML, resulting in incorrect coloring.
- Fixed a bug that resulted in a Slick-C stack when committing in Subversion.
- Fixed a bug that caused a Slick-C stack when attempting to access extension-specific options for TLD or XSD files.
- Fixed a bug that caused the **xml_validate** command to crash the editor when DTD's directory name contained foreign characters.
- Fixed a bug that prevented removal of personal comment annotations.
- Fixed several issues with Block Comments that resulted in errors, redundant behavior, alignment issues, and extra spaces.
- Fixed several issues with Comment Erase that resulted in incorrect indentation, blank lines, and lost tab characters.
- Fixed a bug that prevented binding of the 5 key on the numeric keypad.
- Unsurround now indents line comments as well as code.
- Fixed a bug in Word Completion that caused unexpected buffer modification.
- Fixed a bug in Dynamic Surround that prevented correct use of Auto-Complete.
- Fixed a bug that prevented Auto-Complete from working initially in Javadoc comments.
- **Slick-C®**
 - The Save Macro dialog (which appears automatically when you stop macro recording) has been enhanced. Now, when you attempt to save a macro with a name that is already taken, SlickEdit displays a message asking if you want to overwrite the existing macro. The Save Macro dialog also has a new button: **Save and Bind to Key**. This saves the new macro and allows you to quickly create a key binding for it with the Key Bindings dialog. You can also create/change key bindings for recorded macros by clicking **Bind to Key** on the List Macros dialog (**Macro > List Macros** or **list_macros** command).
 - Structs now behave like Slick-C classes and are their own data type, instead of being represented in the interpreter as arrays.
 - Slick-C now supports ISO C struct initializers.
 - Wildcard regular expressions are now supported for **parse**.
 - A **foreach** statement has been added to Slick-C, that works with arrays, hash tables, strings (same as Bourne shell), structs (iterates over the fields of the struct), and classes (if instance of `Iterable`, otherwise like structs). There is an optional key, which is useful for hash tables. The value may be omitted (`key => . in ht`). **value** and **index** can be auto-declared using the **auto** keyword. If **value** is auto-declared, its type will be inferred from the type of the collection. The implementation uses **_nextel()**. The syntax of **foreach** is:

```

foreach
    ( v in a ) {
        statements;
    }

```
 - Slick-C now provides the interfaces **sc.lang.IEquals** and **sc.lang.IComparable**, which the compiler and interpreter hook into.
 - Slick-C now provides the interface **IToString** for implicit string conversion. If a class implements `IToString`, then an instance of that class can be implicitly converted to a string, without calling the **_toString()** method.
 - Slick-C now supports the overloading of operator **[]** and operator **:[]**.

SlickEdit® Version History

- New #pragma options have been added:
 - **#pragma option(twopass,on)** - Specifies that the Slick-C compiler do a two-pass compilation. This allows the compiler to verify all function call signatures even for functions that are declared later in the file.
 - **#pragma option(strictarglists,on)** - This makes it an error to have implicitly typeless arguments, for example: `void first_char(s) { return substr(s,1,1); }`
 - **#pragma option(strictincludes,on)** - Verifies that all #import and #include statements precede any real code in the current module. This is required for two-pass compilation.
 - **#pragma option(strict2,on)** - Second generation of strict Slick-C compilation checks. All the options in #pragma option(strict,on), plus the three new options above.
- Slick-C hash tables now support indexing by class objects.
- The Slick-C const declaration now works with Slick-C classes and namespaces. It is no longer be deprecated by default, constants support type inference (so the compiler can tell ints from booleans from strings), constants work in namespaces and classes, constant names and values are stored in the state file in order to allow introspection, and now tagging recognizes the const declaration correctly.
- Slick-C now provides better support for global variable initialization across modules.
- Slick-C now includes **namespace default;**, which will return you to the "default " global namespace.
- Fixed a bug that caused switch statements in Slick-C to have the incorrect scope.
- Fixed a bug in Slick-C that caused a Slick-C stack when initializing a class containing a default-initialized array instance.
- Fixed a bug in Slick-C that resulted in incorrect code generation with **_makeempty()** causing a Slick-C stack.
- Fixed a bug that caused strings with integer values to be treated as **int**, resulting in a Slick-C stack.
- Fixed a bug that caused a Slick-C stack when initializing an instance of an empty class that extends another class.
- Fixed a bug that prevented Slick-C namespaces from being beautified according to the specified brace style.
- **Vim Emulation**
 - In Vim emulation, you can now press Esc to not only dismiss any codehelp windows, but also enter Command mode. A new configuration variable, **def_vim_esc_codehelp** (off by default), is available to turn this feature on and off.
 - Fixed a problem in Vim emulation (or when using named clipboards) in **list_clipboards**.
 - Fixed a bug with WRAPSCAN/NOWRAPSCAN where toggling this setting and repeating a search with 'n' would not respect the new setting.
 - Fixed a bug where '*' and '#' would leave the cursor at the end of a word, instead of at the beginning.
 - Fixed a bug where 'y' in conjunction with a named clipboard was not working properly.
- **Workspaces/Projects, Build, and File Operations**
 - Organize Imports now supports Annotations and Enums, properly screens out package-private classes, and allows for static imports.
 - The Customize Project Types dialog (**Project > New > Customize**) now shows the types in a tree structure for easier viewing and navigation.
 - A browse button has been added to the Link Order dialog (**Project > Project Properties, Compile/Link tab**, click the button to the right of **Libraries/Objects**).

SlickEdit® Version History

- Fixed a bug that caused the Files tool window to keep the modified flag even after saving with Save As.
- Fixed a bug that prevented **save_all** from working when all windows were iconized.
- (Linux only) Fixed a bug in Save As that prevented population of the **File name** field with the name of the current buffer.
- Fixed a bug that caused the **File name** combo box on the **File > Open** dialog to result in an error when using TweakUI to turn off **Common Dialogs > Remember previously used filenames**.
- Fixed a bug that prevented files added to a project via the Open tool window from being displayed in the Files tool window.
- Fixed a bug in the Files tool window that prevented closing of an untitled buffer.
- Fixed a bug that caused the Preview tool window to function improperly for the current item in the Files tool window.
- The vsbuild utility calculates a CRC which it stores and uses to compare against the previous compile so that it can determine when compile or link options changed. Previously, CRC was case-sensitive, resulting in unnecessary forced rebuilds. Now, strings are up-cased before the CRC is calculated on platforms where files are case-insensitive.
- A new configuration variable is available: **def_fast_auto_readonly**. When set to 1, this option speeds up the Auto read only feature by only checking the attribute on disk (not opening every file).
- Fixed a bug in the Tools tab of the Project Properties dialog (**Project > Project Properties**) that prevented the Move arrows from functioning.
- Fixed a bug that prevented **Build > Configurations** from showing configurations added/removed in **Project > Project Properties**.
- Fixed a bug that caused the **Current Project** and **Current Configuration** toolbar boxes (**View > Toolbars > Customize > Categories**) to lose content on SlickEdit restart, when the Projects tool window was not docked.
- Fixed a bug that prevented the Open tool window from respecting Explorer shell settings for the display of system files.
- Fixed a bug that prevented File Tabs from updating correctly for modified buffers.
- Fixed a bug that caused SlickEdit to close open files when using **next_error**, when the option **Automatically close visited files** was on.
- Fixed a bug where the Java Package view in the Projects tool window would not work properly if the Location field for a project was manually modified.

Version 13.0.1

Important Notes

- To provide more consistent functionality on all supported platforms, the Activations licensing option has been removed from SlickEdit 2008 in the v13.0.1 release. If you have previously used Activation to authenticate your license you will need to download a license file, which is available on your registered products Web page. If you were using a license file for v13.0, that license file will still work with v13.0.1. We have made some changes to where the license files are stored, which may cause SlickEdit to think your product is not licensed. If the License Manager is displayed when you first run v13.0.1, you can use it to browse for the location of your existing license file.
- SlickEdit version 13.0.1 uses GDB 6.8. You can get the tarball from our Web site here: www.slickedit.com/gdb. A patch file is also available on the site that contains a performance optimization we have added to GDB that

you can optionally use if you are attempting to build GDB 6.8 for yourself identical to the one we have built. The optimization has been submitted to GDB.

Appearance Enhancements

- Language-specific View options have been added to the Options dialog (Special Characters, line numbers, Hex, etc.). These match the view settings that formerly could only be set per-document (**View** menu items). To get to the language-specific options, from the main menu, click **Tools > Options > Languages > [LanguageCategory] > [Language] > View**. Additionally, a new language-specific View option has been added for setting the line number area width. You can set this a fixed with (the default is 6), or, you can use "Automatic" mode, which will adjust the amount of space in the prefix area based on what line in the file you are in. So if you're on line 4764, you'll have room for 4 digits. You can set a minimum width for this feature, so if you want to keep the margin from bouncing around, you can set a minimum and have it not go lower than that. The **View > Line Numbers** menu item has also been updated. When you turn on line numbers via this switch, SlickEdit checks for and uses your language settings. If you either have no language settings or your language settings are set to no line numbers, SlickEdit uses the default - fixed width line numbers.
- HTML Clipboard formats have been enhanced for background colors and color schemes with non-white backgrounds for pasting into documents with white backgrounds.
- The New Window Size dialog formerly accessed through the menu item **Window > New Window Size**, is no longer available and the options have been moved to the Options dialog (**Tools > Options > Editing > Editor Windows**).
- Fixed an issue with Selective Display preprocessor directives not working correctly.
- Fixed some small Color dialog bugs.
- Fixed a problem that caused missing bitmaps in dialog titlebars.
- Fixed an issue with **Selective Display > Search text** - RegExpr drop-down box was empty when restored from dialog history.
- Fixed an issue that caused the Color Coding listing scheme to show as modified when it was not.
- Fixed a color coding problem with current selected line color and line comments.
- Fixed a C++ beautifier error with do ... while with comment.

Building/Compiling/Running/Debugging Enhancements

- SlickEdit version 13.0.1 uses GDB 6.8. You can get the tarball from our Web site here: <http://www.slickedit.com/gdb>. A patch file is also available on the site that contains a performance optimization we have added to GDB that you can optionally use if you are attempting to build GDB 6.8 for yourself identical to the one we have built. The optimization has been submitted to GDB.
- In addition to Apache Ant, SlickEdit now supports NAnt, a .NET build tool similar to Ant. See "NAnt" in the **Help > Index** for more information.
- Added "-doNotCreateObjectDir" option to **vsbuild**.
- The Configure Error Parsing dialog, accessed from the menu item **Build > Configure Error Parsing**, is still available but the same fields and options are now also located on the Options dialog (**Tools > Options > Tools > Configure Error Parsing**).
- The C/C++ and Java Compiler Properties dialogs, accessed from **Project > Project Properties > Compile tab > Ellipses button**, are both still available but the same fields and options are now also located on the Options dialog (**Tools > Options > Languages > [LanguageCategory] > [Language] > Compiler Properties**).

SlickEdit® Version History

- The make program used on Execute Makefile Target is now customizable, so that users with custom make utilities can use standard makefiles. Formerly it was hardcoded to be "make". The fix was to add **def_default_make_program** and also refactor out the function **build_make_command**.
- Debugger options have been moved to the Options dialog (**Tools > Options > Debugger**). Formerly, a tabbed dialog was displayed when you use the menu item **Debug > Debugger Options** or the **debug_props** command. Now, this menu item/command displays the information about your debugger and provides a button to access the options, which can also be accessed through a new command, **debugger_options**. A **Debugger Info** menu item has also been added, right after **Detach**, since it is associated with a running debugger. This menu item is disabled when the debugger is not running.
- The Debugger Options dialog is now resizeable.
- Fixed a problem that prevented error markers from working in certain situations.
- Fixed a but that prevented Java files from compiling when saved in directories with foreign characters in the path.
- Fixed build issues with projects/workspaces/filenames with international characters.
- Fixed a bug that caused a Slick-C stack when closing the Memory tool window.
- Fixed the Memory tool window so that it uses the same font as editor (SDBCS font).
- Fixed a problem in the Memory tool window that prevented hex editing for small windows of memory.
- Fixed a problem with debugging that allowed you to attempt to expand locals when not suspended. The Locals, Members, Watches, and Autos tool windows should not allow you to expand variables or modify variables when the debugger is not in a suspended state.
- Fixed a problem with debug mouse-over expression evaluation that caused it to change current stack frame used by GDB.
- Fixed some typing issues in Watches tool window.

Context Tagging® Enhancements

- An option has been added to control the max number of references found in a references search.
- Some optimizations have been added to **debug_get_mouse_expr()** to cache previous result. This will increase performance when the option **Show info for symbol under mouse** enabled. Note that searching through huge workspaces still takes time, but it is faster than before.
- Support has been added for managed C++ **^** (**^** is equivalent to **__gc***).
- Tagging now supports new managed C++ keywords: **ref class Name { ... }**, **value class Name { ... }** and **interface class Name { ... }**.
- Tagging now recognizes files that start with "process" as COBOL.
- The Select a Tag dialog has been renamed to Select Symbol, and the size is now remembered.
- Adding or removing files from the workspace tag file is no longer allowed.
- Support has been enhanced for Visual Basic for Applications.
- An option has been added to make **codehelp_complete** (Ctrl+Space) case-sensitive (see **Tools > Options > Languages > [LanguageCategory] > [Language] > Context Tagging**).
- The Current Context tool window now shows the function even if the cursor is in a preceding comment.
- A new configuration variable is available to sort the Current Context tool window by line number instead of strictly by tag names.

SlickEdit® Version History

- Context Tagging now supports List Members for C99-style designated initializers, provided you do not mix them with positional initializers.
- Fixed auto completion problems when accessing outer class private members from inner class.
- The Class browser find (**cb_find**) now screens out definitions, and now when you are prompted for duplicate symbols, you are prompted using the standard "Select Symbol" dialog instead of sellist. Also, if there are no matches, it will just give you a message box saying the declaration was not found; it will not take you to the old Find Tag (**_pushtagbookmark_form**) dialog.
- Fixed a bug that caused AutoComplete to complete unpreferred choice.
- Fixed a bug that caused previous version tag files to hang in subsequent later versions.
- Fixed a bug that caused a silent crash when editing *.tagdoc file.
- Fixed a refresh issue in the Current Context tool window that was causing flicker and "flaky" behavior when scrolling with the scrollbar.
- SlickEdit can now parse "implements" clause in PHP.
- Fixed bugs in tag_tree_type_is_statement() and tag_tree_type_is_data().
- Fixed an issue with tagging becoming confused with hanging if statements in VB code.
- Fixed a bug that caused call tree to display multiple instances for classes.
- Fixed a bug that caused seek position of enumerator constants in C/C++ to be off by one.
- Fixed a bug that caused local variable information to be wiped out when switching buffers.
- Fixed a bug that caused local variables that appear to be prototypes not being short-circuited in tagging search.
- Fixed a bug that prevented **next-tag** from expanding Selective Display.
- Fixed a performance problem with integer-laden source file.
- Fixed a problem that caused references to not be correct when tag file is only slightly out of date.
- Fixed a bug that caused Select Symbol (formerly Select a Tag) to not offer options when invoked from mouse-over info.
- Fixed a bug that caused a Slick-C stack in mouse-over code.
- Fixed a bug that caused Slick-C tagging to not parse prototype for "call".
- Fixed some performance issues when searching for references to **p_completion** in `sysobjs.e`.
- Fixed some issues with slowness when editing very large file.
- Fixed a bug that caused static global variables in other modules to appear in tag navigation when they shouldn't.
- Fixed a bug that caused the Tag Files dialog to omit files that were on non-existent paths.
- Fixed a bug that caused Find Symbol to not respect the "Current File" option.
- Fixed a bug that caused the C/C++ source tagger to treat all C and C++ files as if they were just C++ files.
- Fixed issues with Go to Definition being inconsistent.
- Fixed bugs causing performance issues in the Class tool window (inner classes and sort by name).
- Fixed problems causing infinite loop when removing files from a tag file.
- Fixed issues causing the find tag preferences for Go to Definition/Declaration being reset.

SlickEdit® Version History

Documentation Enhancements

- Many sections in the documentation have been expanded and improved. Notably, the "Working With Files" section has been improved and includes a new section about the working directory (see **Help > Index** keyword: "files"), more information has been added and terminology defined in the "Programmable Macros" section (see **Help > Index** keyword "programmable macros"), the language-specific documentation for Java has been improved and expanded, and the *Slick-C Macro Programming Guide* has been enhanced to cover recent code enhancements and changes.

Editing Enhancements

- Language support has been added for Windows PowerShell, found under **Tools > Options > Languages > Scripting Languages**. Support includes Color Coding, Syntax Indent, Syntax Expansion, and Context Tagging for functions, filters, and aliases.
- Language support has been added for SystemVerilog (an extension of Verilog) and Vera. Support includes Color Coding, Syntax Indent, and Syntax Expansion (note that support for projects and Context Tagging have not yet been implemented). To set options for these languages, go to **Tools > Options > Languages > Hardware Description Languages**.
- The option Automatically close visited files formerly located at Tools > Options > Appearance > General has been moved to Tools > Options > Editing > Search > Bookmarks.
- You can now create a new lexer when adding a new language (**Tools > Options > Languages > Language Manager**).
- The commands **goto_bookmark/gb** and **delete_bookmark** now support command line completion.
- Support for the **vimtutor** command in Vim emulation has been added. This command opens a tutorial file that you can edit as you learn Vim commands.
- A Search option has been added to the Option dialog to push a bookmark when using **top_of_buffer/bottom_of_buffer** (**Tools > Options > Editing > Search**). Formerly only the **def_top_bottom_push_bookmark** configuration variable was available to set this option.
- Project support has been enhanced and results of **GetProjectFiles()** cached for better performance.
- Syntax Expansion has been enhanced to be more tagging-aware.
- A new item has been added to the editor's right-click context menu: **Show file in Projects**. This shows the current file in the Projects tool window. If the file is in a collapsed node, the node is expanded to show the file. If the file occurs in multiple projects, you are prompted to select which one(s) to expand. A check box in the prompt allows you to just expand all relevant projects.
- A new configuration variable is available: **def_max_mffind_output def-var**. This sets the maximum size for search results before switching to listing matching files only.
- A new Search option is available on the Options dialog (**Tools > Options > Editing > Search**): **Maximum search results output (KB)**. Use this option to specify the maximum amount of search results, in kilobytes, to return after a search operation.
- New escape sequences have been added for Aliases: **%\un** (line not included if function params are expanded; included if function params are not expanded), **%\vn** (line not included if return types are expanded; included if return types are not expanded), and **%()**, used to separate identifier characters: for example, **%\u%()n** has the effect of the **%\u** option followed by a literal "n". It is recommended that **%()** be used to separate alias escape sequences ending with a letter from other identifier characters so that new aliases escape sequences won't break existing aliases you have. Don't write **%\dx**. Write **%\d%()x** instead.

SlickEdit® Version History

- Annotation dates are now entered and changed using a calendar dialog. The displayed date is no longer hand-editable.
- Fixed a bug that caused Tab indent after double colon to indent # SPACES from the middle colon.
- Fixed a bug that prevented Smart Tab from working as expected in the Emacs emulation.
- Fixed a bug that prevented indenting from working for PERL embedded in HTML.
- Fixed a bug that caused SlickEdit to convert "public " to "public:".
- Fixed a bug that prevented JSP document mode from working properly after editing the Java portion.
- Fixed refresh issues in tool list on Project Properties dialog.
- Fixed a bug that caused a stack when using **copy-to-clipboard** and **copy-to-cursor** (Text mode).
- Fixed a bug in Project Properties where SlickEdit would not initialize the directory browser with the correct path if the path was relative to the project file instead of being an absolute path.
- Fixed a bug that caused a Slick-C stack in projects with wildcards.
- Fixed an issue that caused SlickEdit to expand while after a **do** loop.
- The `.i` extension is now associated with C/C++.
- Fixed a bug that caused a Slick-C stack when matching preprocessing in ANSI-C.
- The Auto-Complete options **Auto select unique items** and **Insert selected** are now mutually exclusive.
- Fixed a bug that prevented block comment border settings from preserving blanks.
- Fixed a bug that caused **goto-bookmark** to navigate to wrong place.
- Fixed a spin control error in C++ options - Comment Wrap.
- Fixed a bug that caused a stack in COBOL when starting embedded language (via EXEC) for a language that does not exist.
- Fixed a problem that caused an HTML file to shift focus when entering the > character.
- Fixed a bug in comment skeleton and wrapping in embedded languages.
- Fixed a problem in Dynamic Surround when typing braces for a function definition.
- Fixed a problem with HTML sometimes inserting two closing tags.
- Fixed a bug in Indent with Tabs not being honored when inserting to, pasting to, or deleting from text past end of line column.
- Fixed a bug in that project open commands were not ran for associated workspaces/projects.
- Enhanced **removeDuplicateFunctions** to support more style signatures.
- Fixed a bug that caused SmartPaste and Reindent to leave out blank line below paste in some languages.
- Fixed a problem that caused a Slick-C stack when replacing words.
- Fixed a bug in Vim emulation that prevented the IGNORECASE option from working correctly with **:s** command.
- Fixed a bug in Vim emulation that caused **:w** to prompt user to overwrite current file with selection when it should not.
- Fixed a few issues with **complete-prev** (Word Completion).

SlickEdit® Version History

- Fixed the Replace tool tip so it appears at correct location when searching with option cursor at end of match.
- Fixed the Regex Evaluator so it doesn't lockup if match is found when using `\c` (place cursor) token in regular expression.
- Fixed previous selection so it is always restored in Replace.
- Fixed a problem that prevented changing the search option **Wrap at beginning/end**.
- Fixed Replace in selection so that BLOCK selection is restored.
- Fixed inappropriate completion list for (quick-)search/replace.
- Fixed a scrolling bug with Replace - Preview All.
- Fixed Alias problems when using "tabledesc" and "table" parameter names.
- The Alias escape sequence `%\o` now returns full signature.
- The `%\n` and `%\o` escape sequences now work inside doc comment alias expansions.
- Fixed a bug that caused the HTML symbol translation editor to show alias editor for another language.
- Fixed a problem with Bookmarks so that it notifies the Preview window on focus.
- Fixed a problem with **pop-bookmark** missing the target.

Files Enhancements

- File tabs have been enhanced so that, when adjacent files differ only by file extension, the name of the file on the tab is abbreviated to only show the extension. This saves space and provides better visibility for associated files. For file names to be abbreviated in this style, their paths and base file name must match exactly. For example, `C:\rectangles\BorderRectangle.cpp` would not abbreviate with `C:\src\include\BorderRectangle.h`. You can turn this feature on and off for all file tabs by right-clicking in the File Tabs tool window and selecting **Abbreviate similar files** from the context menu.
- A **close_others** command has been added and available on the right-click context menu of file tabs. This closes all other files except the one you've clicked on.
- Some file tabs, like those for search results buffers, build output, and File Manager operations, display a picture in their file tabs by default. An option has been added to the right-click context menu of file tabs to show or hide these pictures.
- Backup History options (**Tools > Options > File Options > Backup**) have changed: 1) The option **Make backup files** now has three available settings: Select **Create backup history on save** to use Backup History. This creates a version of the file each time you save, allowing you to compare and restore to earlier versions. Select **Create backup file on first save** to create a single backup file that preserves the file contents prior to editing. Or select **None** for no backup history. 2) The option formerly named **Backup directory option** has been renamed as **Backup location**. This specifies where the backup files will be created. This option is not available when using **Create backup history on save**. If you select **Global directory** or **Global nested directories** you will need to provide a value for **Backup directory path**. 3) The option **Backup directory path** is used to specify the full path to the location used for creating backup files. This is used only when the value for **Backup location** is **Global directory** or **Global nested directories** or when the value for **Make backup files** is set to **Create backup history on save**. Press **Delete** to clear the value and use the default.
- The **Save configuration** option (**Tools > Options > Application Options > Exit**) now has an additional choice: **Save configuration immediately**. When this value is set, configuration changes are immediately saved when they are detected. Also:

SlickEdit® Version History

- Fixed a bug that prevented the Print dialog from properly setting the **def_tprint_command** variable (**print-selection**).
- Fixed color printing/preview so that it functions correctly in Windows.
- Fixed a bug that caused **close-all** to not save buffer positions the way close does.
- Fixed a problem that caused files not being removed from File Tabs when closing a file using the Files tool window.
- Fixed a problem with Backup file options with regard to **def_maxbackup**.
- Fixed an erroneous error message "bottom of file reached...".
- Fixed a bug that prevented the scanning of open buffers to see if any files should be remapped after creating a new extension.

Keyboard Enhancements

- New key bindings have been added for Emacs-like S-expression navigation that work in all emulations: **next_sexp** (**Ctrl+Alt+Right**) and **prev_sexp** (**Ctrl+Alt+Left**) can be compared to **next_word** and **prev_word**, except that they treat blocks as words. **backward_up_sexp** (**Ctrl+Alt+Up**) is the shortcut for navigating to the start of the immediately enclosing block (in Lisp, this would be the open paren). **forward_down_sexp** (**Ctrl+Alt+Down**) is used to drill into a block. If the cursor is at the start of a block, it moves the cursor to the first S-expression within the block; otherwise, it behaves like **next_sexp**. **select_prev_sexp** (**Ctrl+Alt+Shift+Left**) and **select_next_sexp** (**Ctrl+Alt+Shift+Right**) extend a character selection from the cursor position to the start of the next or previous S-expression, respectively. Like the **Shift+Cursor** commands do to CUA style selections, if executed repeatedly, they will extend the current character selection. **cut_prev_sexp** (**Ctrl+Alt+Backspace**) will delete the S-expression to the left of the cursor and copy it to the clipboard. Again, if called repeatedly, this will prepend the subsequent deletions to the clipboard.
- Fixed a performance issue in the tree control for Key Bindings filtering.
- Fixed a problem that, if VK_APPS (Windows menu key on right side b/w Windows key and Ctrl key) was unbound, a default Windows menu came up.
- Fixed an issue that prevented shifted menu hotkeys from working.

Macros Enhancements

- Fixed an issue that caused the reloading of `stdcmds.e` (in hotfix or using **load** command) to clear dialog history.
- Fixed several issues with incorrect macro recording regarding Dynamic Surround, List Members, and Auto-Complete.

Miscellaneous Enhancements

- You can now generate GUIDs in SlickEdit. A menu item has been added to the Tools menu: **Tools > GUID Generator** (**gui_insert_guid** command) or you can use a command line version of the tool with the **insert_guid** command. Select from seven formats to generate the new GUID and optionally insert it at the cursor location.
- Fixed a bug that caused a stack when clicking Language manager buttons without selection.
- Fixed a bug that caused the look up of reference from history drop-down to fail.
- Fixed a bug that prevented **_use_timers** from being restored on editor startup.
- Fixed a problem with too much margin on left side of tree control.

SlickEdit® Version History

- Fixed a refactoring problem that prevented parsing expression containing **(A[0])++**.

Options Dialog Changes

- All of the Advanced language-specific options have been moved to the General language-specific option node (**Tools > Options > Languages > [LanguageCategory] > [Language] > General**).
- An **Expand All Children** option has been added to the right-click context menu of the Options dialog tree. This only acts on the current node.
- Many keyboard navigation shortcuts have been added to the Options dialog, and the documentation has been updated (see **Help > Index** keyword "Options dialog shortcuts"). Also, Tab now works better for keyboard navigation.
- A new category, **Hardware Description Languages**, has been created under the **Languages** node of the Options dialog. This category now contains Verilog and VHDL (moved from the **Application Languages** category) as well as the newly supported SystemVerilog and Vera.
- Fixed an issue with the Options dialog that caused the **Apply** button to collapse expanded nodes.

Slick-C® Enhancements

- New class introspection functions have been added to Slick-C (**_typename**, **_instanceof**, etc.) See "Classes > Introspection" in the "Types" chapter of the *Slick-C Macro Programming Guide* (**Help > Contents**) for more information.
- Many **#pragma** options (such as **strictnames**) have been added in Slick-C. See "**#pragma**" in the "Preprocessing" chapter of the *Slick-C Macro Programming Guide* (**Help > Contents**) for more information.
- An **_insertel()** method for arrays and corresponding C API functions **vsHvarArrayInsertEI()** and **vsHvarHashtableInsertEI()** have been added to Slick-C.
- Fixed an invalid number argument Slick-C stack while stepping in the Slick-C debugger.
- Fixed a problem causing a crash in `vstw.exe` loading module.
- Added support for **#!** in Slick-C code - Slick-C does not have to do anything with **#!**, but it now allows you to have a **#!** at the top of the file without having an error.
- Fixed a bug that caused a crash when using **#error**.
- The following keywords are not used in Slick-C and have been removed: **by**, **end**, **endif**, **extproc**, **fkeytext**, **include**, **then**, **to**. Note that **end** is still an event name.

Version Control Enhancements

- Fixed a problem that caused Subversion diff to fail if prompted for password.
- Fixed a problem with version control that caused **DisplayOutputFromView** to leak views.
- Fixed a Perforce "Bad structure length" error.

SlickEdit 2007 (v12)

SlickEdit 2007 (v12.0.0) was released in Spring, 2007. With this version, SlickEdit changed the nomenclature for major releases.

Version 12.0.0

New Features in this Version

- **New Class Tool Window** - The new Class tool window provides an outline view of both the members of the current class as well as any visible inherited members, and also shows the inheritance hierarchy of the current class. This is useful for object-oriented programming languages such as Java™. Note that this is a new tool window, not to be confused with the tool window formerly known as "Classes," which has been renamed to "Symbols" in SlickEdit 2007. To use the Class tool window, select **View > Toolbars > Class**, or use the `activate_tbclass` command.
- **XML/HTML Formatting** - Content in XML and HTML files may be set to automatically wrap and format as you edit according to user-defined formatting schemes. A formatting scheme is comprised of any number of XML or HTML tags, each of which can be configured individually for indent levels, wrapping, and tag structure. Multiple schemes can be defined--for example, you may want one scheme for HTML files and another for XML files, or perhaps you are required to code certain files to various standards. Schemes can be saved and imported, so they can be shared with your team. Tags for each scheme can be entered manually or you can import tags from the current file. For information about enabling/disabling this feature and configuring settings, see the Help topic on **XML/HTML Formatting**.
- **Java Project Improvements** - Improvements were made to the Java Options dialog (**Build > Java Options**), and new menu options have been added to the **Projects tool window** (docked as a tab on the left side of the editor by default). Right-click on a Java package in the tool window to add a new enum, class, interface, or file.
- **Java Live Errors Using Java Compiler** – Java Live Errors identifies errors in your Java code as you type. Use the `rte_next_error` command to jump through all live errors in the current file. Our previous implementation used Jikes™ to compile your code, but Jikes does not support JDK 5 and later. Java Live Errors now uses javac from Sun Microsystems™ JDK 6. To activate Live Errors, JDK 6 must be installed on your system. The root directory of your JDK 6 installation must be specified on the Live Error tab of the Java Options dialog (**Build > Java Options**). If already installed, SlickEdit will attempt to detect that location automatically the first time a Java project is opened. See the Help topic on **Java Live Errors** for more information.
- **New Find Symbol Tool Window** - This new tool window (**Search > Find Symbol**) is used to locate symbols (tags) in your code. It allows you to search for symbols by name using either a regular expression, substring, or fast prefix match. See the Help topic on **Symbol Browsing** for more information.
- **Dynamic Surround and Unsurround** - Dynamic Surround provides a convenient way to surround a group of statements with a block statement, indented to the correct levels according to your preferences. SlickEdit enters Dynamic Surround mode automatically, immediately after you expand a block statement (for instance, by typing "if" then pressing Space). After expanding the statement, a box drawn around it as a visual guide, and you can pull the subsequent lines of code or whole statements into the block by using the Up, Down, PgUp, or PgDn keys. Pressing any other key will exit Dynamic Surround mode. To disable Dynamic Surround, select **Tools > Options > File Extension Setup**, select the **Indent tab**, and deselect the option **Use Dynamic Surround**. The Unsurround feature allows you to remove structures from a code block. Right-click on a selected code block and select **Unsurround**, or use the `unsurround` command.
- **Copy and Paste in Color** - SlickEdit now provides HTML and RTF clipboard formats, allowing you to paste formatted and color-coded text into other applications (as well as plain text). Select **Tools > Options > General**, then click the **Selections tab**. Select one of the **Clipboard format** options.
- **Enhanced Symbol Preview** - The tool window known as "Symbol" in previous versions has been enhanced and renamed to "Preview" (**View > Toolbars > Preview**) to be more consistent with its functionality. This window provides a portal for viewing information in other files without having to open them in the editor. It automatically shows this information when you are working with certain interface elements and features. The Preview window

SlickEdit® Version History

also features the "Documentation Comments Preview"—a pane that displays information and code comments for the symbol under the mouse.

- **Documentation Comments Preview** - This feature is implemented as a pane in the Preview Tool Window (formerly called "Symbol tool window"). Documentation Comments Preview displays information and code comments for the symbol at the cursor. This gives you a mouse-free way to see the same information provided by the option Show info for symbol under mouse (**Tools > Options > File Extension Setup, Tagging tab**), which shows the information in a screen tip when you hover over a symbol with the mouse.
- **New Files Tool Window** - A new Files tool window lets you view open buffers, project files, and workspace files, sortable by file name or path. Includes incremental search and file name filters. This feature replaces the Select a Buffer dialog found in previous versions. To use the Files tool window, select **Document > List Buffers**, or use the **list_buffers** command. Alternatively, you can select **View > Toolbars > Files**, or use the **activate_files** command.
- **Drag-and-Drop Support for KDE® and GNOME™** - For KDE and GNOME users, files can be opened in SlickEdit by dragging and dropping them from the file manager into the SlickEdit editor pane.
- **Microsoft® Windows® Vista™ Support** - SlickEdit now works with the latest Microsoft Windows operating system, Vista .
- **Line Ruler** - The **Draw box around current line** option (**Tools > Options > General, General tab**) has been enhanced. You can now choose a box with tab stops, syntax indent levels, or decimal points shown.
- **New Color Picker** - This is a more up-to-date color picker that appears when you change the **Foreground** or **Background** colors on the Color Settings dialog (**Tools > Options > Color**).

Additional Enhancements and Changes

- **New, Enhanced, and Better-Named Tool Windows** - If not already docked in the editor by default, you can toggle-display by selecting from the **View > Toolbars** menu.
 - The tool window known as "Classes" in previous versions has been renamed to "Symbols " (**View > Toolbars > Symbols**) to be more consistent with its intended functionality. It contains the symbol browser, which lists the symbols from all of the tag files.
 - The tool window formerly called "Symbol" has been improved, and renamed to "Preview " in this version to be more consistent with its intended functionality.
- **New Language Support**
 - ActionScript language mode now supports syntax highlighting and tagging. For Flash 8 and Flash MX 2004, SlickEdit will automatically generate global tag files for global classes and functions, using the global classpath as defined by the Macromedia Flash IDE registry settings for Windows platforms only. For Flash 8, it automatically chooses the Flash 8 packages for Context Tagging®. For ActionScript 3.0 (Flex SDK) Context Tagging support, source files for global classes will need to be tagged manually (see documentation for **Context Tagging**).
 - Python™ language mode now supports the following enhancements:
 - Statement Level Tagging (right-click in the Defs tool window and select **Show Statements**).
 - Select Code Block (**Edit > Select > Code Block** or **select_code_block** command). This feature selects the lines in the current code block (the definition of the current block depends on the language). Invoking this menu item or command multiple times in succession selects larger code blocks.
 - Smart Tab (**Tools > Options > File Extension Setup > Indent tab > When tab key reindents the line**).
 - Improvement in the tagging engine. Our Python tagging engine now tags local variables, variables declared within statements, and global variables declared within a function.

SlickEdit® Version History

- Find Matching Paren (Ctrl+G or **find_matching_paren**) also matches the colon (:) token and the end of context.

New Options, New and Enhanced Commands, and Dialog Box Changes

- An **Exclude** option has been added to the Add Tree dialog (**Project > Project Properties > Add Tree** and **Tools > Tag Files > Add Tree**). This option can be used to exclude files, file types, and paths using full path names or wild cards.
- A new menu option has been added to the main menu: **File > Export to HTML**. This is used to save the current open buffer as an HTML file with formatting and color coding.
- The tab formerly known as "Select Styles" on the General Options dialog (**Tools > Options > General**) has been renamed to "Selections" in this version.
- New clipboard format options have been added to the **Selections tab (Tools > Options > General)**. These are useful for pasting formatted and color-coded text to other applications (as well as plain text). Choose from RTF or HTML.
- The File Merge dialog (**Tools > File Merge**) has been updated to contain options for selecting from history, selecting a file, or selecting a buffer.
- The **GNU C/C++ Wizard** for creating new projects (**Project > New**) now provides a way to turn **-O** flags back off.
- The **comment** and **comment_erase** commands have been enhanced. If one of these commands is invoked when there is no active selection, the current line is commented or uncommented.
- The **cursor_error** command now looks in C++ compiler include directories.
- Commands and key bindings have been added for Bookmarks (**Search > Bookmarks**). The **alt_bookmark** command is used for setting a bookmark, and **alt_gtbookmark** is used for navigating back to it. Bookmarks set with these commands take their names from the key combination used to set them.
- The buffer navigation commands **top_of_buffer** and **bottom_of_buffer** can now push bookmarks so you can get back quickly. This is particularly useful when you just need to jump to the top of the file to look at your #includes. This option is not on by default. To use it, set the macro variable **def_top_bottom_push_bookmarks (Macro > Set Macro Variable)**.
- A **Quick Replace** operation been added. Quick Replace gets the current word or selection at the cursor, prompts for replacement text on the command line, then highlights each occurrence of the word and prompts if you want to replace the text. Right-click on a word or selection and select **Quick Replace**, or use the **quick_replace** or **qr** commands.
- A new command, **end_line_text_toggle**, has been added for jumping to the end of the line, but first stopping before trailing whitespace, and also stopping at the vertical line column (if applicable). A corresponding option has been added to the Redefine Common Keys dialog (**Tools > Options > Redefine Common Keys**). This is useful when you want to "fix" some long lines by trimming extra spaces, because it gives you a natural and quick way to get to your vertical line column and the last non-blank column.
- (UNIX®, Linux®) Dot files (files with names beginning with a dot character) are now hidden by default on UNIX and Linux platforms. An option has been added to the Open and Save As dialogs to change the default in order to show these files. This option is controlled by the **def_filelist_show_dotfiles** configuration variable. On Windows, the default value of this variable is 1. Change to 0 to view dot files. On UNIX platforms, the default value is 0. Therefore this option is on by default for Windows, and off by default on UNIX.
- A new command has been added to display the operating system's file manager, for example, Windows Explorer on Windows, Finder on Mac OS® X, Konquerer on Linux KDE desktop, and nautilus on Linux Gnome desktop. Select **Tools > OS File Browser**, or use the **explore** or **finder** commands. If you are editing a document, the file

SlickEdit® Version History

manager will be rooted in that file's directory, otherwise it will default to the current working directory. Using the - option after the command (eg: **explore -**) will ignore any file directory or working directory and go the system root.

- A new command has been added for Selective Display to copy only visible text. Select **View > Copy Visible** or use the **copy_selective_display** command. Normally, if you have lines hidden with Selective Display, copying a selection that spans multiple lines would also pick up those hidden lines. This command ignores those hidden lines and will only copy the visible text. This does not work with column (block) selections.

Tagging Enhancements

- Case-insensitive matching has been improved for List Members (**Tools > Options > File Extension Setup > Tagging tab**).
- SlickEdit can now tag source code in ZIP files.
- Support has been added for navigating to C++ destructors.

Version Control Enhancements

- Version control now includes non-branch tags in the history graph of a file.
- Version labels have been added to the version tree in the CVS history dialog.

Refactoring Enhancements

- Refactoring support has been added for the **__is_base_of(x,y)** type trait function found in Visual Studio® 2005.
- Refactoring support has been added for the implicit forward declaration of **type_info** found in Visual Studio 2005.
- The refactoring process for parsing a file has been made smoother.
- Refactoring now has the ability to handle **typedef** types with destructor member access expressions.
- Refactoring now displays an error when assigning to pointer to pointer to const.
- "Using" statements now pull in all overloads of a symbol.
- "Using" declaration now checks for already defined variables.

Miscellaneous Enhancements

- Color coding for selections has been enhanced. Font colors and styles are now preserved in selections and for current line coloring (**Tools > Options > File Extension Setup > Advanced tab**, select **Current line**). Predefined color schemes (**Tools > Options > Color**) now include less-obtrusive selection background colors.
- The number of lines or characters in the current selection is now displayed in the editor status area (located along the bottom right edge of the editor). If there is no selection, the indicator is dimmed, with the text "No Selection."
- Installation for SlickEdit on Windows is now accomplished through a Microsoft Installer Package (.msi). This provides a more manageable installation.
- SlickEdit now supports TrueType fonts on Mac OS X using the Xft library. The first time SlickEdit is invoked, Xft will need to build a font cache. Building the font cache may take several minutes, depending on system speed and number of fonts. Currently, while Xft is building the cache there will be no visual indication of progress. Building the cache will only be necessary the first time SlickEdit is run after installation.
- You can now close the Build window by typing exit directly within it.
- Support has been added for wildcard searches using **pos()** and **vsStrPos()**. Use the **&** option, just like **search()**.

SlickEdit® Version History

- Support for smooth vertical and center scrolling horizontally has been added to be more consistent with Windows applications.
- Support has been added for multi-column tree controls to display help tooltips on mouseover.
- The Slick-C® language has been enhanced. Please see this post on the SlickEdit Community Forums for more information: <http://community.slickedit.com/index.php?topic=691.0>.
- IPv6 hosts are now supported.

Documentation Enhancements

- The Help system (**Help > Contents**) and *SlickEdit User Guide* have undergone significant reorganization, with improved and expanded content.
- More cross-referencing links and tips and notes in color have been added.
- Documentation has been improved and expanded in many areas, including the topics "Workspaces and Projects" and "Aliases."
- The Getting Started Guide no longer ships with the product. Instead, a "Quick Start" topic is now included in the Help system and *SlickEdit User Guide* (**Introduction > Quick Start**).
- SlickEdit now ships with a PDF document titled "Slick-C Macro Conventions and Best Practices for End Users." It is located in the `docs` subdirectory of your installation.

Version 12.0.1

General Enhancements

- This release of SlickEdit includes the revised Key Bindings dialog. The following aspects of the Key Bindings feature were completed after the documentation deadline:
 - A **Run** button has been added to the Key Bindings dialog (**Tools > Options > Key Bindings** or `gui_keybindings` command). Click this button to run any selected command or user-recorded macro listed in the dialog.
 - The Key Bindings dialog is no longer displayed when you select **Macro > List Macro** from the main menu. Instead, a new List Macros dialog is displayed, containing buttons to run the selected macro, edit the source of the selected macro, delete the selected macro, or bind the selected macro (**Bind to Key** button). When you click **Bind to Key**, the Key Bindings dialog is displayed from which you can click **Add** to add a new binding for the selected macro.
- A new option, **Open in Current Window**, has been added to the right-click context menu in the Files tool window (**Document > List Open Files** or `list_buffers` command). This option provides the same functionality as **Window > Link Window** (`link_window` command), which is used to open a selected file in the current editor window. This is useful for viewing two files side-by-side in the same editor window. You can do this by splitting the window and then linking another file to it. To split an editor window, click **Window > Split Horizontally** or **Window > Split Vertically** (or use the `hsplit_window` or `vsplit_window` commands). To open another file in the same window, select **Document > List Open Files**, select the file you want to open, then right-click and select **Open in Current Window**.
- The Project Properties dialog (**Project > Project Properties** or `project_edit` command) is now resizeable, and the size and position are remembered between sessions.
- A new `align_selection_center` command is available for block-type selections (right-click and drag, click **Edit > Select Block** or use the `select_block` command) that will center-align the text enclosed in the block selection.

SlickEdit® Version History

You can also use the commands **align_selection_left** or **align_selection_right** to align the text to the left or right edges of the selection.

- Right-clicking in the Build tool window (docked at the bottom of the editor by default) and selecting **Send Compile Output to Editor Window** now opens the .process window in SlickEdit.
- The limit for the number of file tabs that can be displayed has been increased from 64 to 255. File tabs can be toggled on/off by selecting **View > Toolbars > File Tabs**.
- SlickEdit provides a configuration variable, **def_top_bottom_push_bookmark**, that can be set to push a bookmark whenever you jump to the top or bottom of the buffer. New functionality has been added so that even if this variable is set, no bookmarks are pushed when using the current buffer as a build window (.process buffer).
- A new **Preprocess File** button has been added to the Test Parsing Configuration dialog for C/C++ Refactoring (**Tools > C++ Refactoring > Test Parsing Configuration**) for preprocessing the file and sending the results to an editor window.
- SlickEdit now provides auto-completion of C-style comment block start and end characters. This is controlled by the configuration variable **def_auto_complete_block_comment**, which is set to TRUE by default. See the Help topic for **def_auto_complete_block_comment** for more information.
- SlickEdit now recognizes ColdFusion CFC files, which by default, refer to CFML (**Tools > Options > File Extension Setup**).
- The **q** command (which is the same as the **quit** command) now takes arguments for which file(s) to close, which is useful when you have too many open files. For example, **q c:\temp*.*** closes all files in `temp` directory that you had open.
- The view in the Files tool window (**Document > List Open Files**) is now remembered between sessions. The **activate_files** command has been changed to only activate the Files tool window, while a new command, **activate_files_files**, not only activates the tool window but changes the view to list files open for editing (as opposed to files in the current workspace or project).
- The option to **Automatically close visited files** on the General tab of the Extension Options dialog (**Tools > Options > General**) now works with Next Error (**Build > Next Error** or **next_error** command) and Previous Error (**Build > Previous Error** or **prev_error** command).
- Dynamic Surround (**Tools > Options > File Extension Setup > Indent tab**) can now be cancelled with a mouse click.
- The Font dialog (**Tools > Options > Font**) has been enhanced to be more accessible from the keyboard (Up/Down arrows now scroll through items in drop-down lists).
- Support is now provided for validating XML schema definitions and XML documents that use namespaces and schemas. Also added support for allowing validation to simply do a well-formedness check for XML documents that have no doctype or schema definition.
- The Generate Debug command now supports C#.
- Fixed a bug that caused SlickEdit to hang on C files where the last line contained no new line characters.
- (Linux) Fixed a bug that prevented **File > Save All** from saving all files.
- Fixed a bug in the Dialog Editor that caused the Picture property on an image control to return quotes in the file name.
- The **Files of type** drop-down list on the Add Workspace File dialog (**Project > Organize All Workspaces > Add Workspace**) now includes all supported workspace file types.
- Fixed a bug in the implementation of the restart attribute of the Hot Fix loader.

SlickEdit® Version History

- Fixed a bug that caused incorrect matching block color if the current line had a different background.
- Fixed a bug that caused Escape to cancel a selection when using Block Insert Mode (**Edit > Other > Block Insert Mode** or **block_insert_mode** command).
- Fixed a bug that caused auto-complete of "forea" (foreach) to insert an extra paren when using **Auto select unique items** (**Tools > Options > File Extension Setup > Auto-Complete tab**).
- Fixed a bug causing inconsistent behavior with SmartPaste® regarding public, protected, and private statements in C++.
- Fixed a bug that caused Surround With not to work when dashes were used in the alias name.
- Fixed a bug that resulted in file names/paths in the Defs tool window not being shrunk/unshrunk when resizing.
- Fixed a bug that caused problems when using Save As on a file that contained dashes in the name.
- Fixed a bug in Brief emulation that prevented the Search Results window from scrolling when using the **search_again** command (Shift+F5).
- Fixed a bug that resulted in a new Open command to be run immediately when clicking **OK** on the Project Properties dialog (**Project > Project Properties > Open tab**).
- Fixed a bug that caused a vertical line to be drawn in COBOL files when **Truncation** is on (**Tools > Options > File Extension Setup > General tab**).
- Fixed a bug that caused Backup History to fail for custom configuration directories with names beginning with a dot on UNIX.
- Fixed a bug that prevented some tool windows from being dockable.
- Fixed a bug that prevented **Window > Link Window** from switching to the correct buffer.
- Fixed a bug that caused a Slick-C stack when attempting to use the SlickEdit default selection style in BBEdit, CodeWright, and Xcode emulations.
- (UNIX) Fixed a bug that prevented dragging and dropping files with names containing brackets from Nautilus to SlickEdit.
- Fixed a bug in Color Coding that incorrectly inserted escape characters when adding a keyword that contained a space.
- Fixed a bug in Color Coding that prevented using double quotes in tokens.
- Fixed a bug that caused **count_lines_in_selection** to modify the selection unexpectedly in Brief mode.
- Fixed a bug that prevented %n from being replaced with the file name when choosing to build from within the Projects tool window.
- Fixed a bug that prevented SlickEdit from restoring its size correctly when maximized.
- (UNIX) Fixed a bug that prevented menu accelerator mnemonics from displaying correctly on UNIX platforms.
- Fixed a bug with **Make Jar** where the **jar** command was improperly built.
- Removed unnecessary restriction on **jar** packaging which required non-Java files to reside in the project directory.
- Fixed bug with Java Live Errors where it was possible for error -9108 to be thrown.
- Fixed a bug with Vim searching and macro recording where playing back a macro (using either standard macro playback or the "." command) involving a **d/search_string** command would prompt for *search_string*.

SlickEdit® Version History

Beautifier Enhancements

- A new option, **Indent access specifier** has been added to the **Indenting tab** of the C/C++ Beautifier (**Tools > Beautify** or **gui_beautify** command). When this option is off, specifiers are aligned directly underneath the class. When this option is on, specifiers are *indented* under the class.
- Fixed a bug in the C# Beautifier that prevented beautification of certain statements.
- Fixed a bug in the C# Beautifier that caused incorrect indentation in @"\" notation.
- Fixed a bug in the C# Beautifier that caused incorrect indentation of **foreach** if it was not in braces.
- Fixed a bug in the C# Beautifier that caused incorrect rendering of namespace brackets.
- Fixed a bug related to the C/C++ Beautifier that caused the unavailability of **Apply to function braces** in the C/C++ Formatting Options dialog.
- Fixed a bug in the C/C++ Beautifier that caused incorrect alignment and indentation of member access specifiers in structs.
- Fixed a bug in the C/C++ and C# Beautifiers that caused incorrect brace-style formatting of namespace keywords.

Context Tagging® Enhancements

- SlickEdit now has the capability to handle the tagging of @identifier modifiers in C code.
- Tagging functionality has been added to differentiate between compilation units and header files when handling the **static** keyword.
- Tagging performance has been improved for C/C++ with large amounts of embedded code (for example, SQL or assembly), and the handling of ASM and EXEC tagging in C/C++ has been improved.
- Fixed a bug that caused SlickEdit to crash when tagging Boost 1.33.1.
- Fixed a performance issue with Context Tagging which produced dramatic improvements in the performance searching for references.
- Fixed a bug correctly handling [classname] :: outside of function definitions in C++.
- Fixed a bug that prevented the Preview tool window from updating on refocus.
- Fixed a bug that caused a Slick-C stack when expanding inherited members in the Symbols tool window.
- Fixed a bug that prevented line numbers from being displayed in the current buffer when using **Show info for symbol under mouse (Tools > Options > File Extension Setup > Context Tagging tab)**.
- Fixed a bug that prevented ALM from working with members of a C++ namespace.
- Fixed bugs in the handling of the **volatile** variable.
- Fixed a bug that prevented tagging of abstract classes in PHP.
- Fixed a bug that caused a Slick-C stack when using **Show info for symbol under mouse** in Ruby.
- Fixed a bug that prevented the Up arrow on the spin box for **Max size of files to tag** from incrementing properly.

GDB Debugger Enhancements

- SlickEdit now uses GDB version 6.6. It can be downloaded from www.slickedit.com/php/gdb.
- For users who can't or do not want to use Cygwin for GDB, you can download and install the MINGW-based version of GDB, **gdb-6.3-2.exe**, from <http://www.mingw.org/download.shtml>. This version of GDB is known to work

SlickEdit® Version History

with SlickEdit and does not depend on **cygwin1.dll**. Use the **Configurations tab** on the Debugger Options dialog (**Debug > Debugger Options** or **debug_props** command) to make it the default native GDB debugger configuration.

Version Control Enhancements

- Fixed a bug that caused the **svn_diff_with_tips** command to fail if the CVS option to always use "-d" was on.
- Fixed a bug that caused **cvs_gui_mfupdate** to occasionally not report an error message in the case of a failure.
- Fixed a bug that caused **cvs_gui_mfupdate** to sometimes miss new directories.

Version 12.0.2

General Enhancements

- This release includes the new Code Annotations feature. Notes:
 - In this version of Code Annotations, the Date Control field type just stores dates as text. Columns of this type are sorted lexicographically, not by date. We plan to introduce true date processing in a subsequent release. In the meantime, you can get the best result for dates by storing them in this format: *yyyy/mm/dd*. For example, 2007/07/04. This will allow dates to sort correctly.
 - Handling annotation type conflicts works differently than documented: SlickEdit detects differences between two annotation types with the same name when an annotation file containing a conflicting type is loaded. The user is asked to rename the type being loaded from the file. The type definition changes names, and all relevant annotations are loaded as the new type.
- Java™ compiler configurations - SlickEdit now supports multiple tag files and compiler configurations for Java projects. Each Java project can now specify the JDK to use for tagging and building. In the Auto Tag dialog (**Tools > Tag Files**, click **Auto Tag**), you can pick from the installed JDKs for tagging. You can also create new Java compiler configurations (click the **Browse** button). In the **Compile/Link tab** of the Project Properties dialog (**Project > Project Properties**), you can select from the **Compiler** drop-down list to configure which JDK to tag and build with for all configurations or any specific configuration. You can also add new Java compiler configurations by clicking the Triple Dot icon beside the **Compiler** drop-down. The current project and configuration determines which JDK to use for tagging and building Java projects. You can have multiple projects in the same workspace using different Java compiler configurations. However, please be aware that only the Java compiler configuration for the active project and active configuration will be used for builds. For example, if you build a project that has dependencies on other projects, the dependent projects will be built using only the current Java compiler configuration for the active project, and will not use the configuration set in the dependent project.
- Added a document mode for Java property files--property names, values, and operators are now color-coded.
- In addition to Javadoc™ and XMLdoc, comments and comment wrapping now support generic doc comments, better readable by tools such as Doxygen. See Help Index keyword "comments" for more information.
- The functionality of **Document > Comment Lines** (**comment** command) has been enhanced--you can now specify the column in which line comments should start. **Document > Uncomment Lines** (**comment_erase** command) now checks for well-formed comments, adheres to settings for spaces versus tabs when making adjustments, and removes characters for nested comments in such a way that if Comment Lines were applied to the result, selected lines would be returned to their previous state.
- Three new commenting configuration variables are available:
 - **def_CW_use_width_for_box** - When the value is set to 1 (the default), **Document > Comment Block** (**box** command) uses Comment Wrap settings to determine the width of the comment, if Comment Wrap is enabled

SlickEdit® Version History

for block comments. The Comment Wrap settings are ignored if they specify a width that is not wide enough for the selected text.

- **def_CW_EnterEndsLineComments** - When the value is set to 1, wrapping for line comments stops on the first Enter keystroke. The default value is 0.
- **def_CW_DoubleEnterEndsLineComments** - When the value is set to 1 (the default), wrapping for line comments stops on the second consecutive Enter keystroke.
- You can now specify the width and height of new editor windows. From the main menu, click **Window > New Window Size**, or use the **new_window_size** command, to display the New Window Size dialog. If **Use defaults** is selected, editor windows are floating when they are first opened, and if you change the size, the size is remembered thereafter for new editor windows. Click **Custom** to specify your own custom width and height for new windows. Type the size in pixels directly into the width and height boxes, or pick from two predefined values: select **Use Current Window** to use the width/height of the current window as the default size for new windows, or select **Use Maximum Available** to have new windows stretched to fill the size of the editor pane.
- The **File tab** on the **File > New** dialog (**new** command) has been enhanced:
 - The **Document Mode** list has been enhanced to include incremental search, and to show the last *N* recently used modes at the top. A new option on the tab lets you configure the number of recent modes to store.
 - A new default document mode has been added, called **Automatic**, which means SlickEdit will automatically determine the document mode based on what you type in the **Filename** box. If the file name has no extension, the new document is created as a plain text file. If the file name has an extension, the new document is created based on looking up the extension in the list of defined document modes. If no document mode lists the extension, you are prompted whether to create the file as plain text or return to the dialog. If the file name contains an extension, but it conflicts with the selected document mode, you are prompted whether or not to continue.
 - **Add to Project** is now checked by default.
 - The `.sh` extension now defaults to the Bourne Shell document mode.
- SlickEdit supports Mac OS® X v10.4. Support for v10.3 has been discontinued.
- Block editing (Block Insert Mode) is now supported in Replace mode as well as Insert mode.
- A new option to exit SlickEdit upon AutoSave is available on the **AutoSave tab** of the File Options dialog (**Tools > Options > File Options**).
- SlickEdit now displays a dialog when using **File > Close All** prompting to save any modified buffers.
- A new option, `SELECT_PROC_NO_COMMENTS`, is available for the **select_proc** command to specify that function header comments should not be part of the selection.
- The column sizes in the debugger Watch tool window are now remembered between debugging sessions.
- Entity tables in `vslick.vlx` and `builtin.html.tagdocs` have been updated to include more consts and keywords.
- Fixed a bug that prevented printing a selection when using Selective Display when the **Visible Lines Only** option on the Print dialog was checked.
- Fixed a bug that failed to honor **Document > Margins** settings.
- Fixed a bug that prevented docking a floating tool window when using Aero™ Glass on Windows Vista™.
- Fixed a bug in the Font dialog that under certain conditions, caused an "invalid font name" error and an error in **Show fixed fonts**.

SlickEdit® Version History

- Fixed a bug that caused the **Preserve case** option on **Search > Replace** to work improperly for UTF-8 files.
- Fixed a bug that caused the **help** command to work improperly under certain conditions.
- Fixed a bug that caused the **Create backup history on save** option (on the **Backup tab** of the File Options dialog) to work improperly.
- Fixed a bug that modified open file timestamps when right-clicking on a file tab and selecting **Save All**.
- Fixed a bug in **Search > Set Bookmark** that prevented naming bookmarks with spaces in the names.
- Fixed a bug that caused Hex mode to exit when a file was auto-reloaded.
- Fixed a bug in Backup History that prevented refreshing on multiple sorts.
- Fixed a bug that caused an incorrect **%f** header when printing.
- Fixed a bug that caused **linewrap_delete_char** in XML files to work improperly.
- Fixed a bug in Auto-Complete that caused incorrect text to be inserted when typing a comment in XML.
- Fixed a bug that prevented GDB debugging from parsing long strings correctly.
- Fixed a bug that caused SlickEdit to hold onto JAR files in the classpath after Java debugging was completed.
- Fixed a bug that caused a continue prompt on long regular expression searches (should fail quietly).
- Fixed a bug that occurred on Windows where excluding full paths when performing a background search did not work properly.
- Fixed a bug in C++ Syntax Expansion where the semicolon after the end brace was not being inserted for templated classes or structs.
- Fixed a bug in Aliases where prompt values containing percent signs (%) were expanded incorrectly.
- Fixed a bug on UNIX regarding locking problems with NFS, that caused background searching to work improperly.

Context Tagging® Enhancements

- A new command is available, **mou_push_tag**, that can be bound to a mouse event, for example, **Ctrl+LButtonDown**, in order to navigate immediately to the symbol under the mouse as if the document were a hypertext. You could then bind the **pop_bookmark** command to **Ctrl+RButtonDown** to use the mouse to jump back. Use the Key Bindings dialog to bind these commands.
- Added functionality for the Preview tool window to pick up **exception** types.
- Fixed a bug in Context Tagging List Members that resulted in the wrong item being inserted when typing too quickly.
- Fixed a bug in Context Tagging List Members for XML files that under certain conditions, inserted two spaces instead of one.
- Fixed a bug in C# List Members that caused missing data.
- Fixed a bug in Context Tagging that prevented jumping to a forward class definition.
- Fixed a bug that prevented the Defs tool window from redrawing properly when switching between large and short files.
- Fixed a scrolling problem in the tree control for the Symbols tool window.
- Fixed a bug in `_ruby_maybeBuildTagFile()`.

SlickEdit® Version History

DIFFzilla® Enhancements

- Fixed a bug that caused **diff_from_cursor** to work improperly when using tiled windows.
- Fixed a bug that prevented the diff state file from being saved when using Multi-file diffs under certain conditions.
- Fixed a bug that prevented typing of numeric keypad keys when using DIFFzilla.

Files Tool Window Enhancements

- Fixed a bug that caused SlickEdit on UNIX to hang if retagging with a file open and the Files tool window docked.
- Fixed a bug in the Files tool window that caused the current buffer background highlight to appear intermittently.
- Fixed a bug in the Files tool window that caused case-sensitivity in the Filter.
- Fixed a bug that prevented prefix matching in the Files tool window.
- Fixed a bug in the Files tool window that caused a Slick-C stack when using it to open a file.

Slick-C® Enhancements

- A generic **loop** statement has been added to Slick-C, like that found in Oberon and D languages. **loop { ... }** is equivalent to **for(;;) { ... }** except that **loop** requires the braces and is more simple and obvious.
- The following class introspection functions have been added to Slick-C: **_typename**, **_instanceof**, **_construct**, **_fieldindex**, **_fieldname**, **_getfield**, **_setfield**, **_findmethod**, and **_callmethod**.
- Support for the **instanceof** operator (found in Java) has been added to Slick-C. The Slick-C operator determines if a variable containing a class instance is of a type that matches or derives from the given type or interface name. It can be used in two ways: **x instanceof MYCLASS**, or **x instanceof "MYCLASS"**. "MYCLASS" does not need to be a constant string, and x may be a typeless container variable.
- Functionality has been added for Slick-C **_length()** to work on strings, hash tables, classes, and structs.

Version Control Enhancements

- Fixed a bug in CVS setup that prevented users from specifying their own CVS-based executable.
- Fixed a bug in Subversion support that caused incorrect version numbering when diffing two versions.
- Fixed a bug that caused the **Commit** button to be unavailable when using **svn_gui_mfupdate**.
- Fixed a bug in Subversion that prevented the **Add** and **Remove** commands (on the Project tool window's right-click context menu) from working.
- Fixed bugs in CVS and Subversion support that prevented adding files with spaces in the paths.

Vim Emulation Enhancements

- **Visual Mode Enhancements**
 - The following inner object selection commands have been added: **iw** (select word), **iW** (select WORD), **is** (select sentence), **ip** (select paragraph), **if** (select "{" defined block...also **iB** and **i}**), **if** (select "(" defined block... also **ib** and **i)**)
 - The following navigation commands have been modified to work in visual mode: **w**, **b**, **W**, **B**, **e**, **E**, **\$**, **G**, **gm**, **gP**, **gp**, **ge**, **gE**, **%**, **{**, **}**, **(**, **)**, **[[**, **]]**
 - The following Ex commands now work on selections: **:copy** (or **:t**), **:delete**, **:substitute** (or **:s**), **:join**, **:list**, **:move**, **:number**, **:print**, **:put**, **:read**, **:yank**, **sg**, **z**, **!**, **=**, **>**, and **<**

SlickEdit® Version History

- > and < now work in visual mode to indent selections.
- Block Insert Mode has now been integrated into the Vim emulation. **c** and **C** now enter block insert mode after the cut is performed (when in visual block mode).
- **Additional Commands**
 - **.** (Go to last edited line.)
 - ***** and **#** (Search forward or backward, respectively, for the current word.)
 - **:bufdo** (Allows you to enter a command which will be performed on all open buffers. For example, **:bufdo 1,\$s/help/HELP/g** will perform the specified substitution command on all open buffers.)
 - **[{, [(** (Move backward to enclosing "{" or "(", respectively.)
 - **],)** (Move forward to enclosing ")" or ")", respectively.)
 - The following split window key bindings have been added: **Ctrl+w,w** (next window), **Ctrl+w,W** (wrapping window above), **Ctrl+w,v** (vertical window split), **Ctrl+w,q** (kill window).
- **Changes to Current Commands**
 - **showmode** has been turned on by default.
 - **yy**, **dd**, **p**, and **u** will now display how many lines yanked, cut, or pasted in the status bar.
 - A new config variable, **def_vi_always_highlight_matches**, has been added which will make **/** and **?** always highlight all occurrences of the found text in the buffer, when set to 1. Support for the ***#** option has also been added, which will manually accomplish the same thing. For example, **/foo/*#** will highlight all instances of **foo**.
 - **f** and **F** will now behave as they do in actual Vim, where if on a "q", **fq** will move to the next "q".
 - The config variable, **def_vim_change_cursor**, has been added (and DEFAULTED TO ON), which will show an underscore cursor when in command mode, and the normal, vertical cursor when in insert mode. Set to 0 if you wish to have the same cursor shape in command mode as in insert mode (pre 12.0.2). Note: There is a known bug on UNIX and Linux systems where the underscore cursor is appearing longer than it should when editing UTF-8 files.
- **Miscellaneous Changes**
 - Fixed a bug in Vim emulation where **__ex_number** was not listing the numbers of lines.

Workspaces & Projects Enhancements

- The **Project tab** on the New dialog (**Project > New** or **workspace_new** command) has been enhanced:
 - The **Project types** list is now a tree view, listing each primary language type with subtypes underneath. It also shows the last *N* recently used types at the top. A new option on the tab lets you configure the number of recent types to store.
 - A new option, **Create project directory from project name**, will create a new subdirectory with that name at the location specified in the **Location** box.
 - A read-only area on the tab shows the full path to the new project. When you check the option **Create project directory from project name**, this area shows the concatenation of the **Location** and **Project name** fields.
 - When creating a new project, if you enter the name of an existing project, you are now prompted to add the project to the current workspace.
 - Selecting one of the Microsoft placeholder project types now immediately displays a message containing information about creating the workspace/solution in Visual Studio®.

SlickEdit® Version History

- Fixed a bug that preventing setting of pre- and post-build commands on the **Build tab** of the Project Properties dialog.
- Fixed a bug that caused a Slick-C stack when using **Build automatically on save** (on the Java Options dialog) for a Java project.
- A **Create Directory** button has been added to the Choose Directory dialog used when creating new projects.
- SlickEdit now prompts before overwriting existing `.vproj` and `.vprojw` files for associated projects.
- Fixed a bug in the Project Properties dialog Tools tab that prevented the **Command line** from being cleared when adding a new command.

Version 12.0.3

General Enhancements

- SCIM support under Linux is now working. This allows you to type accent characters and use the input method editor for typing Japanese/Chinese/etc. characters. By default, the SCIM support is turned off. To turn it on, set the `VSLICK_XIM` environment variable to **scim**. You can place this setting in your `vslick.ini`. Unfortunately, when you turn on the SCIM support, the **Ctrl+Space** key sequence will likely be taken over by SCIM so you will no longer be able to use **Ctrl+Space** for symbol completion or alias expansion. You may be able to work around this problem by removing the **.UTF-8** suffix from your `LANG` environment variable. For SlickEdit 2008, we expect to have the SCIM support turned on by default and also to provide an option for configuring **Ctrl+Alt+Space** to display the SCIM input method editor instead of **Ctrl+Space**.
- Loaded hot fixes are now listed in **Help > About SlickEdit** in the Program Information pane.
- A **Grayscale** color scheme has been added to the default color schemes (**Tools > Options > Color**). This color scheme is printable and works well for cutting and pasting code (when the RTF Clipboard Format option is on [see Selections tab on the General Options dialog]) into documents for color-coded code samples. This is useful for documents that will be distributed primarily in non-color print form.
- Fixed an issue where **on_change** events could be issued while repopulating a tree.
- Fixed an issue where Organize Imports would delete static import in the `.*` case. Added support for Add Import/Organize Imports on Java™ annotations.
- Fixed an issue in **filter_command** (**Edit > Other > Filter Selection**) to allow for redirection pipe (`|`) in command.
- Fixed an issue that caused a silent crash when using Add Tree (**Project > Project Properties > Files tab**).
- Fixed a bug in the `bbcompile` button (View > Toolbars > Customize > Project > Categories tab > Project category).
- Fixed a bug that caused a GDI resource leak on Windows when the Preview window was open during redraws and when tooltips appeared during mouseovers on the status line.
- Fixed an issue with Preview when selecting the Files tool window.
- Fixed an issue where adding a file to a project from the Open tool window did not update the Files tool window.
- Fixed problem where Defs tool window would highlight the wrong item.
- Fixed an issue with Selective Display Expand/Collapse Block where the block was collapsed too far.
- Fixed an issue in Auto-Complete that was using an incorrect lexer for embedded keywords.
- Fixed an issue with UTF-8 file encoding not automatically recognized when set to **Auto Unicode2**.
- Fixed batch file issues with REM line comment color coding.

SlickEdit® Version History

- Fixed an issue where line comments were not honoring **Only if first non-blank character in line** setting for Color Coding.
- Fixed a bug that caused a Slick-C® stack when opening the XML/HTML Formatting Options dialog from the config menu with no buffers open.

Code Annotations Enhancements

- A new command has been added, **activate_annotations**, to show the Code Annotations tool window.
- Fields added to annotation definitions are now automatically added to any pre-existing instances as well.
- Pressing **Esc** while editing an annotation now acts like the **Cancel** button, canceling any changes and closing the dialog.
- When saving a source file to a new name, you can now choose to copy its annotations over as well.
- Deleting or modifying an annotation no longer affects the filters in the Code Annotations tool window.
- New SCA (annotation) files are now created empty.
- Note that modified source files must be saved before allowing annotations, to avoid cases where annotations could be made on lines that don't get saved.
- Fixed an issue that caused a possible Slick-C stack when creating annotations the first time v12.0.2.0 was run, but before saving the configuration.
- Fixed an issue that caused a possible Slick-C stack when creating annotations with empty list controls.

Commenting Enhancements

- Fixed an issue where line comment wrap failed when immediately followed by a block comment.
- Fixed an issue where line comment wrapping was confused by an adjacent block comment.
- Fixed an issue where **comment_erase** caused extra spaces to be left in the code. Improved reindent support for **comment_erase** in additional languages.
- Fixed an issue where capitalization was not respected for comment strings.

Context Tagging Enhancements

- Added begin/end pair matching configuration for Ruby:
(if),(while),(unless),(class),(do),(begin),(case),(module),(for),(until),(def)|(end)
- Improved algorithm for selecting a unique match in List Members, allowing it to find even a unique case-insensitive prefix match.
- Improved qualified tag name expression searching in the Find Symbol tool window.
- Fixed a possible Slick-C stack in Auto Tag dialog.
- Fixed an issue with Java tag file names and contents being out of sync.
- Fixed a tree scrolling issue in the Find Symbol tool window and a bug that caused a Slick-C stack when undocking the tool window.
- Fixed a bug that caused incorrect tagging using namespace.
- Fixed a C/C++ preprocessing bug involving **#if** and **#elif**.

SlickEdit® Version History

Refactoring Enhancements

- Fixed a Slick-C stack that occurred in Pull Up to Superclass refactoring.
- Fixed a bug in Pull Up to Superclass that caused a "No files were changed" message when there were files that had been changed.

Version Control Enhancements

- Fixed an issue that caused a Slick-C stack when committing a new directory and file under the directory in Subversion.
- Fixed an issue regarding Perforce version control support, where user information in the SCC local path and auxiliary path did not match the current user name because they were checked in. The SCC local path and auxiliary path are now kept in `.vpwhist`. Do not check in this file.
- Fixed a bug in SCC version control support that resulted in a crash on exit because DLLs were not always freed.
- In SCC version control, if **Prompt for files** is off and **Auto check-out on edit** is on (**Tools > Version Control > Setup**), when you type in a read-only file, SlickEdit will now auto check out the file and not prompt.

Vim Emulation Enhancements

- Fixed an issue that prevented some Vim commands from being functional in read-only mode.
- Fixed an issue with case sensitivity in Vim `:s` command.
- Fixed bug in Vim where it was possible to silently save the active selection over the current file using `:w`, without specifying `!`.

SlickEdit v11

SlickEdit v11 was shipped Spring, 2006.

Version 11.0.0

- **Code Templates** - Define templates for commonly used code, like a standard class definition or design patterns. You can create templates for whole files or multiple files. Add a template item to your current project by choosing **Project > Add New Item** from the main menu. You are prompted for values to substitute in the new instance as needed.
- **Comment Wrapping** - Comments can now be set to automatically wrap to the next line as you type. Wrapping options can be configured for C, C++, C#, Java, and Slick-C files.
- **Auto-Generation of Javadoc™ and XMLdoc** - Automatically creates a skeleton comment when you type the begin comment characters, like `/**` for Java.
- **Enhanced Search and Replace** - Search and replace operations have been greatly enhanced and combined into one improved, dockable tool window. Enhancements include:
 - Save your frequently used search and replace operations for future use.
 - Use wildcard characters to specify a search pattern.
 - Search incrementally via the tool window.
 - Search for hidden text when selective display is enabled.

SlickEdit® Version History

- Highlight matches and bookmark matches in the editor window.
- Preview the differences before committing changes to a file.
- **Regex Evaluator** - The Regex Evaluator provides the capability to interactively create and test regular expressions.
- **Vim emulation** - The vi emulation has been extended to include Vim functionality. The emulation has been renamed as Vim emulation.
- **Word Completion on file names** - This new feature assists you when typing a command or an argument to a command. If you press the spacebar when typing the beginning of a command or argument, this feature automatically inserts as much of the argument as possible. Both the command line and many dialog input fields support completion.
- **Specify Tag Jump Order** - For C/C++, specify whether to jump to a symbol's declaration or definition when conducting tag navigation. Select **Tools > Options > File Extension Setup**, then choose the **Tagging tab**, and enable one of the two **Go to Definition** options.
- **Additional Quick Refactorings** - Quick Refactoring provides operations based on the Context Tagging engine rather than parsing and type analysis, like our C++ Refactorings. These operations are available on C++, C#, Java, and Slick-C. Quick Refactoring is generally faster and less stringent than C++ Refactoring.
- **Enhanced Auto-Completions** - Auto-Complete offers suggestions for how syntax, keywords, symbols, and lines of code may be completed by the editor. It works by looking at the word prefix under the cursor and using several different queries to find and suggest completion options. Each of these types of suggestions can be individually enabled or disabled.
- **Enhanced Bookmarks** - Bookmarks can be set and accessed through a new, dockable tool window. Additional bookmark configuration options have also been added.
- **Font support for Linux using Xft** - SlickEdit now supports Xft fonts for Linux.
- **Additional enhancements** - Along with the new features listed above, we have made many other enhancements to SlickEdit for version 11. The following are particularly noteworthy:
 - The References tool window maintains all of its previous functionality, but now you can turn off the preview pane by clicking on the plus/minus icon. The Symbol view can now be used as a preview for the References view, whether or not the References preview pane is visible. If you like using the Symbol view as a preview, you may prefer to dock the References view on the left so it does not obscure the Symbol view.
 - The Context Tagging group of options has been moved from the **Tools > Options > General** dialog to the **Tools > Options > File Extension Setup** dialog.
 - Enhanced foreign keyboard support—Foreign keyboards (UNIX only): Users no longer have to invoke SlickEdit with the “-sua” switch to enable the AltGr (right-Alt) key. SlickEdit will now automatically recognize when the right-Alt is to be used as AltGr.
 - Support for the Ruby language has been added.
 - Added support for J#.
 - The Mac OS X version is shipped as a Universal Binary that will run on both PowerPC and Intel® Macintosh® computers.
 - Support for MSDN help has been added. Select **Help > Configure F1 MSDN Help**.
 - “Build Automatically on Save” will launch a build when a file is saved. To enable this, select **Build > Build Automatically on Save**.

SlickEdit® Version History

- Backup History has been converted to a tool window, allowing it to be docked. It displays the history for the active file.
- The Color Settings dialog now contains a button to synchronize the background color of related elements. Select **Tools > Options > Colors**.
- For Microsoft Visual Studio solutions or files, you can launch Visual Studio to edit a file or solution using the commands **vstudio-edit-file** and **vstudio-open-solution-file**.

Version 11.0.1

- ActionScript language support added.
- Macromedia Flash project support added.
- Python language improvements for Context Tagging, code navigation, and more.

Version 11.0.2

- New menu item to install Hot Fixes.
- Enhanced GDB debugging performance.
- Improved tagging of Boost 1.31.
- Many other bug fixes and enhancements.

SlickEdit v10

SlickEdit v10 was shipped Spring, 2005.

Version 10.0.0

- **GUI enhancements** – This release introduces a new appearance in the graphical user interface including:
 - **Icons** - The addition of new icons.
 - **Dockable tool windows** – View windows can be drag-dropped into tabbed groups within the editor, or dragged off to the side to make them free-floating. Docking options are accessed from the right-click context menu.
 - **Tab groups** – The editor is now divided into sections where tool windows can be docked.
 - **Auto Hide** – An auto-hide option has been added for tool windows so they can be automatically hidden when not in use. To access this option, use the tool window's right-click context menu or toggle the Pin icon.
- **Subversion support** – SlickEdit's Subversion support provides easy convenient access to information about the files with which you are working, and also a GUI checkout dialog. To get started, go to **Tools > Version Control > Setup**, and set the "Command line system" to "Subversion". After you enable these settings, you can diff any file with the current version on its branch, view the history of the file, and update or commit the file. You can also select **Tools > Version Control > Compare Workspace with Subversion** to compare your local workspace with the files in the repository.
- **Contact Product Support** - Contacting SlickEdit product support is now easier than ever. The SlickEdit support team can be contacted by clicking the **Contact product support** link on the Help menu.

SlickEdit® Version History

- **View Maintenance expiration date** - Maintenance expiration dates are viewed easily using this feature. To view the maintenance expiration date, click the **Check Maintenance** link on the Help menu.
- **Unattended Installation** - Use unattended installation to install multiple copies of SlickEdit on many computers at one time. This feature is useful if you are installing SlickEdit in an environment with many users.
- **Mouse wheel spin events** - Mouse wheel spin events can be bound to SlickEdit commands. Using this feature you can bind “Ctrl + c_wheel_up” to perform a page-up for example.
- **Quick Rename** - Quick Rename uses the tagging engine to rename a symbol under the cursor or any symbol selected in the class browser or symbol definitions toolbar. This operation works for all tagged languages. It is faster than the rename provided by C++ Refactoring, but it is less stringent.
- **Debugger features** – This release provides new features for debugging:
 - **Mixed-mode view in debugger** - When debugging, you can view your source code with the disassembled code displayed between each line of source. In this mode you can step execution at the assembly language level for greater control over debugging. The buffer is changed to read-only so that the SlickEdit product can maintain synchronization between source and disassembled code.
 - **Multiple session debugging** - Multiple session debugging provides the ability to start more than one debugging session within a single instance of SlickEdit. You can have one session debugging using GDB, and one using Java at the same time.
 - **Hex, Octal, and Binary view** - This feature is available for viewing numbers in multiple bases.
 - **Debugger watchpoints** - With this feature you can define an expression that will cause the debugger to break when the expression evaluates to true. For example, you can break a loop when the value of a counter reaches a particular value. This feature is scheduled for release in SlickEdit v10.0.1.
- **C/C++/C# features** – This release provides new features for working with the C/C++ and C# languages:
 - **C++ Refactorings** – The following new refactorings are available:
 - **Modify Parameter List** – This refactoring enables you to add, delete and reorder parameters for a selected function.
 - **Extract Super Class** - Use this refactoring to break a large class into a better abstraction by moving some responsibilities into a new class/interface, the extracted class becomes the super class of the original class.
 - **Extract Class** - This refactoring adds the ability to break a large class into a better abstraction by moving some responsibilities into a new class/interface. This refactoring creates new files.
 - **Pull Up to Super Class** - Use this refactoring to automate moving members from a selected class to one of its directly inherited base classes. Class members are pulled up to the super class one level at a time.
 - **Push Down to Derived Class** - Use push down refactoring to move class members from a super class to one of its selected subclasses. This refactoring provides the ability to select from a list of classes that directly inherit from the superclass.
 - **Visual Studio key emulation** – This release of SlickEdit provides support for key-bindings that emulate Visual Studio .NET. These key-bindings are not the same as the ones used in Visual C++, though there might be some overlap.
 - **Visual Studio project support** - This release of SlickEdit provides additional Visual Studio project support, including bug fixes and expanded support to include more project types.
 - **Microsoft Visual C++ toolkit support** - You can now create projects that use the Microsoft Visual C++ Toolkit and Microsoft .NET Framework Software Development Kit.

SlickEdit® Version History

- **Java features** - This release of SlickEdit contains additional support for new Java features:
 - **Java Live Errors** - Java Live Errors feature flags syntax and compilation errors as you edit your code.
 - **Organize Imports** - Organize imports automates the management of import statements in Java files. This feature minimizes the amount of time that it takes to compile code by only importing the classes that are used. Existing import statements are also sorted in a readable format and are more consistent between different java classes in the same project.
 - **JavaScript™ support** - SlickEdit now supports JavaScript 1.4.
 - **JSP™** - Updated the code documentation, tagging, and color coding to support JSP 2.0. Tagging is available when using usebean attributes. The available beans are added to tagging so that they can be selected during code completion. Tagging now works with taglib attributes. Any symbols in the tld specified by the taglib attribute are added to tagging with the taglib prefix so that they can be selected during code completion. Parsing now looks for the DOCTYPE attribute in XML files. If the XML file has a DOCTYPE of taglib then the XML file is interpreted as a tld file instead of as an XML file.
 - **Hot-Swap Debugger** - The Hot-Swap Debugger enables you to edit a file during a debugging session, reload it, and then, continue to debug.
 - **JUnit support** - JUnit tests can be run from within SlickEdit. The results can be viewed and code that fails the testing can be easily reconciled. The JUnit menus are visible only if the project has unit testing enabled. They are enabled when the context menus are brought up for nodes in the active project, and only if a single item is selected.
 - **J2ME** - SlickEdit provides a new project type for working with the Java 2 Micro Edition used for embedded Java development. This feature is scheduled to be released in SlickEdit v10.0.1.

Version 10.0.1

- **J2ME support** – A new project type has been added to enable you to create, build, run, and debug J2ME projects in SlickEdit. Select **Project > New** and choose "Java - J2ME Midlet" from the project type list. This project depends on having a J2ME-compliant tool chain already installed. If you do not have one, you can download the J2ME Wireless Toolkit from Sun Microsystems to explore this capability (see <http://java.sun.com/j2me> for more information).
- **Debugger watchpoints** - The capability has been added to break execution when a specified variable is modified or read. To set a watchpoint, select a symbol, right-click, and choose "Set Watchpoint". Specify when to break in the breakpoint properties dialog. The capability to evaluate an expression and break execution is provided on normal breakpoints, not on watchpoints. Conditions can be specified in the breakpoint properties dialog.
- **Microsoft Visual Studio 2005 project support** - SlickEdit provides the capability to open MS Visual Studio solution files. In v10.0.1 we updated this to be compatible with the beta 2 release of Visual Studio 2005, also known as "Whidbey".
- **Debug Other Executable** - Allows you to start a GDB debugging session to debug an executable independent from your current project.
- **FTP and SFTP enhancements** – The following enhancements were made for FTP/SFTP:
 - **SFTP Authentication support** - Added support for public-key, host-based authentication. Note: See the OpenSSH client documentation for instructions on setting up and exchanging keys with an ssh server if this is the first time you are using these authentication types.
 - **SFTP/FTP Auto reconnect** - Added support for auto reconnect after a connection dies. Note: The reconnect is not attempted until an operation is performed.

SlickEdit® Version History

- SFTP/FTP speed enhancements.
- SFTP/FTP changing directory now easier.

Version 10.0.2

SlickEdit v10.0.2 contains only bug fixes and minor enhancements.

Version 10.0.3

- **Assembly files** – Added support for #if...#endif matching in Assembly files.
- **Java** – Improved performance finding references and added ability to jump to file/line of Java stack trace in Build window.
- **JavaScript** – Added support for color coding JavaScript regular expressions.
- **Makefiles** – Added support for automatically setting Makefile mode for extensionless makefiles (e.g. "Makefile").
- **C# project support** - Added support for Visual Studio .NET Link files which are common in C# projects.

SlickEdit v9

Released under the name “Visual SlickEdit,” this version was shipped Spring, 2004.

Version 9.0.0

Language/Environment Features

- **C++ Refactoring** - Visual SlickEdit now includes C++ refactoring, an industry first.
 - **Rename** - Provides the ability to rename variables, methods, and classes. Updates the rest of the code to use the changed name.
 - **Extract Method** - After selecting a set of lines, Extract Method creates a new method with the selected lines as the body. Any undeclared variables are discovered and created as parameters to the new method.
 - **Encapsulate Field** - Generates getter and setter methods for the specified variable and makes that variable private.
 - **Move Method** - Moves a method from one class to another and updates references accordingly.
 - **Move Static Field** - Moves a static data member from one class to another and updates references accordingly.
 - **Convert Static Method to Instance Method** - Changes a static method to an instance method and updates any references to change how the method is accessed.
 - **Convert Global to Static Field** - Moves globally declared variables into a static field in a class. Updates references to refer to the new static variable.
 - **Convert Local to Field** - Moves a local variable from the body of a method to be a class member. References to the local variable are replaced with references to the new data member.
 - **Replace Literal with Declared Constant** - Replaces the selected literal with a constant, replacing use of the literal with the new constant.

SlickEdit® Version History

- **Create Standard Methods** - Creates an assignment operator, copy constructor, default constructor and destructor for the selected class.
- **Java GUI Builder for Swing and AWT** - The new Java GUI builder in Visual SlickEdit v9 enables developers to construct Java GUIs in Swing or AWT while also allowing simultaneous code editing. The Java GUI builder provides a toolbar of GUI widgets and a design pane for assembling the GUI. A property window enables setting the properties for each element, such as specifying layout managers. The GUI builder generates code to implement the screen layout and updates the layout mode when the code is edited.
- **CLR (Common Language Runtime) Debugger for .NET applications** - The CLR debugger in Visual SlickEdit allows developers to debug any .NET program compiled for the CLR environment from the familiar Visual SlickEdit debugger window. By including this capability in Visual SlickEdit, developers can now edit, build, run and debug .NET CLR applications using Visual SlickEdit. This feature is supported for C, C++, C#, and Visual Basic .NET.
- **Code Completion for CLR applications** - Code completion for CLR applications allows C# developers to view function help and parameter information for CLR library functions from within Visual SlickEdit.
- **Surround With language-specific structures** - The Surround With feature provides the capability to select a code block and surround it with language-specific structures. For example, in C/C++ and Java, the developer can now select a number of lines of code and surround them with an **if**, **while**, **for**, or **do**. Developers can modify these structures through the existing Visual SlickEdit alias capability. Surround With supports the following languages and constructs:
 - **C/C++** - case, catch, default, do...while, finally, for, if, if...else, ifdef, ifndef, include once - #ifndef (tag) #define (tag) text #endif // (tag), struct, switch, try, type struct - typedef struct, type union - typedef union, while
 - **Java/JavaScript** - case, catch, default, do...while, finally, for, if, if else, switch, try, while
 - **C#** - case, catch, default, do...while, finally, for, if, if...else, ifdef, ifndef, include once - #ifndef (tag) #define (tag) text #endif // (tag), struct, switch, try, type struct - typedef struct, type union - typedef union, while, foreach
 - **HTML** - anchor, blink, bold, definition list, definition list definition, definition list term, emphasis, font, general tag (prompts for tag name), heading 1, heading 2, heading 3, heading 4, heading 5, heading 6, italic, link, list item, ordered list, ordered list with li*, paragraph, preformatted*, style, table, table data, table row, underline, unordered list, unordered list with li*
 - **XML** - general tag (prompt for tag name)
- **Statement Level Tagging** - Statement level tagging in Visual SlickEdit provides a detailed view of items in the Defs Tab (formerly the Procs Tab) for C/C++, Java, VBScript, and Visual Basic .NET. Along with definitions, developers can now view constructs like **if**, **while**, and **for** statements.
- **Improved VBScript tagging recognition** - VBScript tagging in Visual SlickEdit has been improved and now provides more accurate tagging recognition.
- **Ada Beautifier** - Ada language support is enhanced in Visual SlickEdit v9 to provide the following beautifications:
 - **Indentation** - Easily change indentation and whether to indent with spaces or tabs.
 - **Statements or declarations** - Change reserved word case and number of statements or declarations per line.
 - **Horizontal spacing** - Change padding after binary operators, semicolons, commas and parentheses.
 - **Vertical alignment** - Align on colon or in/out. Specify a number of blank lines after different language constructs.

SlickEdit® Version History

Cross-Language Features

- **Dual monitor support** - Visual SlickEdit now enables developers to take advantage of the extra real estate provided by dual monitors. Visual SlickEdit remembers window locations and, once an item has been moved to a particular monitor, will display that item on the correct monitor in future sessions.
- **Full-Screen Editing mode** - The new full screen editing mode in Visual SlickEdit allows developers to expand the edit window to the full size of the screen to provide additional coding real estate. Full screen editing is particularly useful when the developer is editing large files but does not need Visual SlickEdit toolbars and windows at that time.
- **CodeWright® emulation** - Visual SlickEdit now includes CodeWright emulation, which provides CodeWright users with the ability to keep their familiar key bindings and easily migrate to Visual SlickEdit.
- **Softwrap** - The new Softwrap feature Visual SlickEdit enables the developer to easily view long lines of code without scrolling. The line is wrapped as though a carriage return was inserted, however, the file itself is not modified.
- **Background Search** - Visual SlickEdit now includes a background search capability, which allows the developer to continue editing while the search is being performed. All of the powerful search features in Visual SlickEdit, such as color or regular expression search, are available in background searches.
- **Limit Tag Display** - Large files, particularly XML files, may have a very large number of tags. The limit tag display feature in Visual SlickEdit limits the number of tags displayed so that the Defs Tab (formerly called the Procs Tab) is not overly large.
- **Backup History** - Visual SlickEdit now provides a backup history, which is a powerful mechanism to view and restore previously saved versions of a file. By using backup history, developers can easily revert to a previous version that was not checked into version control.
- **Multiple Last-Recorded Macro** - Visual SlickEdit now includes a new command called **execute_last_macro_key()** to allow developers to quickly bind recorded macros to a set of keys, such as Ctl+0 through Ctl+9. To use this feature, first, bind **execute_last_macro_key** to the keys that you wish to be able to run recorded macros on. Ctl+0...Ctl+9 are typical. Then, when recording a macro, press the key that you wish to bind it to when you stop recording. For example, when done recording press Ctl+3 to bind the newly recorded macro to that key. Subsequent key presses of this key will run that macro.
- **DIFFzilla® retains settings** - When run from the command line, the DIFFzilla differencing system now retains settings for screen size and location as it has done from within Visual SlickEdit.
- **HTML Help** - Visual SlickEdit is now includes a new help system that displays help using HTML, which allows easier navigation. To ensure that all Visual SlickEdit users can use the new Help, the Mozilla browser is provided, if a browser is needed, for use on all supported platforms.
- **Product tutorial** - Visual SlickEdit now includes an interactive product tutorial, which can be launched from Help. This new tutorial enables developers who are unfamiliar with many of the capabilities of Visual SlickEdit to quickly get oriented to many of the commonly used features of the product.
- **Update Manager** - The new Visual SlickEdit Update Manager automatically notifies the developer when new patches or versions of the product are available. When a new patch or version becomes available, the Update Manager guides the user to the Visual SlickEdit download site to view what is in the release and to easily download and install the new patch or release.

Version 9.0.1, 9.0.2, 9.0.3

Visual SlickEdit® v9.0.1, v9.0.2, and v9.0.3 contain only bug fixes and minor enhancements.

SlickEdit® Version History

Version 9.0.4

- **C++ Refactoring** – C++ Refactoring is now enabled for Visual SlickEdit on Mac OS X. Many C++ Refactoring performance improvements have been added.
- **C++ STL and ANSI C++ support** – Many improvements added, including support for C++ using statement and namespace aliases.
- **Java 5.0 (1.5) support** – Many improvements added including support for enumerated types, generics/template support, support for varargs, and more.
- **Statement Tagging enhancements** – New bitmaps for different types of statements, added filtering by statement type in Defs tab.
- **Improved Context Tagging performance.**
- **Improved handling of nested namespaces in class browser.**