Your Short Cut to Knowledge

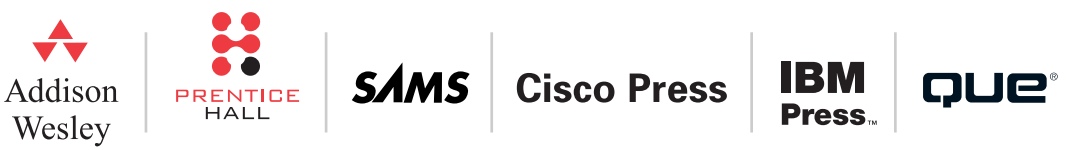
The following is an excerpt from a Short Cut published by one of the Pearson Education imprints.

Short Cuts are short, concise, PDF documents designed specifically for busy technical professionals like you.

We've provided this excerpt to help you review the product before you purchase. Please note, the hyperlinks contained within this excerpt have been deactivated.

Tap into learning—NOW!

Visit **www.informit.com/shortcuts** for a complete list of Short Cuts.



Source Code Editing

Tip 1: Outsmart Search & Replace (1)

The Search & Replace dialog (Ctrl-H) supports regular expressions, making it an extremely powerful tool when searching and replacing text.

Regular expressions use a special syntax that can express a wider range of search patterns than plain strings can. For a complete list of regular expressions and how to use them, see `http://java.sun.com/javase/6/docs/api/java/util/regex/Pattern.html`.

Here are a few examples of symbols that are commonly used in search terms.

- `\n` stands for a newline character.
- `\w` stands for any alphanumeric character (as opposed to whitespace or punctuation).
- `^` stands for the beginning of a line and $ for the end of a line.
- `[a-z]` stands for any character from *a* to *z*.
- `[^a-z]` stands for a complement set (any character except *a* to *z*).
- `+` stands for one or more and * stands for zero or more.

In the replace term, you can use normal characters as well as back-reference symbols such as `$0` and `$1`. Back references allow you to modify parts of a searched string while keeping other parts untouched. For more information on back references, see the following URL:

www.regularexpressions.info/brackets.html

Every time you surround a substring in the search term with parentheses, it is copied into something similar to a numbered clipboard. You can then paste the substrings into the replace term by writing `$1` for the first one, `$2` for the second one, and so on.

Look at the following example where we use a back reference to remove empty lines from a file.

Searching for two or more newline characters will find all lines that are followed by one or more empty lines. Symbols such as `\n` cannot be used in the replace term, though. So we use a typical *regex* trick: We capture one of the newline characters in parentheses (`\n`), and use the back reference `$1` to smuggle it into the replace term. Figure 1 shows the regular expression in the Replace dialog.

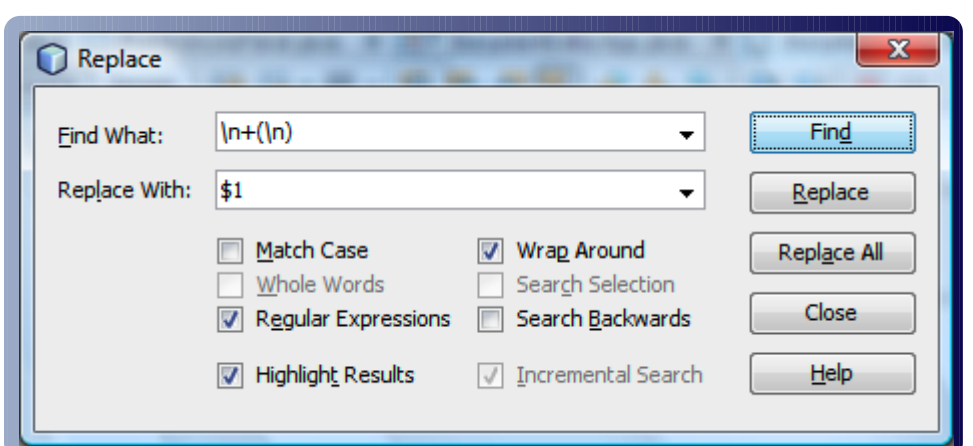


FIGURE 1: Replacing two or more newline characters with one

Try it! Click the Find button to see how the pattern is recognized and selected. Then click the Replace button to see how several newlines are replaced by one. Click Replace All to remove all empty lines from the file in one go.

If you are intrigued, learn more about the power of regular expressions in Jan Goyvaerts's Regular Expression Quick Start at `www.regular-expressions.info/quickstart.html`, and on `java.sun.com/javase/6/docs/api/java/util/regex/Pattern.html`.
This nifty editor feature saves you a lot of time that you would spend making all these changes manually.

Tip Source

`http://blogs.sun.com/roumen/entry/another_regexp_trick_in_netbeans`
`http://blogs.sun.com/roumen/entry/regexp_search_in_netbeans`
`http://blogs.sun.com/seapegasus/entry/100_ide_hacks`

Tip 2: Outsmart Search & Replace (2)

Here is a second example where we use a back reference to keep one part of the string and modify another. Say you have a long row of different variable declarations—but now you need all public integers to be private and initialized to zero.

```
...
public int w;
public int x;
public String y;
public int z;
```

You want all other variable declarations and integers in other contexts to remain unchanged. The following code sample shows what you want to achieve:

```
...
private int w = 0;
private int x = 0;
public String y;
private int z = 0;
```

Obviously, you cannot use basic search and replace here due to the integers' unique variable names that you want to keep as they are. But you know that variable names are composed of alphanumeric characters, and the regular expression for *a string of alphanumeric characters* is `\w+`.

This means you capture the variable names in the search term with the parenthesized (`\w+`) symbol. Then you make your substitutions in the string, and paste the variable names back in, by calling the back-referencing `$1` symbol. Figure 2 shows what the regular expressions look like in the Replace dialog.

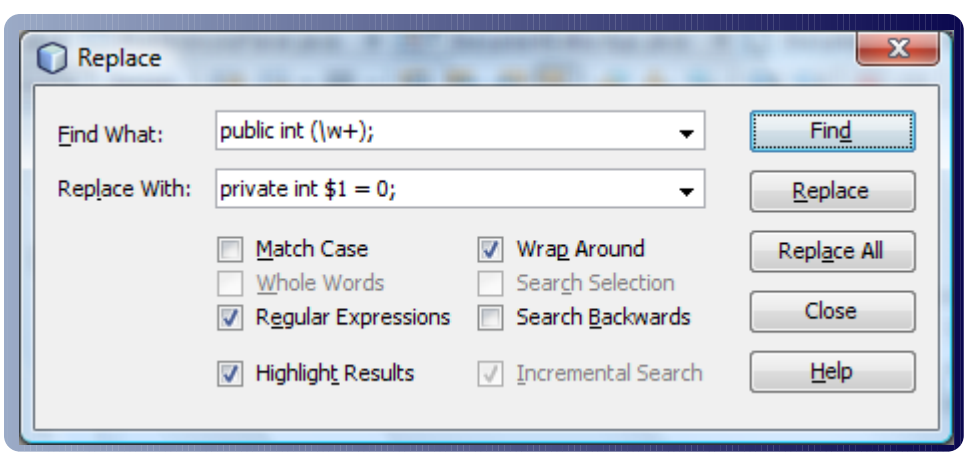


FIGURE 2: The Replace window

Remember that you can use `$2`, `$3`, and so on for the additional parenthesized expressions in the same regular expression.

Tip Source

```
http://blogs.sun.com/roumen/entry/another_regexp_trick_in_netbeans
http://blogs.sun.com/roumen/entry/regexp_search_in_netbeans
http://blogs.sun.com/seapegasus/entry/100_ide_hacks
```

Tip 3: Bookmark Lines in the Editor

If you need to quickly cycle back and forth between certain lines within a file, flag the lines with bookmarks.

- Press Ctrl-Shift-M (Mac: Command-Shift-M) to bookmark the current line. You can tell a line is bookmarked by the blue bookmark icon in the left sidebar.
- Press Ctrl-Shift-Period (Mac: Command-Shift-Period) to jump to the next bookmarked line.
- Press Ctrl-Shift-Comma (Mac: Command-Shift-Comma) to jump to the previous bookmarked line.
- To remove a bookmark, place the caret into the line and press Ctrl-Shift-M (Mac: Command-Shift-M) again.

If you forget the keyboard shortcut, you can always right-click the left side of the editor and use the Bookmark menu.

Also have a look at the error stripe on the right side, as shown in Figure 3. It displays a dark gray mark for each bookmark. Click a mark to jump to the bookmark.

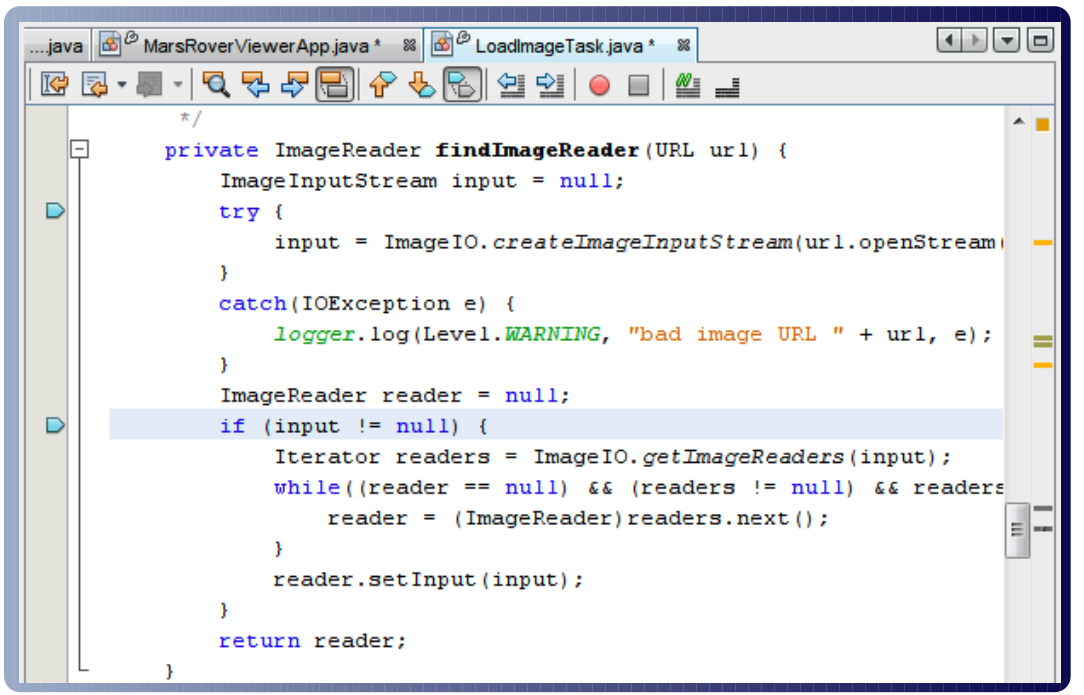


FIGURE 3:
Blue editor bookmarks (left) and corresponding dark gray error stripe marks (right)

Tip Source

`http://netbeans.dzone.com/news/10-handy-editor-shortcuts-netbeans-ide-60`

Tip 4: Duplicate Lines

Place the caret on the line of code that you want to duplicate, and press Ctrl-Shift-Down Arrow (Mac: Command-Shift-Down Arrow). The line is copied and pasted one line below, and the caret will move to the new line.

If you want the caret to stay on the upper line after duplication, press Ctrl-Shift-Up Arrow (Mac: Command-Shift-Up Arrow) instead.

These shortcuts also work nicely with selected blocks of code! (See also Tip 26, *Select Chunks of Code*.)

Tip Source

`http://netbeans.dzone.com/news/10-handy-editor-shortcuts-netbeans-ide-60`

Tip 5: Move Lines

Place the caret on the line of code that you want to move downward, and press Alt-Shift-Down Arrow (Mac: Ctrl-Shift-Down Arrow). The whole line is cut and pasted one line below.

To move the line upward, press Alt-Shift-Up Arrow (Mac: Ctrl-Shift-Up Arrow).

These shortcuts also work nicely with selected blocks of code! (See also Tip 26, *Select Chunks of Code*.)

Tip Source

`http://netbeans.dzone.com/news/10-handy-editor-shortcuts-netbeans-ide-60`

Tip 6: Switch Characters from Uppercase to Lowercase

Among Java programmers it is a convention that constants (such as `KEY_N` and `DEVICE_MOUSE`) are all uppercase; this is to distinguish them from variables, which are all lowercase. You can convert selected text to upper- or lowercase with a keyboard shortcut.

For instance, type and select the string `"MyExampleText"`. Then do the following.

- Press Ctrl-U and then U (Mac: Command-U U) for all uppercase.

  Example: `MyExampleText` becomes `MYEXAMPLETEXT`.

- Press Ctrl-U and then L (Mac: Command-U L) for all lowercase.

  Example: `MyExampleText` becomes `myexampletext`.

- Press Ctrl-U and then S (Mac: Command-U S) to invert a string's case; this means uppercase changes to lowercase and lowercase to uppercase.

  Example: `MyExampleText` becomes `mYeXAMPLEtEXT`.

Note that this is a multikey shortcut.

Tip Source

`http://blogs.sun.com/roumen/entry/editor_actions_discoverability`
`http://blogs.sun.com/roumen/resource/editor_hidden.html`

Tip 7: Auto-Format Source Code

Press Alt-Shift-F (Mac: Ctrl-Shift-F) to format (indent) source code. If there is a selection, only the selected lines are formatted. Use this after pasting code with Ctrl-V (Mac: Command-V) to indent the new lines.

Press Ctrl-Shift-V (Mac: Command-Shift-V) when pasting code: Using this shortcut, the lines are immediately indented.

Tip Source

```
http://blogs.sun.com/roumen/entry/keyboard_shortcuts_i_use_all
```

Tip 8: Toggle Commented Lines

Press Ctrl-/ (Mac: Command-/) to toggle commented lines. This comments out the current line of Java code (or all selected lines) by prepending //. If the lines are already commented out, the comment characters are removed.

Press Ctrl-Shift-T and Ctrl-Shift-D (Mac: Command-Shift-T and Command-Shift-D) to add another layer of comment characters to the line or remove them, respectively. This is useful when commenting out a block that already contains comments.

Tip Source

```
http://blogs.sun.com/roumen/entry/keyboard_shortcuts_i_use_all
```

Editor Insights

Tip 9: Compare Two Files Line by Line (Diff)

In version-controlled files you will notice marks in three colors: Deleted, added, and modified lines are highlighted in red, green, and blue, respectively.

Highlight color	Difference
Red	Deleted lines
Green	Added lines
Blue	Modified lines

Figure 4 shows the color coding in context. The color-coded bars along the editor's left edge allow you to track differences between your local copy and the last repository version. Navigate quickly to modified lines by clicking the corresponding marks in the error stripe to the right.

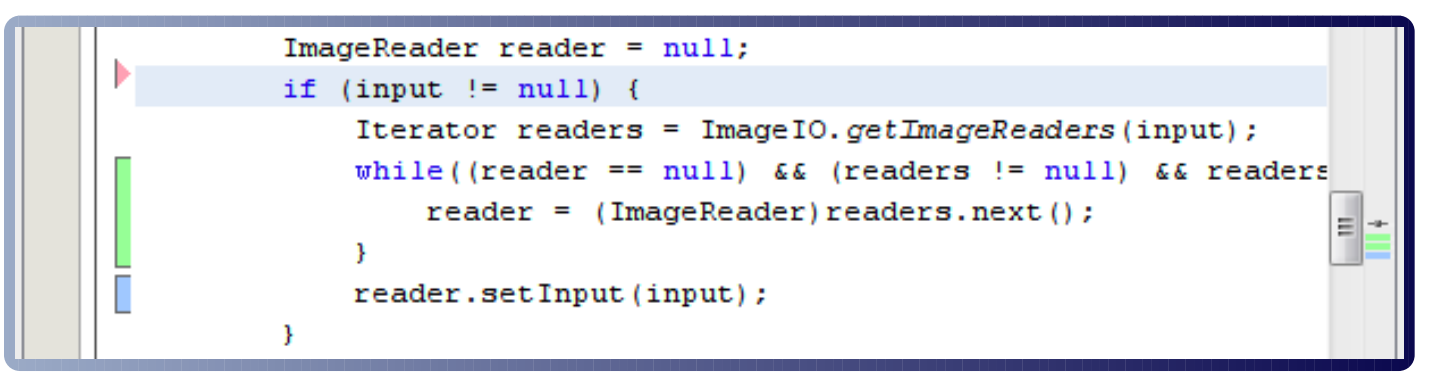


FIGURE 4: Marks representing removed lines, added lines, and modified lines

Click the vertical bars along the left edge for a pop up with inline version control actions (see Figure 5).

- Click the up and down arrows to navigate from one change to the next.
- Click the Revert button to roll back individual changes.
- Click the Diff Window button to compare two versions.

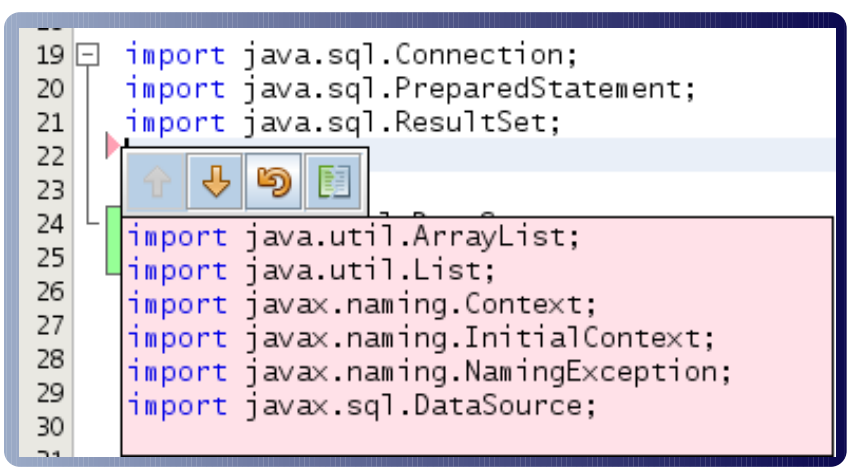
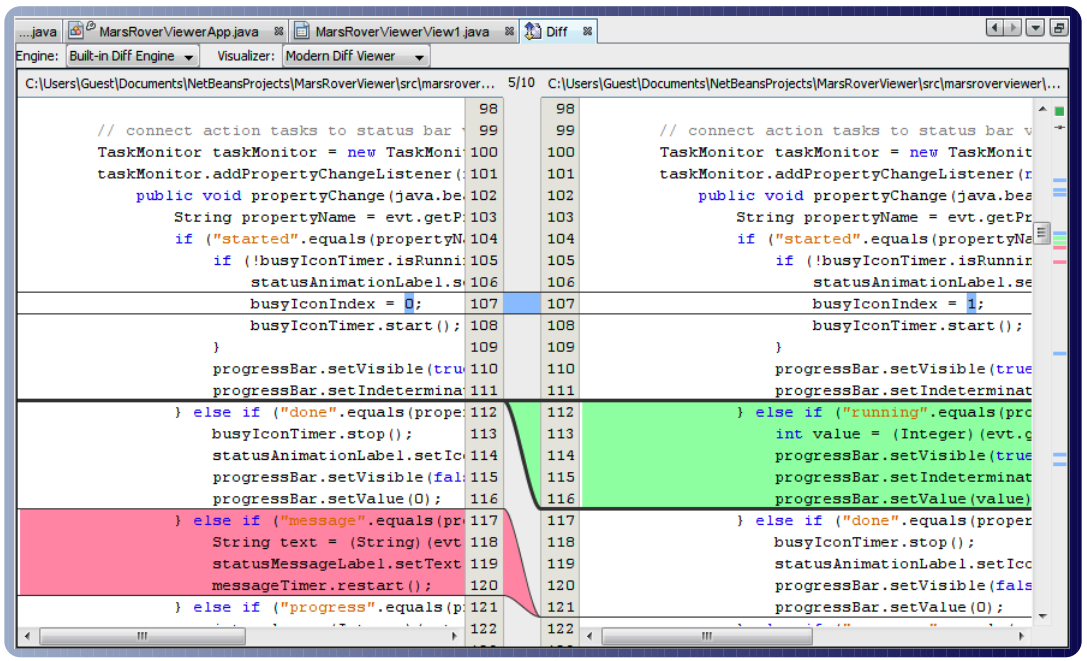


FIGURE 5: Click the bars to access inline version control actions

The built-in visual Diff Viewer highlights differences between two files (or two revisions of one file) side by side, line by line. Again, you can tell what type of change occurred by the color of the line. The Modern Diff Viewer in Figure 6 visualizes changes in a very intuitive way.

FIGURE 6:
Differences between two files



You can even compare two files that are not under version control: The Diff Viewer works on any pair of local files in the Projects, Files, and Favorites windows.

1. Select two similar files in the Projects (or Files or Favorites) window. Click with the Ctrl key (Mac: Command key) pressed to select the second file.
2. Right-click one of the highlighted files, and select Tools > Diff from the context menu. This menu item appears only when *two* files are selected.
3. The graphical Diff Viewer appears and gives you a detailed overview of all differences.

See also Tip 10, *Roll Back Changes in Files Not Under Version Control (Local History)*, and Tip 11, *Restore Deleted Files Not Under Version Control (Local History)*.

Tip Source

`http://blogs.sun.com/roumen/entry/netbeans_quick_tip_10_diffing`

Tip 10: Roll Back Changes in Files Not Under Version Control (Local History)

One advantage of keeping your files under version control is that you can merge changes when you work in a team. Another advantage is that you can revert to a previous revision in case a change breaks the build. The option to roll back changes is also very handy when you work alone and you do not use a version control system.

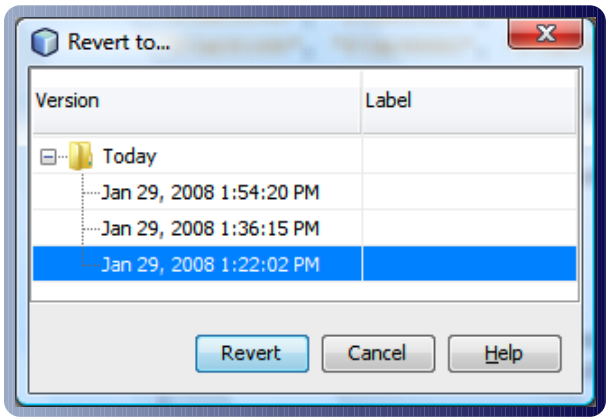
The IDE tracks modifications to project files that are open in the editor, and keeps a local history of changes. Every time you save a file, a new revision is committed.

To get an overview of changes made to a file, select the file in the Projects or Files window. Right-click the file and choose Local History > Show Local History from the context menu. The graphical Diff Viewer opens in the main window and displays two revisions side by side. (See also Tip 9, *Compare Two Files Line by Line (Diff)*.)

This is how you roll back changes and restore a previous revision.

1. Select a file in the Projects or Files window.
2. Right-click the file and choose Local History > Revert to... from the context menu. The Revert to... dialog opens.

FIGURE 7:
Revert to Window



3. Select a revision and click Revert to roll back all changes to the selected revision, as shown in Figure 7.

Note that you can only revert files that were open in the Files or Projects window. Local History does not cover files outside projects (e.g., in the Favorites window).

See also Tip 11, *Restore Deleted Files Not Under Version Control (Local History)*.

Tip 11: Restore Deleted Files Not Under Version Control (Local History)

Keeping your files under version control has many advantages. Among other things, you can even restore files that you have deleted. The NetBeans IDE also offers an equivalent "undelete" action for files that are not under version control.

1. Open the Projects or Files window and locate the project to which the deleted file belonged.
2. Right-click the directory where you deleted the file and choose Local History > Revert Deleted from the context menu. All deleted files and directories are restored.
3. If you do not remember which directory the file was in, right-click the project and choose Local History > Revert Deleted. This will restore all deleted files and directories of this project.