CHAPTER 1

Downloading, Installing, and Customizing NetBeans

Since the NetBeans 5.5 release, many changes have been made in the core of the tool. One of these is the installation process. The NetBeans team has experimented with many different kinds of installation options over the years, such as bundled downloads, separate add-on packs, and individual downloadable modules.

To start using NetBeans, you merely need to download and install it. This chapter describes how to do that as well as how to check for updates and customize some NetBeans settings to suit your own preferences. Finally, we take a quick look at the NetBeans windows you'll use most often.

Downloading Files

Starting with the NetBeans 6 release, you have several types of bundled downloads you can use to install the IDE: basic, standard, and full.

Basic: The basic bundle comprises the stripped-down IDE and basic Java Standard Edition functionality, including the Profiler tool and GUI building capabilities.

Standard: The standard bundle expands on the basic by adding mobility, web, and Java EE features. It also includes several Java application servers, such as Apache Tomcat 6 and the latest build of GlassFish.

Full: The full bundle includes all the preceding features as well as the UML, SOA, and Ruby modules. This is the largest download. But unless you really want a stripped-down version of the IDE, I suggest downloading this bundle. During the installation process, you can pick and choose which features you actually want. If you download a smaller bundle, you have no such choice.

To download the NetBeans bundles, go to netbeans.org. This web site provides several different links for downloading the NetBeans software and related tools. The main download is typically linked off the homepage.

Depending on which path you follow to get to the download section, you may be presented with several choices. The specific operating system version you need will most likely be preselected for you. But if it is not, you can choose from Windows, Linux, Solaris, and MacOS X. At the time of this writing there are 64-bit options for Linux and Solaris, but this is subject to change. You should select the bundle you need and click the Download button. You will then be immediately prompted to download the file.

Installing the NetBeans IDE

Since NetBeans can be installed across numerous platforms, I will mention only the important installation concepts. NetBeans 6 can be installed on almost any operating system for which there is a Java Virtual Machine (JVM) that runs a minimum of Java 1.5.0.11 or later. I am running NetBeans using Java 6, unless otherwise specified.

On the download page at netbeans.org, a list of release documents is provided. In this list is a link to the installation instructions as recommended by NetBeans. These instructions cover the basic installation process for Windows, Solaris, Linux, and Macintosh OS X.

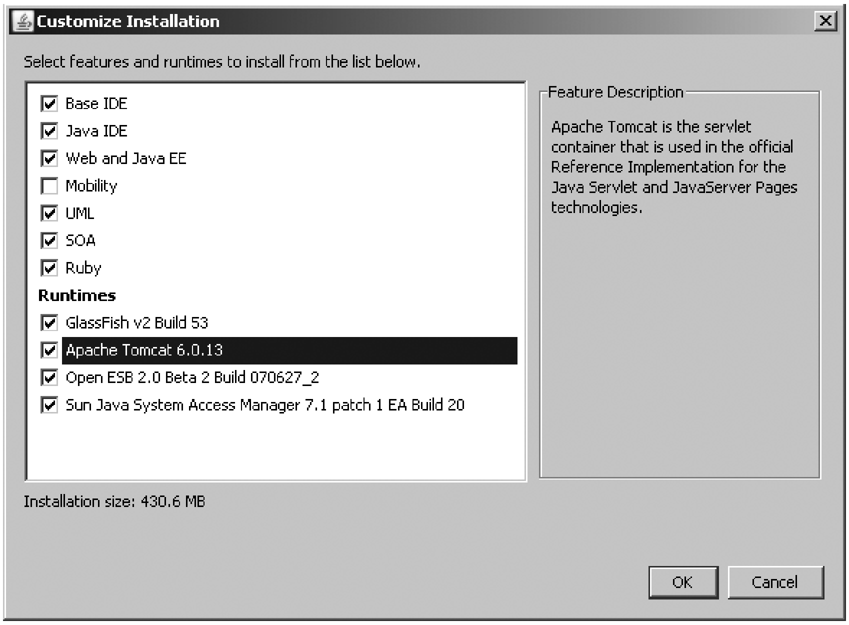
As of version 6.0 of the Profiler, it is recommended that you run it with JDK 1.5.11 or later as previously mentioned with NetBeans 6. However, for optimal performance I suggest using the most recent Java 6 release. In NetBeans 5.5 and 5.5.1 it is possible to profile against JDK 1.4, but an experimental add-on is needed to allow this functionality. Sun does not support the add-on, so you would be using it at your own risk. If your application is written for Java 1.4 and you want to use NetBeans 6 and its profiler, I recommend that you install the latest release of Java 6.0 and set the source compatibility in NetBeans to 1.4. This should resolve any issues with running the Profiler as well as maintain your Java 1.4–based code.

To set the source compatibility for a project, right-click the project name and select Properties from the context menu. With the Sources category selected, you should see a field called "Source Level." Using the drop-down list, you can set the version of Java with which your source code should be compatible.

The first time I downloaded and installed NetBeans, I used Sun's Java 5 on Windows XP, but I have since upgraded to Java 6. After executing the Windows installer, I clicked the Next button, accepted the license agreement, and selected a directory in which to install NetBeans. Personally, I like to group all my Java-related products in one location. I typically start with a `c:\java` directory. Within that directory, I install several JDKs, Java tools such as NetBeans, as well as a directory for all my Java-related projects and applications. I usually end up with the following:

- `c:\java\1.6.0.02`
- `c:\java\1.5.0.12`
- `c:\java\netbeans\6.0`
- `c:\java\projects`

When you execute the installation you will see the NetBeans 6 welcome page. Click the Customize button to select which features you want to install. The list may vary depending on the bundle you downloaded from netbeans.org. You can select or unselect the check box next to each item. By default, the Tomcat application is not checked, so to install the feature you must select it, as shown in Figure 1-1.



Select features and runtimes to install from the list below.

- [x] Base IDE
- [x] Java IDE
- [x] Web and Java EE
- [ ] Mobility
- [x] UML
- [x] SOA
- [x] Ruby

Runtimes

- [x] GlassFish v2 Build 53
- [x] Apache Tomcat 6.0.13
- [x] Open ESB 2.0 Beta 2 Build 070627_2
- [x] Sun Java System Access Manager 7.1 patch 1 EA Build 20

Feature Description

Apache Tomcat is the servlet container that is used in the official Reference Implementation for the Java Servlet and JavaServer Pages technologies.

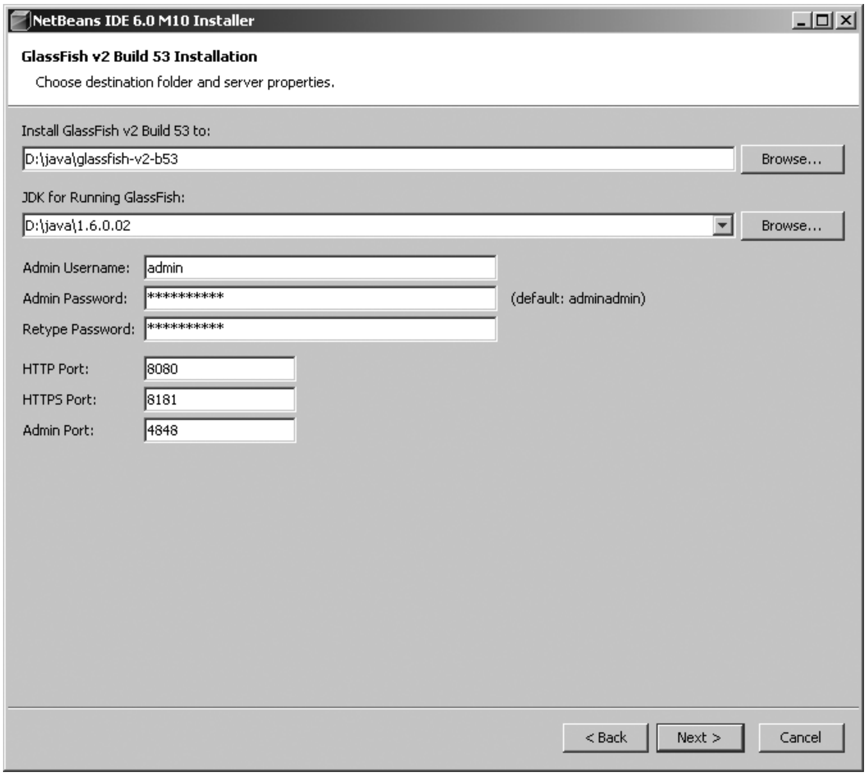
Installation size: 430.6 MB

Figure 1-1. *List of features to include in the installation*

Once you have finished customizing the installation items, click the OK button to continue. The list of items you selected will be displayed. If any of the items under the Runtimes section were previously installed, then the text "Already Installed" will appear next to each name. If you click the OK button, the installation will initialize and display the license page and acknowledge.

Select the check box next to the text "I Accept the terms in the license agreement" and click the Next button. The installation wizard will search your local machine for JDKs and prompt you for several pieces of information. For the "Install NetBeans 6 IDE to" field, click the Browse button and select a directory. As mentioned earlier, I recommend a common directory structure such as `c:\java\netbeans\6.0`. For the "JDK for running NetBeans IDE" field, a JDK is already selected. You can click the Browse button to select a directory for the JDK as well. Click the Next button to proceed to the server runtimes installation configuration.

If you selected to have GlassFish installed, the installation wizard displays a configuration form with suggested values already filled out. The "Install GlassFish to" field specifies the directory where the application server will be installed. The form also allows you to specify which JDK you wish to use when running GlassFish in case it differs from the JDK you are using to run NetBeans. This is a convenient setting to have because you may very well be running GlassFish without having NetBeans open. The configuration form also allows you to set the admin username, password, and ports, as shown in Figure 1-2.


NetBeans IDE 6.0 M10 Installer

GlassFish v2 Build 53 Installation

Choose destination folder and server properties.

Install GlassFish v2 Build 53 to: D:\java\glassfish-v2-b53

JDK for Running GlassFish: D:\java\1.6.0.02

Admin Username: admin

Admin Password: *********** (default: adminadmin)

Retype Password: ***********

HTTP Port: 8080

HTTPS Port: 8181

Admin Port: 4848

Figure 1-2. *The GlassFish application server installation configuration form*

When you have finished setting the GlassFish server fields, click the Next button to continue. Read the section on Java Application Servers in Chapter 13 if you want to know how to change these configuration settings. If you selected Tomcat to be installed, the installation wizard will display a Tomcat application server configuration form. The screen allows you to specify where to install Tomcat 6 by clicking the Browse button next to the "Installation Location" field. After you have set the location, click the Next button to proceed to the summary screen. The summary screen lists the various installation directories, the features you selected to install, and the estimated installation size of the entire package.

Click the Install button to execute the full installation. The installation process runs and installs the features. When installation is complete, click the Finish button to close the installation wizard.

When you load NetBeans for the first time, it creates a new `.netbeans` directory in the user directory. On Windows this is typically `c:\documents and settings\<username>\.netbeans`, and on Unix it's `/home/<username>/.netbeans`. You can safely remove this directory without hurting the core NetBeans installation. However, removing it will essentially undo all the configurations you have set in the IDE and uninstall all the modules you may have downloaded.

Starting with NetBeans 6, you no longer need to download and install the NetBeans Profiler separately. It is included in the core IDE. For usage instructions and best practices, see Chapter 5.

Customizing the NetBeans JVM Startup Options

One thing most people will probably never think to use is the ability to customize the NetBeans JVM startup options. By including several arguments in a NetBeans configuration file, you can tweak the memory usage settings for the JVM in which NetBeans starts up. You can also change the type of garbage-collection algorithm that is used.

If you are working on a semistandard computer (32-bit single processor), you probably won't benefit from changing the garbage-collection routine the JVM uses. However, if you use a JVM other than the Sun JVM or have a machine that is either multiprocessor, multicore, or 64-bit, you might want to consider these options. Your JVM vendor should provide some sort of documentation regarding the garbage-collection routines that run and how to configure them via command-line arguments. These can be passed along to NetBeans during startup.

In NetBeans, you can configure JVM startup arguments by editing the file `/etc/netbeans.conf` in your NetBeans home directory. In this file, you should see a property named `netbeans_default_options`. This property allows you to pass JVM customization arguments to NetBeans.

- The `-J-Xms32m` argument specifies that the initial heap size allocated to the JVM should be 32MB.
- The `-J-XX:+UseConcMarkSweepGC` argument specifies that the JVM should use a more efficient garbage-collection algorithm. It can be especially useful on multiprocessor and multicore machines.
- The `-J-XX:+CMSClassUnloadingEnabled` argument is used to enable class unloading.
- The `-J-XX:+CMSPermGenSweepingEnabled` argument must be used in conjunction with the `CMSClassUnloadingEnabled` argument.
- The `-J-Xmx256m` argument may not be present, by default; but if it is added, it specifies that the maximum heap size that can be allocated to the JVM should be 256MB.

Increasing the value of the `Xms` argument can improve performance in some applications, since the JVM would not have to keep reallocating heap space each time it needed to increase the available space. There is a lot of discussion in the Java industry about the correct way to set these parameters. The safest bet is to set the `Xms` argument to 64MB or 128MB and to set the `Xmx` argument to about 50 to 60 percent of the total memory on your system. This value may need to increase if you work with massive code bases.

You should also note that the `Xms` and `Xmx` arguments specify only the heap size of the JVM and not the total amount of memory the JVM will use, since there are items in the JVM that do not live inside the heap.

Managing Plugins and Updates

One of the most important aspects of a Java IDE tool is the ability to receive updates and fixes. Anyone who has ever written code knows that no program is ever perfect. Mistakes and bugs happen; when they do, the most important thing that can occur (other than fixing the bug) is to deliver the updated code to the users.

NetBeans allows you to check for, download, and install updates to the tools and plugins that are installed within your distribution through the Update Center. It is an integrated tool that checks one or more remote sites for any software updates that may be available. You can also check a remote site for new plugins as well as manually install an update module that was previously downloaded.

Using the Plugin Manager

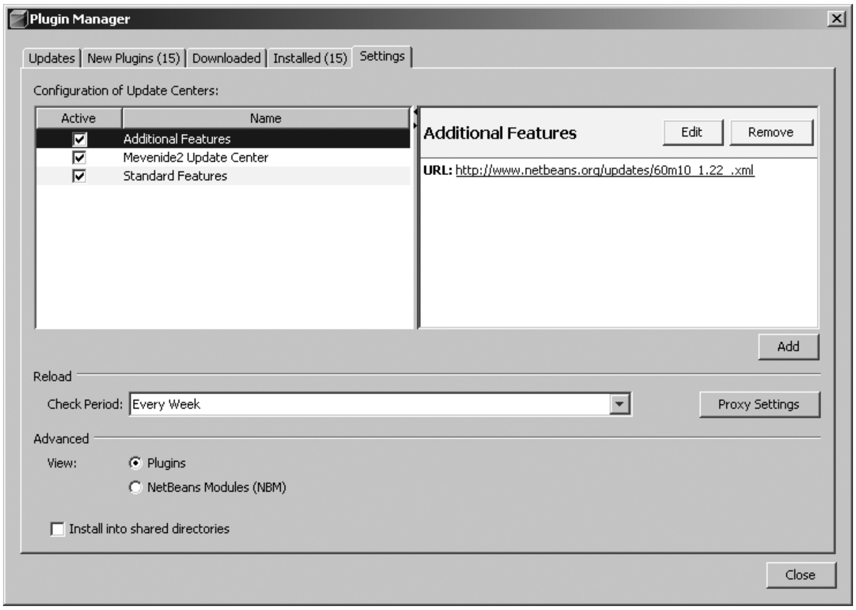
In NetBeans 6 the Update Center and Module Manager are merged into one new tool named the Plugin Manager. This new tool enables you to

- Download NetBeans plugins to install into the IDE
- Manually install previously downloaded NetBeans modules
- Check for updates to existing NetBeans plugins
- Manage already installed plugins (and be able to deactivate them)
- Configure Update Centers to check for plugins

To access the Plugin Manager, select Tools ➤ Plugins. In the window that opens, you should see five tabs along the top: Updates, New Plugins, Downloaded, Installed, and Settings. I next cover them individually, although slightly out of the order in which they appear in the Plugin Manager.

Settings Tab

If you click the Settings tab you will see the list of Update Centers currently configured in NetBeans, as shown in Figure 1-3. As you download and install new plugins, various Update Centers are automatically added to the list.



Configuration of Update Centers:

Active	Name
☑	Additional Features
☑	Mevenide2 Update Center
☑	Standard Features

Additional Features

URL: http://www.netbeans.org/updates/60m10_1.22_.xml

Reload

Check Period: Every Week

Advanced

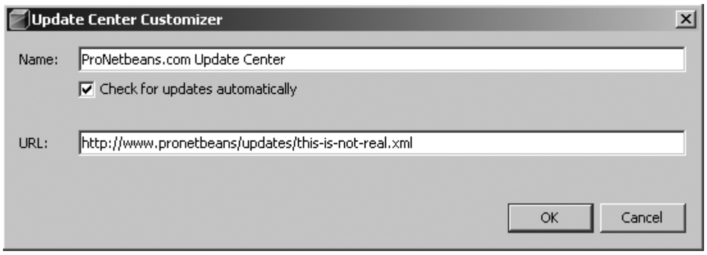
View: [✓ Plugins] NetBeans Modules (NBM)

☐ Install into shared directories

Figure 1-3. *The list of default Update Centers in the Plugin Manager*

The left pane of the Settings tab displays the list of Update Centers. If you unselect the check box next to an Update Center, it will not be searched for updates or new plugins. If you select an Update Center in the list, its information is displayed in the right pane. You can view the Update Center URL, and choose to edit or remove it from the list using the buttons displayed in the right pane, as shown in Figure 1-3.

If you wish to add a new Update Center manually, you can easily do so using the Add button at the center right of the Settings tab pane. If you click the Add button, the Update Center Customizer window is displayed, as shown in Figure 1-4. The window allows you to type in an arbitrary name to identify the Update Center as well as enter the Update Center's URL. If the check box next to the "Check for update automatically" field is selected, then the Update Center is automatically polled during the time frequency specified in the Check Period drop-down on the Settings tab. Once you have configured the desired settings for the Update Center, click the OK button and it will be added to the list.


Name: ProNetbeans.com Update Center

[✓] Check for updates automatically

URL: http://www.pronetbeans/updates/this-is-not-real.xml

Figure 1-4. *The Update Center Customizer window*

Under the Advanced section at the bottom of the Settings tab are the View options. You can set this field to "Plugin" or "NetBeans Modules." If it is set to "NetBeans Modules," you will be able to view individual features and libraries. This setting can be useful if you are trying to understand a plugin's dependencies. If the View field is set to "Plugin," then you will not see the underlying libraries. Only the top-level plugins will be listed. This setting can make it much easier to see what is installed in your IDE without having to scroll through long lists of plugins.

The "Install into shared directories" field appears at the bottom of the Settings tab. If selected, this will install plugins into a shared folder so that they are available to all users and not to just the one who installed the plugin.

The last item to note on the Settings tab is the Proxy Settings button. If you click it, the Basic Options window will appear and display the General section. This is covered later in this chapter in the section "Setting a Proxy."

Updates Tab

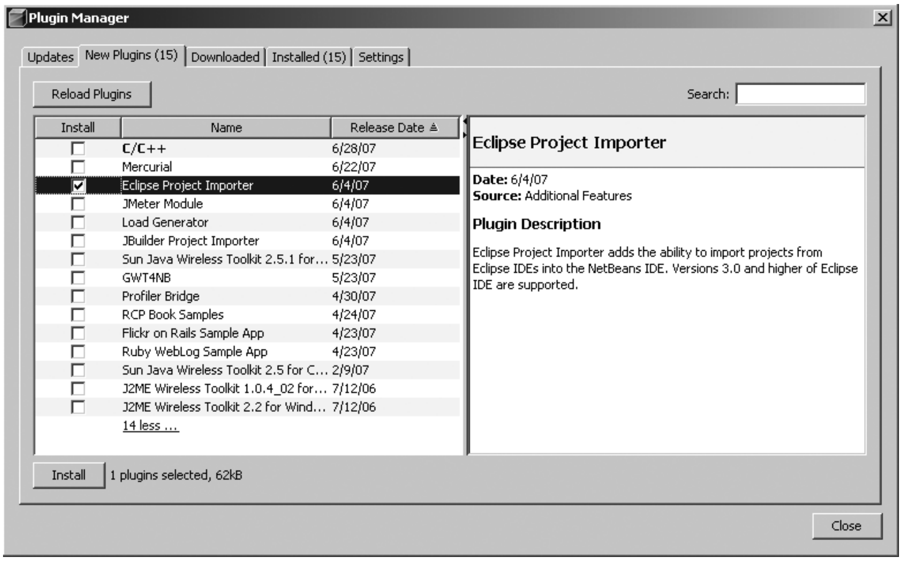
The Updates tab displays any updated plugins or libraries that may have been posted to the Update Centers. You can check for updates by clicking the Reload Plugins button near the top of the pane. This will search the Update Centers and display any updated plugins by name and release date.

To install any updates that appear, simply click the check box next to each item and click the Updates button at the bottom of the pane. The features will be downloaded and installed. Depending on the module, you may be prompted to restart the IDE.

New Plugins Tab

The New Plugins tab will display a list of new plugins and libraries that have been released on the Update Centers. If you add new Update Centers, you can refresh the list of new plugins by clicking the Reload Plugins button.

You can click each plugin name and view the release date, source, and description in the right pane, as shown in Figure 1-5.



Install	Name	Release Date
☐	C/C++	6/28/07
☐	Mercurial	6/22/07
✓	Eclipse Project Importer	6/4/07
☐	JMeter Module	6/4/07
☐	Load Generator	6/4/07
☐	JBuilder Project Importer	6/4/07
☐	Sun Java Wireless Toolkit 2.5.1 for...	5/23/07
☐	GWT4NB	5/23/07
☐	Profiler Bridge	4/30/07
☐	RCP Book Samples	4/24/07
☐	Flickr on Rails Sample App	4/23/07
☐	Ruby WebLog Sample App	4/23/07
☐	Sun Java Wireless Toolkit 2.5 for C...	2/9/07
☐	J2ME Wireless Toolkit 1.0.4_02 for...	7/12/06
☐	J2ME Wireless Toolkit 2.2 for Wind...	7/12/06
	14 less ...	

Eclipse Project Importer

Date: 6/4/07
Source: Additional Features

Plugin Description

Eclipse Project Importer adds the ability to import projects from Eclipse IDEs into the NetBeans IDE. Versions 3.0 and higher of Eclipse IDE are supported.

1 plugins selected, 62kB

Figure 1-5. *The New Plugins tab displaying a list of plugins and libraries*

You can install each plugin by selecting the check box next to the name in the Install column. As you select each check box, a summary appears along the bottom of the window, displaying the number of plugins to be installed and the total installation size.

If the list of plugins is quite long (which it can be if you are viewing the list as NetBeans Modules versus Plugins), you can search the description text. A text box labeled Search appears in the upper right of the window. If you enter text into the Search field and press the Enter key, the list will be filtered to include only plugins that match the search criteria. This is especially useful if you are searching for plugin dependencies or libraries.

Once you have finished determining which plugins you want to install, click the Install button. A summary screen will pop open and display the plugins you selected. Review the items for correctness and click the Next button. The plugin installer will display a screen of license agreements.

In older versions of NetBeans you had to view and approve multiple licenses. In NetBeans 6 you can toggle back and forth between different license agreements and approve them all at once. You can view the different licenses by selecting them from the “Display license for” drop-down. You can then approve them all by clicking the radio button next to “I accept the terms in all license agreements” and clicking the Install button.

Each of the plugins will then download and be verified. When they are done downloading, click the Finish button. Depending on which plugins you chose to install, you may be prompted to restart the IDE. If the IDE does not need to be restarted, then you will see the list of plugins on the New Plugins tab refresh.

Installed Tab

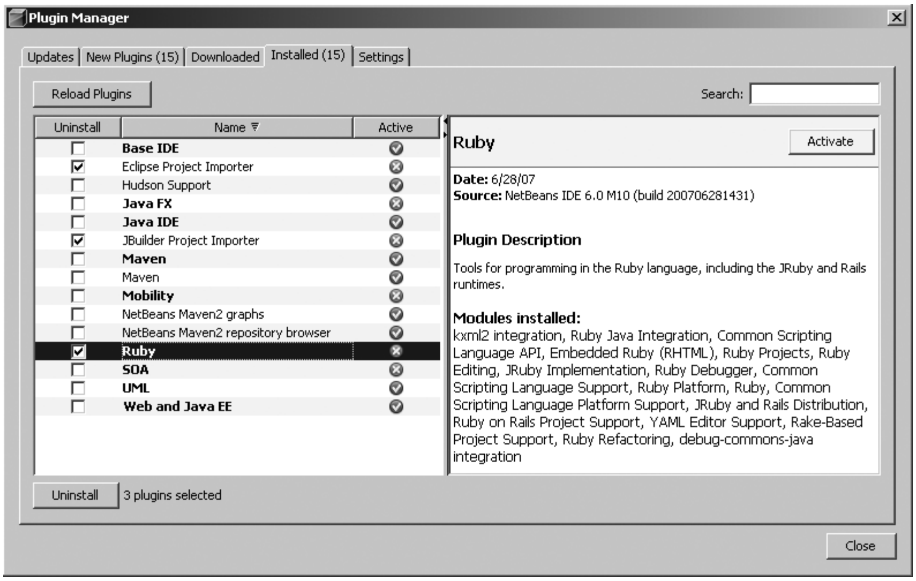
The Installed tab lists the plugins and modules that you have previously installed. This section of the Plugin Manager allows you to uninstall and deactivate plugins and modules.

If you select a plugin from the list, you can view the date, source, description, and required modules in the right pane. You will also see a Deactivate button in the upper right. If you click the Deactivate button, it will disable the plugin without having to restart NetBeans.

■ **Tip** Disabling modules or features that you rarely use can improve the startup time and memory usage of NetBeans. When NetBeans opens, its splash screen displays several status messages, such as "Reading module storage," "Turning on modules," "Loading modules," and "Starting modules." Much of the processing and work that goes on behind the scenes during startup involves activating modules. The fewer modules NetBeans must activate, the better.

If a plugin is deactivated, it will appear in the plugin list with a red X icon in the Active column. If you select a deactivated plugin from the list, the right-hand information pane will display an Activate button. If you click the Activate button, the plugin will be enabled again.

You can also uninstall plugins by clicking the check box next to each plugin in the Uninstall column, as shown in Figure 1-6. After you have selected one or more plugins, click the Uninstall button. A popup window will appear listing each plugin you selected to uninstall. To finalize the process, click the Uninstall button, and the plugins will be uninstalled.



Uninstall	Name	Active
☐	**Base IDE**	✓
☑	Eclipse Project Importer	✗
☐	Hudson Support	✓
☐	**Java FX**	✗
☐	**Java IDE**	✓
☑	JBuilder Project Importer	✗
☐	**Maven**	✓
☐	Maven	✓
☐	**Mobility**	✗
☐	NetBeans Maven2 graphs	✓
☐	NetBeans Maven2 repository browser	✓
☑	**Ruby**	✗
☐	**SOA**	✗
☐	**UML**	✓
☐	**Web and Java EE**	✓

3 plugins selected

Ruby

Date: 6/28/07
Source: NetBeans IDE 6.0 M10 (build 200706281431)

Plugin Description

Tools for programming in the Ruby language, including the JRuby and Rails runtimes.

Modules installed:
kxml2 integration, Ruby Java Integration, Common Scripting Language API, Embedded Ruby (RHTML), Ruby Projects, Ruby Editing, JRuby Implementation, Ruby Debugger, Common Scripting Language Support, Ruby Platform, Ruby, Common Scripting Language Platform Support, JRuby and Rails Distribution, Ruby on Rails Project Support, YAML Editor Support, Rake-Based Project Support, Ruby Refactoring, debug-commons-java integration

Figure 1-6. *Selecting plugins to uninstall in the Plugin Manager*

Downloaded Tab

The Downloaded tab allows you to install plugins you previously downloaded but never installed. It also allows you to install plugins you manually downloaded as .nbm files from a web site.

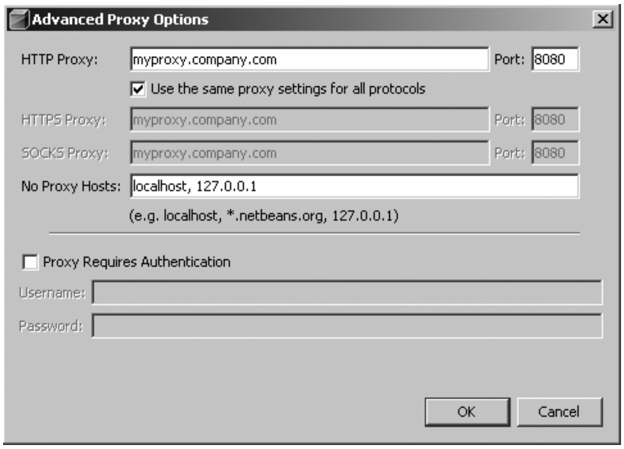
On the Downloaded tab, click the Add Plugins button to get a file dialog to appear. Navigate your file system, select one or more .nbm files, and click the Open button. The list of plugins to install will appear in a list. Once you have added the correct plugins to the list, click the Install button. A summary screen will pop up and allow you to review the plugins prior to installation. Click the Next button to finish installing the plugins.

Setting a Proxy

Many programmers, whether in corporations or on college campuses, need to work behind a proxy. The NetBeans IDE uses an Internet connection for numerous operations, such as downloading updates, linking to certain types of help documentation, and connecting to external database and web servers.

To configure the proxy settings for NetBeans, select Tools ➤ Options. The Basic Options window is displayed by default, and it contains a Proxy section. If it is not visible, click the General tab to see the proxy settings.

You can choose to select No Proxy, Use System Proxy Settings, or Manual Proxy Settings. If you select the radio button next to the Manual Proxy Settings label, then several fields will be enabled, allowing you to specify additional settings. The HTTP Proxy and Port text box allows you to enter specific proxy information. If you click the More button, you can view the Advanced Proxy Options window, as shown in Figure 1-7.



Advanced Proxy Options

HTTP Proxy: myproxy.company.com Port: 8080

☑ Use the same proxy settings for all protocols

HTTPS Proxy: myproxy.company.com Port: 8080

SOCKS Proxy: myproxy.company.com Port: 8080

No Proxy Hosts: localhost, 127.0.0.1

(e.g. localhost, *.netbeans.org, 127.0.0.1)

☐ Proxy Requires Authentication

Username:

Password:

Figure 1-7. *The Advanced Proxy Options window*

The Advanced Proxy Options window allows you to enter the proxy host and port. You can also choose to select the "Use the same proxy settings for all protocols" check box. If this is selected, then the proxy you entered is used for HTTP, HTTPS, and SOCKS. If not selected, you can enter different proxies for each protocol.

An additional field, named No Proxy Hosts, will also appear in the Advanced Proxy Options window. This field allows you to specify a list of hosts to ignore that can be accessed by NetBeans without going through the proxy. If selected, the "Proxy Requires Authentication" check box allows you to specify a username and password that is used to authenticate to your proxy server. By default this field is not selected. Once you have configured the proxy settings to your needs, click the OK button to return to the Basic Options window. Click the OK button again to save the changes and exit the Basic Options window.

Customizing the IDE

Many Java IDE tools allow a wide array of customizations. NetBeans is no exception. Users can customize a variety of settings, such as fonts, colors, text messages, coding preferences, menus, toolbars, shortcuts, and much more. You could spend an exhaustive amount of time examining each and every possible customization, so I have highlighted several key items that I believe are the most relevant and useful.

Setting the Internal Web Browser

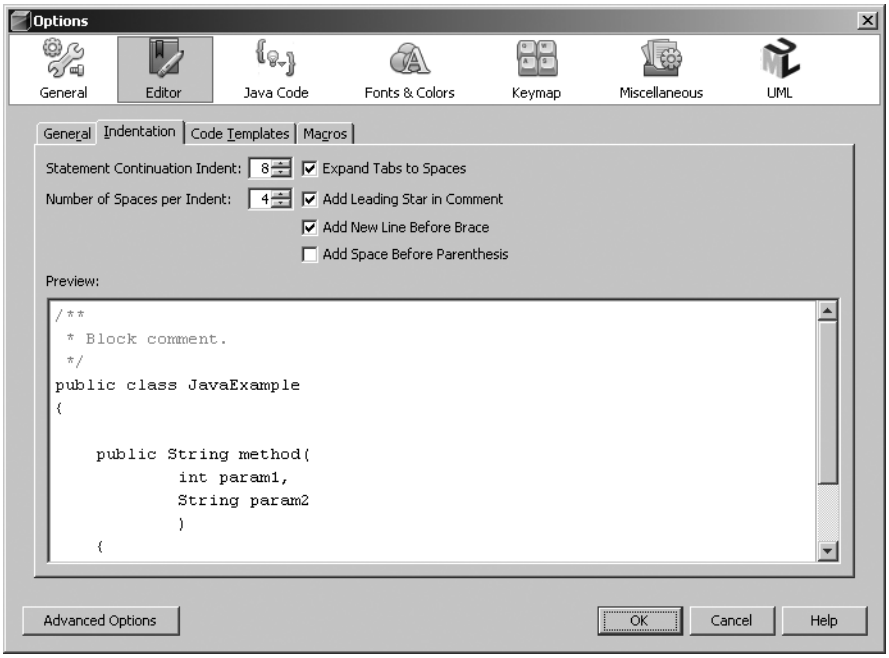
I personally do a lot of development with web-based content, so I need to be able to view that content in a convenient manner in the web browser of my choice. Sometimes I need to test web content in different browsers, especially if I am writing cross-browser JavaScript code. One of the nice features of NetBeans is the ability to set which Internet browser is used to view web content.

If you have a JSP file named `index.jsp` open and want to run it, select Run ➤ Run File ➤ Run and choose `index.jsp`. In the Output window, you will first see the application compiled, packaged, and deployed. Then the bundled Java application server will start, and finally a web browser will open.

NetBeans is initially configured to use the default browser for your system. You can change this by selecting Tools ➤ Options and selecting General in the top pane of the Options window. The Web Browser drop-down list offers Default System Browser and any browsers you have installed on your system. If you always prefer to test your web applications with Firefox, you could select it from the list.

Setting Code Editor Indentation

NetBeans allows some flexibility when configuring code indentation and formatting. When NetBeans formats source code (select Source ➤ Reformat Code) or autogenerates code, it applies several code styles. You can modify these by choosing Tools ➤ Options ➤ Editor and clicking the Indentation tab. As shown in Figure 1-8, this tab contains the code formatting options and a preview that displays how your code would be formatted if you toggled each of the available options. The following sections cover some of the more important settings.



Statement Continuation Indent: 8
Number of Spaces per Indent: 4
☑ Expand Tabs to Spaces
☑ Add Leading Star in Comment
☑ Add New Line Before Brace
☐ Add Space Before Parenthesis

Preview:

```
/**
 * Block comment.
 */
public class JavaExample
{

    public String method(
            int param1,
            String param2
            )
    {
```

Figure 1-8. *Code indentation features*

Statement Continuation Indent

When writing code such as a method declaration with parameters, the line of characters can become quite long, as in this example:

```
public void setEmployeeData(String FirstName, String LastName)
```

Some coders believe it a best practice to leave the entire method declaration and defined parameters on one line. Popular practice is to separate long lines:

```
public void setEmployeeData(String FirstName,
                            String LastName)
```

or:

```
public void setEmployeeData(
          String FirstName,
          String LastName
          )
```

This type of syntax formatting is especially useful if you are defining a method with many parameters. It is far easier to read the parameter name and data type on a separate line than to have to scroll far to the right in a very long line.

In NetBeans, you can control the amount of space by which the continued code is indented. The Statement Continuation Indent option on the Indentation tab allows you to set the number of spaces the continued line is indented. The minimum indent is 1 space, and the maximum is 50 spaces. Try changing the number of spaces and watching the preview at the bottom of the tab.

Number of Spaces per Indent

Part of writing easy-to-maintain and legible code is indenting a statement such as `if-else`, `switch-case`, or any nested statement. Most modern IDE tools have some level of default indentation. When you finish typing a statement and press the Enter key, the cursor is positioned on the next line and indented one or more spaces.

In NetBeans, you can customize this formatting. The default number of spaces for indentation is four. The Number of Spaces per Indent option on the Indentation tab allows you to adjust this setting from 1 to 50. To see this in action, type a line of code like `if(myBooleanValue)` and the opening curly brace, and press Enter. You should see the following:

```
if(myBooleanValue) {
    //start here
}
```

The beginning of the comment marks where the cursor should be positioned. It should be indented four spaces from the position of the enclosing parent element. If you continued to press the Enter key and enter comments, you would see code like this:

```
if(myBooleanValue) {
    //start here
    //comment2
    //comment3
    //comment4
    //comment5
}
```

Notice that the line spacing for each of the lines inside the `if` block is the same. The IDE monitors the fact that all the statements you are typing have the same scope (I use this term loosely) and attempts to maintain similar spacing. After the fifth comment line, add another `if` block and see how the spacing is maintained. After typing `if(myNextBooleanValue)` and an opening curly brace, and pressing Enter, add several lines of comments. The following code should appear:

```
if(myBooleanValue) {
    //start here
    //comment2
    //comment3
    //comment4
    //comment5
    if(myNextBooleanValue) {
        // comment 6
        // comment 7
    }
}
```

If you want to see a better example of this, try setting the Number of Spaces per Indent option to 20 and typing several nested `if-else` blocks of code.

Expand Tabs to Spaces

When indenting code, some developers like to add tabs and some like spaces. For many people this is not a huge issue; but if you open code in different tools, it can cause formatting problems. If your development team works on a variety of operating systems or uses a variety of tools such as Emacs, vi, NetBeans, and Notepad, you may have run into this issue.

The Expand Tabs to Spaces option in NetBeans automatically converts any tab characters into an equivalent number of spaces. One tab character is the number of spaces set for the Number of Spaces per Indent option. This can help reduce formatting issues and spacing problems.

Add New Line Before Brace

A lot of development shops enforce a set of standard coding practices. One element of good coding is to use a consistent method of placing curly braces after method declarations, `if` blocks, `else` blocks, `switch` statements, `for` loops, and so on.

If you've ever worked on code that another developer created, you've probably seen bad coding styles. A typical piece of code for a method might look like this:

```
public int calculate(int x, int y, int z)
{
    int total = 0;
    if(x<y) {
        total = x + y;
    }
    else if(x > y)
    {
        total = x + z;
    }
    else {
        total = x+ y + z;
    }
    return total;
}
```

Notice that the opening curly brace for each block is either on the same line immediately following the statement or on the following line. Depending on how intricate your code is, this can lead to hard-to-read files, especially if numerous code blocks are embedded within code blocks embedded within code blocks.

I remember a project where I inherited code from another developer. I needed to trace what happened throughout a very long method. In the method were numerous loops and `if-else` blocks. Lining up curly braces to see where one block ended and another started soon grew frustrating. Using the NetBeans Add New Line Before Brace option and applying the formatting to the source code would have saved me a lot of headaches.

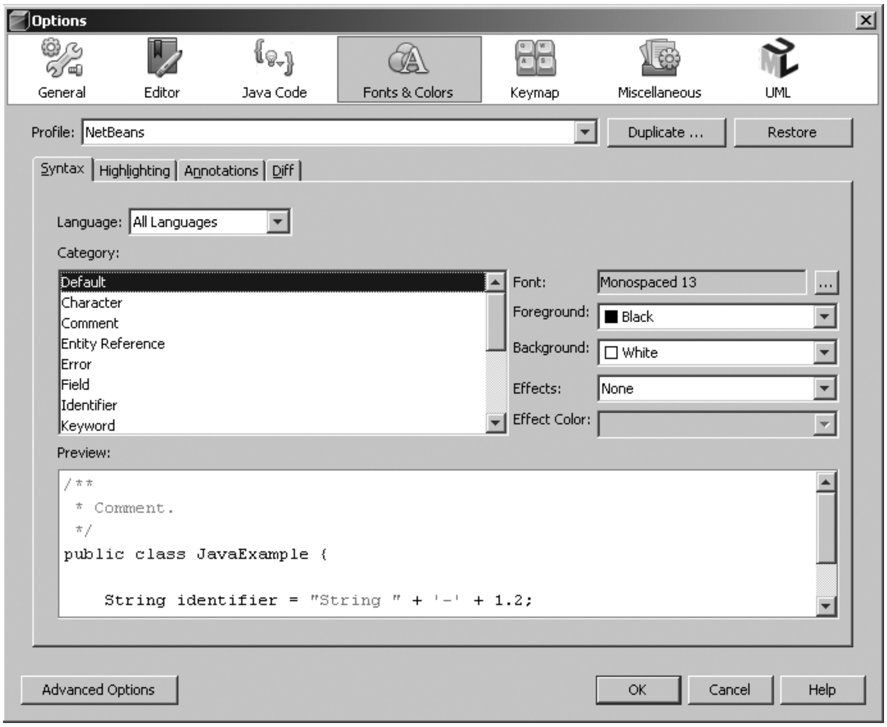
Choosing Fonts and Colors

NetBeans provides the ability to customize the overall appearance of fonts and colors. With the variety of development tools on the market, many developers are familiar with a different look and feel. For example, if you've been coding in Emacs and now are making the move to NetBeans, perhaps you want the code syntax to reflect the color scheme you used in Emacs. NetBeans provides several levels of customization.

Fonts and Color Profiles

The font and color customizations are grouped into a color profile. The profile contains the settings for customizing colors for language-specific syntax, highlighting, and annotations. Color profiles allow you to change numerous settings and toggle back and forth between them.

To customize color profiles, select Tools ➤ Options ➤ Fonts & Colors and select the Syntax tab. This tab displays the relevant settings, as shown in Figure 1-9.



Profile: NetBeans

Language: All Languages

Font: Monospaced 13
Foreground: ■ Black
Background: □ White
Effects: None
Effect Color:

```
/**
 * Comment.
 */
public class JavaExample {

    String identifier = "String " + '-' + 1.2;
```

Figure 1-9. *Fonts & Colors settings in the Basic Options window*

NetBeans provides several color profiles: NetBeans, Earth, NetBeans 5.5, Norway Today, and City Lights. These can be modified as frequently as you wish and later restored to their initial settings by clicking the Restore button next to the Profile drop-down list. I recommend creating a copy of each of the NetBeans color profiles and modifying them as you see fit. You can do so by selecting the profile and clicking the Duplicate button to the right of the Profile drop-down list. This is useful because a profile that is a copy of another profile cannot be restored; it can only be deleted. The Duplicate button is also the mechanism for creating new color profiles. After clicking the Duplicate button, you are prompted for a new color profile name.

I use color profiles most often when giving presentations. For writing code, I typically use a color profile similar to the system default profile. When giving presentations, where I need to project my screen, I switch to an alternate profile that uses a larger font size and darker font color scheme.

The standard code syntax highlighting and coloring changed in NetBeans 6. Many developers used to the older NetBeans syntax coloring can switch to the NetBeans 5.5 profile by selecting it from the Profile drop-down.

Language-Specific Color Settings

For each color profile in the Fonts & Colors section, a list of language-specific settings is available. The list includes settings such as Java, JSP, HTML, SQL, XML, and more. By selecting one item from the list, you can customize the language-specific syntax elements. For example, if you select JSP, you will see a category list of items such as EL, HTML, JSP, and XML elements. Once you have selected an item from the category list, you can change the font size and color and other attributes using the options on the right (see Figure 1-9).

Highlighting

Through the Highlighting tab of the Fonts & Colors window, you can customize the foreground and background colors for elements that appear highlighted in various places in NetBean's Source Editor window. One possible use of this includes changing the color of the line numbers in the line number bar so that they are easier to see at a glance. I like to set the Code Folding Bar foreground to red so that the lines that appear next to each method stand out and are easier to trace when I am scrolling quickly through a class.

Diff Coloring

The Diff tab of the Fonts & Colors section allows you to specify the coloring for the new NetBeans 6 Diff tool. The Diff tool is used by the various version control modules and local history in NetBeans (see Chapter 6).

You can modify the color scheme for the Added, Changed, and Removed Text. If you select each field, you can specify the color using the Background drop-down at the right of the window. Then when you use the Diff tool, the color changes will display.

Annotation Coloring

The Annotations tab of the Fonts & Colors window allows you to customize several items related to glyphs that appear in the glyph margin. The glyph margin is the gray vertical strip that appears in the Source Editor to the immediate left of any source code. If you select View ➤ Show Line Numbers, the line numbers are also displayed in the glyph margin. You can change the foreground color and the background color of annotations as well as set the color of the Wave Underlined property for the Parser Annotation (Error) category. The Wave Underlined property pertains to the wavy line that appears under parser errors in your code, such as statements that will not compile.

Tip If you really want a parser error to stand out, you can set the background color to black, the foreground color to green, and the Wave Underlined property to red. Once you have done this, it should not be too difficult to spot compilation errors.

Configuring Keymaps

Every good software tool should provide shortcut keys (also known as hotkeys). NetBeans is no exception. Since the majority of the work carried out in NetBeans involves typing code, it is obviously convenient not to have to take your hands off the keyboard very often. Many menu commands, actions, and tools can be activated via keyboard shortcuts. NetBeans categorizes a group of shortcuts as a *keymap*.

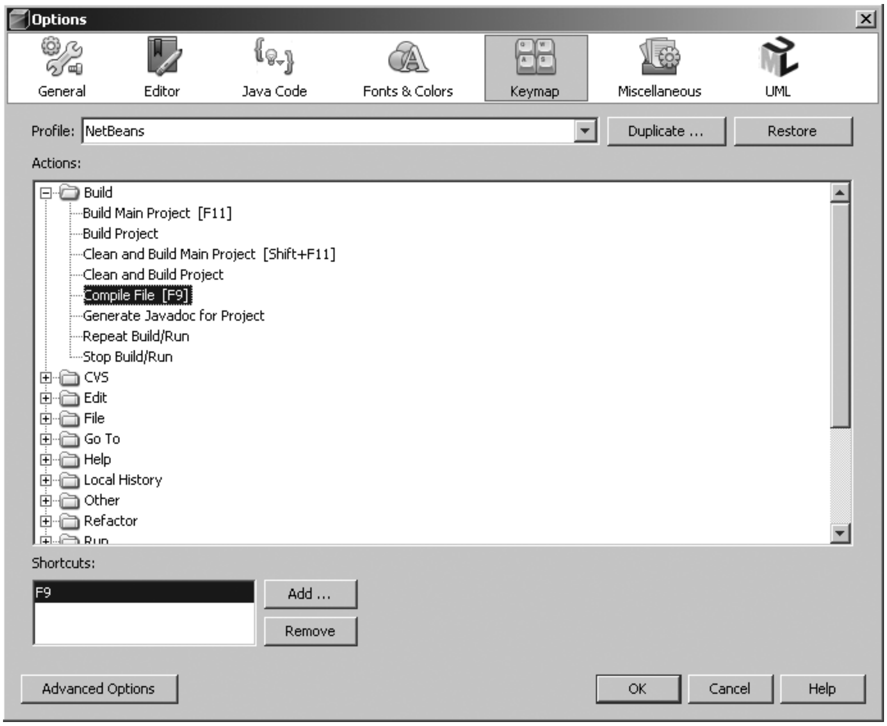
Keymaps can be configured in the Basic Options window. Select Tools ➤ Options and choose Keymap from the top pane in the Options window. The default set of keyboard shortcuts (the keymap) is named NetBeans. You can copy any of the existing keymaps by selecting one from the Profile drop-down list and clicking the Duplicate button. The new keymap profile can then be customized any way you wish.

Tip The Profile drop-down contains an entry named Eclipse. If you select this option, then the NetBeans keymap will attempt to use the standard keyboard shortcuts prevalent in the Eclipse IDE. This is useful for developers who switch between the two IDEs.

To modify a specific keyboard shortcut, select the profile and locate which action you want to change. For example, you may want to change the shortcut used to compile a single file. To modify the shortcut, do the following:

1. Select the profile in the Keymap section of the Options window.
2. Click the plus sign next to the Build node.
3. Select the Compile File node under the Build node.
4. In the Shortcuts section at the bottom of the window, click the Remove button.
5. Click the Add button.
6. In the window that opens, press the key or keys that you want as the shortcut. The text representing those keys should be added to the Shortcut field.
7. Click the OK button.

NetBeans will prevent you from adding a duplicate keyboard shortcut. If the key or keys you pressed in the Shortcut pop-up window match another shortcut, then after clicking the OK button you will receive an error message. If the shortcut was successfully assigned to the action, it will be displayed in the Shortcuts list, as shown in Figure 1-10.



Profile: NetBeans

Actions:

- Build
  - Build Main Project [F11]
  - Build Project
  - Clean and Build Main Project [Shift+F11]
  - Clean and Build Project
  - Compile File [F9]
  - Generate Javadoc for Project
  - Repeat Build/Run
  - Stop Build/Run
- CVS
- Edit
- File
- Go To
- Help
- Local History
- Other
- Refactor
- Run

Shortcuts:

F9

Figure 1-10. *Customizing keymaps in the Basic Options window*

Setting Advanced Options

In the Basic Options window (select Tools ➤ Options), you'll find a button to access the Advanced Options window. From the Advanced Options section, you can customize a wider variety of features than is possible through the Basic Options section. The settings for toolbars and menu bars are covered here. Other Advanced Options settings, such as those for system and external tools, are covered in later chapters.

Menu Bars

NetBeans allows you to customize the system menu bars. In the Advanced Options section, select IDE Configuration ➤ Look and Feel ➤ Menu Bar. Here, you can rename the menu options such as File, Edit, View, and so on. If you really wanted to, you could rename the View menu to Look At Stuff.

By right-clicking menu names, you can delete menus, change menu-item ordering, and add menu separators. By right-clicking menu items, you can cut or copy a menu item and paste it into another menu.

A useful application of this feature is to reorder menu items to be grouped in a manner that makes more sense to you. For example, on the Refactoring menu, I have rearranged the items I use most often to appear on the top, followed by a separator and the rest of the items. I have also found this functionality useful on the Run and Window menus.

Toolbars

NetBeans also allows you to customize toolbars in much the same way you customize menu bars. You can create new toolbars, reorder the elements of a toolbar, and copy and paste elements from one toolbar to another.

Being able to create custom toolbars is an important feature if you have installed a lot of plugins that are similar and want to group quick-access icons in one toolbar location.

Server and External Tool Settings

As previously mentioned, you can set a default web browser in the NetBeans Basic Options section. However, you were not previously able to determine where the browsers were located on your system or to add new ones. The Advanced Options window allows you to do so.

Navigate to IDE Configuration ➤ Server and External Tool Settings ➤ Web Browsers. Expand the Web Browsers node to view the list of Web Browsers that have automatically been configured. On a Windows machine you would typically only see a listing for Internet Explorer. If you installed Firefox, it would also be listed.

If the browser you want to use is not in the list but is installed on your machine, you can easily add it. Right-click the Web Browsers node and select New ➤ External Browser. In the New External Browser window that appears, type in the name of the browser and click the Finish button.

The new browser will appear under the Web Browsers node, allowing you to select it. Once you have selected it, the right pane of the Advanced Options window will display the Browser Executable property. This field specifies where on your machine NetBeans must look to find the browser. You can change this location anytime by clicking inside the field and typing or by clicking the ellipsis button next to it. If you click the ellipsis button, the Browser Executable Properties window will appear, as shown in Figure 1-11.

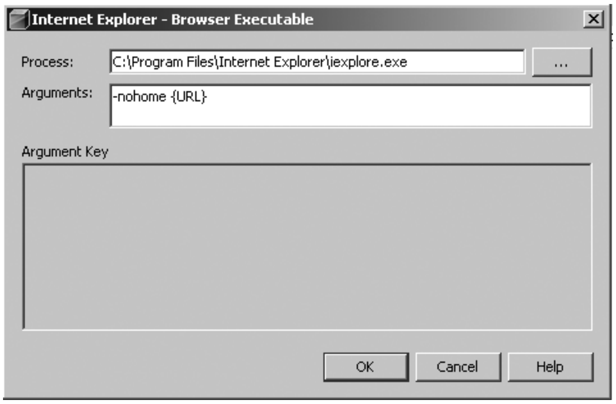

Internet Explorer - Browser Executable
Process:
C:\Program Files\Internet Explorer\iexplore.exe
...
Arguments:
-nohome {URL}
Argument Key
OK
Cancel
Help

Figure 1-11. *The Browser Executable Properties window*

This window allows you to specify the path to the browser executable. You can also click the ellipsis button next to the field to display a File Open dialog to search for the executable. You can also specify command-line arguments using the Arguments text box. NetBeans passes the `-nohome` argument to Internet Explorer so that your default browser homepage will not load, saving time. The `{URL}` argument specifies a placeholder that NetBeans will replace with the actual URL you choose to load when you run a web application, single JSP or HTML file, etc.

There are a number of web browser–specific command-line arguments you can specify to assist you in web application development; see Tables 1-1 and 1-2. Once you have entered any command-line argument, click the OK button to close the window.

Table 1-1. *Firefox Command-Line Arguments*

Argument	Description
ProfileManager	Starts Firefox with a Profile Manager dialog. Allows you to specify which profile you wish to use to load Firefox.
Jsconsole	Loads a JavaScript console. This can be a big time saver when doing web development.
height	Specifies the height the browser window will be when it opens. Example: height 500.
width	Specifies the width the browser window will be when it opens. Example: width 500.
inspector	Displays the DOM inspector. Loads the DOM inspector into the top half of Firefox and the web page to be displayed in the bottom half.
{URL}	Loads the specified URL in Firefox when it opens.

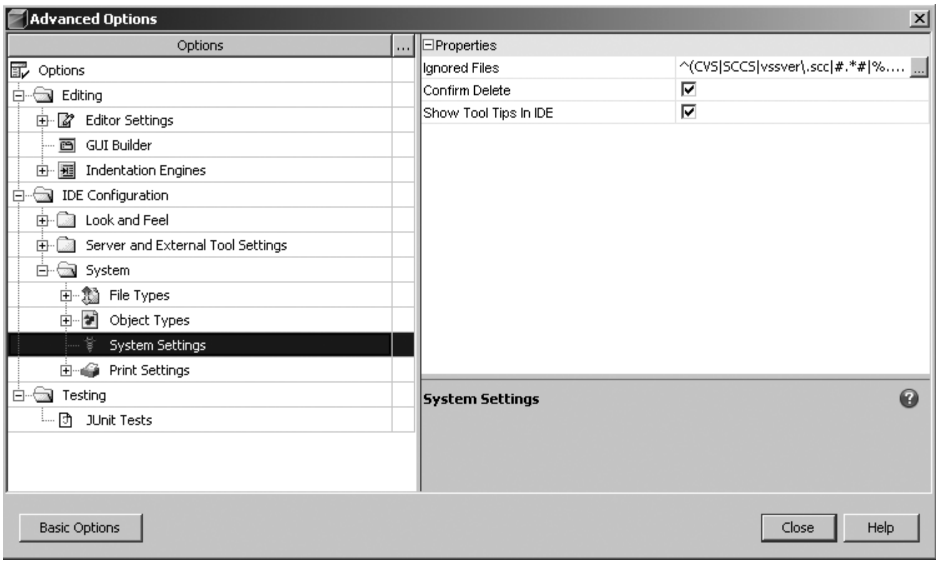
Table 1-2. *Internet Explorer Command-Line Arguments*

Argument	Description
nohome	Specifies that the default browser homepage will not load
k	Specifies that the browser will open in full-screen mode
new	Specifies that the browser window will open a new browser, and thus a new process, instead of reloading in the same window

System

There are also several system settings you can customize in the Advanced Options window. Select IDE Configuration ➤ System ➤ System Settings. Several properties will appear in the right-hand pane, as shown in Figure 1-12. If you select the “Show Tool Tips in IDE” check box, then tool tips will be displayed throughout NetBeans as you mouse over various icons and features.

The “Confirm Delete” check box is a nice safety feature. If this is selected, you will be prompted to confirm a deletion before a file, folder, or other item can be deleted. I recommend leaving this checked, which it is by default, since even the most experienced users can mistakenly delete something.



```
Properties
Ignored Files            ^(CVS|SCCS|vssver\.scc|#.*#|%....
Confirm Delete           ✓
Show Tool Tips In IDE    ✓
```

System Settings

Figure 1-12. *The Confirm Delete setting in the Advanced Options window*

Navigating and Understanding the IDE Layout

Once you have installed NetBeans and customized it to your liking, it is time to start working on projects. But first you need to become familiar with the general layout of the NetBeans IDE. It provides numerous windows that enable you to view specific items or pieces of data.

Initial Layout

When the IDE first opens, the Projects, Files, and Services window is displayed on the left. The primary menu and toolbar are displayed along the top, allowing quick access to commonly used features. Other than those features, it is up to you to decide what to display and which windows you will use. See the following section ("Windows") for an explanation of some of the different windows.

The primary toolbar displayed along the top contains only a bare minimum of icons that allow you to perform frequently used operations, icons such as Open Project, New Project, New File, Cut, Copy, Paste, Build Main Project, and Run Main Project. You can customize the toolbars that are displayed and items that appear in them by right-clicking on an empty spot of the toolbar.

The context menu that appears allows you to select or unselect different toolbars from being displayed, such as Build, Debug, Edit, File, and Memory. You can also completely customize the toolbars by selecting Customize on the context menu. This will open the Customize Toolbars window, as shown in Figure 1-13. This window offers the same functionality as the toolbar configuration section in the Advanced Options window, but it has one additional benefit: you can click and drag icons from the Customize Toolbars window directly onto the primary toolbar in the IDE.

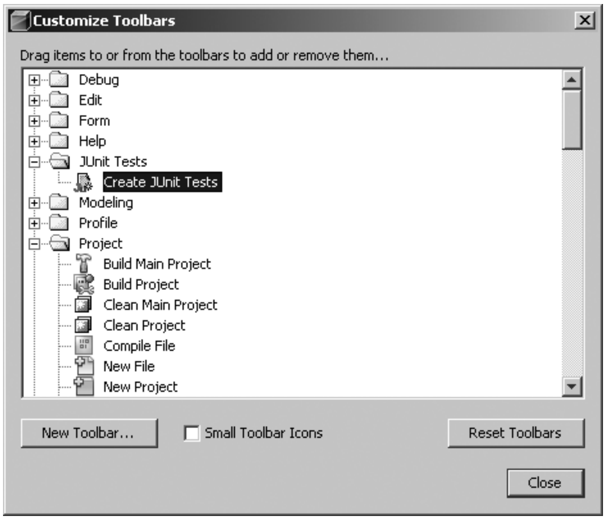


Figure 1-13. *The Customize Toolbars window*

One of the nice features on the toolbar context menu I like to use is the Small Toolbar Icons option. If this option is selected, then the primary toolbar in the IDE will shrink the icons to a smaller size. This saves space on the screen. (I like to have as much visual space as possible when working with code in the Source Editor.)

If you're programming an application and you have concerns about memory, I highly recommend selecting to display the Memory toolbar from the context menu that is displayed by right-clicking the primary toolbar. You can also activate it by going to View ➤ Toolbars ➤ Memory. This will display a JVM memory meter in the toolbar that lists the amount of memory currently used versus allocated. The memory toolbar can be useful when you launch web applications, since you can watch the memory allocated as a web application server starts up. This can be a quick and dirty way of monitoring memory usage without having to profile an application or use a tool such as JConsole.

■ **Tip** If you haven't already figured it out, you can click the Memory toolbar to force garbage collection. If you have a long-running process or a running application server started from inside NetBeans that is hogging resources, you can try to reclaim it.

Windows

There are multiple windows you can open and use throughout the IDE windowing system. Each window has a specific purpose and can be opened, minimized, or closed. Each window can also be dragged around and docked in virtually any place in NetBeans. I cover some of the more commonly used windows next.

Projects Window

The Projects window displays all the currently opened projects. It is the main entry point for NetBeans to categorize and group files for use in an application. A project can be a Java Application, an EJB Module, a Web Application, a Mobile Class Library, a NetBeans Module Project, or another type. If you need to jump quickly to the Projects window, you can toggle to it by pressing Ctrl+1.

The layout and usage of elements in the Projects window when working with files, folders, and projects is covered in more detail in various chapters throughout this book.

Files Window

The Files window provides a more normal file-based view of open projects. It contains the same information that is displayed in the Projects window but is organized in a manner that may be more familiar to you. The files in a project are organized in a folder-and-file structure that represents how your project would look if you used a file explorer outside NetBeans to view it. If the Files window is not active, you can toggle to it by pressing Ctrl+2.

Services Window

The Runtime window is where you can find important resources such as HTTP servers, database servers, web services, DTD and XML schema catalogs, and processes. You can access the Services window by selecting Window ➤ Runtime or pressing Ctrl+5.

Navigator Window

The Navigator window provides a quick-and-easy view of a node that has been selected in the Projects window or Source Editor. It can display the methods, constructors, and fields in a class in a traditional list view or as an inheritance tree. For classes with numerous methods, this can be a convenient way to jump back and forth between methods or fields. You can display the Navigator window by selecting Navigator from the Window menu or by pressing Ctrl+7.

Source Editor

The Source Editor window is where you edit code and other files. This is where the "magic" happens. When you open files, they appear in the Source Editor window as a tabbed view. The files displayed in this window can be arranged in several different ways (more on this in the next chapter, where you'll create your first Java application). If you have a file already open in the Source Editor, you can quickly toggle to it by pressing Ctrl+0.

■ **Tip** Pressing and holding Ctrl+Tab allows you to select and switch back and forth between open tabs in the Source Editor window, similar to how you can toggle between open applications in Windows using Alt+Tab.

Output Window

The Output window can display a variety of information. If you choose to build your project, compile a single file, or run a file that outputs text to the standard output or standard error stream, the information and results are displayed in the Output window. If the Output window is not displayed, you can select Window ➤ Output ➤ Output or press Ctrl+4 to open it.

Properties Window

The Properties window displays the attributes and properties of either the element currently selected in the Projects window or the item that is highlighted in the Source Editor. To see this in action, open the Properties window by selecting Window ➤ Properties or pressing Ctrl+Shift+7.

Once the Properties window opens, navigate through the Projects window and select a Java source file in an open project. The Properties window will display several attributes for the Java file, such as the filename, file size, last modified date, and classpath.

Double-click the source file. Once it opens in the Source Editor, select any method by clicking the method name. Notice that the Properties window changes to display the attributes of the method, such as the method name, access modifier, parameters, return type, any exceptions, and Javadoc comments.

The Properties window can be very convenient when used in conjunction with the Projects window for quickly navigating file structures and viewing attributes.

Palette Window

The Palette window displays a context-sensitive list of elements that are useful for the current file you are editing in the Source Editor. You can open the Palette window by selecting Window ➤ Palette or pressing Ctrl+Shift+8. If you selected a JSP file, the Palette window would display HTML, JSP, JSTL, and database elements. If you open a Java Swing source file in the Source Editor, the Palette window is filled with visual elements to be used in Swing projects.

Summary

In this chapter we discussed installing NetBeans, updating features, and customizing internal properties.

We covered installing NetBeans into standard directory structures and associating the correct JDK. We also covered how to customize the application server properties during the installation procedure. Performance considerations for the tool's startup time were also discussed. We reviewed several suggested settings that you may want to tweak to obtain optimal performance, such as altering the garbage collector that is used and specifying heap and memory arguments.

You can configure a variety of NetBeans settings based on your preferences. In this chapter we also discussed how to configure the Basic and Advanced Options (accessible from the Tools main menu) to set these properties. You can configure everything from code formatting to fonts and colors and menus and toolbars.

Lastly, we reviewed some of the windows that programmers will see and use most often. These windows provide various pieces of information and quick access to certain features. Understanding where and how to use them is critical to becoming an expert user of NetBeans.