

### CONTENTS INCLUDE:

- About NetBeans IDE
- Java Editor Overview
- NetBeans IDE Java Quick Start Tutorial
- Keyboard Shortcuts and Code Templates
- Hot Tips and more...



# NetBeans IDE Java Editor

By Geertjan Wielenga and Patrick Keegan

## ABOUT NETBEANS IDE

The NetBeans IDE has seen adoption snowballing over the past years, particularly with the introduction of a completely new, rewritten, slick Java editor. You'll find this reference card helpful if you want to get as much out of the Java editor as its authors intended when creating it. It lists all the keyboard shortcuts in

carefully thought out categories and it provides a thorough exposition of optimal handling of Java code in the editor, covering viewing, navigation, source handling, and refactoring. Get NetBeans IDE: <http://www.netbeans.org/downloads>

## JAVA EDITOR OVERVIEW

Buttons for the most frequently used actions, such as comment/uncomment.

Click on lightbulbs for tips (Alt-Enter).

Error underlinings in red, by default.

Common colors for common syntax elements, which can be modified in the Options window.

Right margin, set to 80px by default, can easily be changed in the Options window (under Tools).

Right-click inside the editor to produce a list of menu items, for refactoring and more.

Error marks in the right sidebar can be clicked to jump to the line in which the error/warning occurs.

Sidebar indicators tell you, at a glance, whether or not a file has errors/warnings.

## WHAT'S NEW FOR JAVA IN NETBEANS IDE 6.5

The following features are new in NetBeans IDE 6.5 and can be of particular use to you in the context of the Java editor.

<b>Enhanced Java Web Start Support</b>	Right-click a project, choose Properties, and you can configure applets to be deployed via Java Web Start on JDK 6 Update 10 and above.
<b>Javadoc Analyzer</b>	Changes in your code mean changes in your Javadoc. Choose Tools   Analyze Javadoc and the Javadoc will be updated according to your code changes.
<b>Groovy Support</b>	Mix and match Groovy and Java for the first time in NetBeans IDE 6.5.
<b>Quick Search</b>	Know about an action but not how to invoke it? Click Ctrl-I and the cursor lands in the text field in the top right of the editor, where you can type a keyword (such as 'format') and then related items appear.
<b>Eclipse Project Import and Synchronization</b>	Enhanced feature in 6.5, letting you resynchronize your work in the IDE to its original Eclipse project. Ideal for teams using multiple IDEs simultaneously.

**NetBeans IDE 6.5** is the latest release of Sun's award-winning open-source IDE that enables developers to rapidly create Web, enterprise, desktop, and mobile applications with Java, C/C++, JavaScript, Ruby, Groovy and PHP.

## NETBEANS IDE JAVA QUICK START TUTORIAL

This tutorial provides a very simple and quick introduction to the NetBeans IDE workflow by walking you through the creation of a simple "Hello World" Java console application. Once you are done with this tutorial, you will have a general knowledge of how to create, build, and run applications in the IDE.

To follow this tutorial, you need the following software and resources:

Software or Resource	Version Required
NetBeans IDE	Version 6.5, Version 6.1 or Version 6.0
Java Developer Kit (JDK)	Version 5 or higher

1. Start NetBeans IDE. In the IDE, choose **File > New Project (Ctrl-Shift-N)**.
2. In the New Project wizard, expand the Java category and select Java Application. Then click Next.
3. In the Name and Location page of the wizard, type **"HelloWorldApp"** in Project Name and type **"helloworldapp.HelloWorldApp"** in the Create Main Class field. Then Click finish.
4. Because you have left the Create Main Class checkbox selected in the New Project wizard, the IDE has created a skeleton class for you. You can add the **"HelloWorld!"** message to the skeleton code by replacing the line:
 

```
// TODO code application logic here
```

 with the line:
 

```
System.out.println("Hello World!");
```
5. From the IDE's menu bar, choose **Run > Run Main Project (F6)**. The Output window should show you the **"HelloWorld!"** message.

## KEYBOARD SHORTCUTS & CODE TEMPLATES

### Finding, Searching, and Replacing

Ctrl-F3	Search word at insert point
F3/Shift-F3	Find next/previous in file
Ctrl-F/H	Find/Replace in file
Alt-F7	Find usages
Ctrl-Shift-P	Find/replace in projects
Alt-Shift-U	Find usages results
Alt-Shift-H	Turn off search result highlights
Ctrl-R	Rename
Ctrl-U, then U	Convert selection to uppercase
Ctrl-U, then L	Convert selection to lowercase
Ctrl-U, then S	Toggle case of selection
Alt-Shift-V	Paste formatted

### Opening and Toggling Between Views

Ctrl-Tab (Ctrl-`)	Toggle between open documents
Shift-Escape	Maximize window (toggle)
Ctrl-F4/Ctrl-W	Close currently selected window
Ctrl-Shift-F4	Close all windows
Shift-F10	Open contextual menu
Alt-Shift-D	Undock window

### Navigating through Source Code

Ctrl-O/Alt-Shift-O	Go to type/file
Ctrl-Shift-T	Go to JUnit test
Alt-O	Go to source
Ctrl-B	Go to declaration
Ctrl-G	Go to line
Ctrl-Shift-M	Toggle add/remove bookmark
Ctrl-Shift-Period/Comma	Next/previous bookmark
Ctrl-Period/Comma	Next/previous usage/compile error
Ctrl-Shift-1/2/3	Select in Projects/Files/Favorites
Ctrl-[	Move caret to matching bracket
Ctrl-K/Ctrl-Shift K	Next/previous word match
Alt-Left/Alt-Right/Ctrl-Q	Go backward/forward/to last edit

### Compiling, Testing, and Running

F9	Compile package/ file
F11	Build main project
Shift-F11	Clean & build main project
Ctrl-Q	Set request parameters
Ctrl-Shift-U	Create JUnit test
Ctrl-F6/Alt-F6	Run JUnit test on file/project
F6/Shift-F6	Run main project/file

### Debugging

Ctrl-F5	Start debugging main project
Ctrl-Shift-F5	Start debugging current file
Ctrl-Shift-F6	Start debugging test for file (JUnit)
Shift-F5/F5	Stop/Continue debugging session
F4	Run to cursor location in file
F7/F8	Step into/over
Ctrl-F7	Step out
Ctrl-Alt-Up	Go to called method
Ctrl-Alt-Down	Go to calling method
Ctrl-F9	Evaluate expression
Ctrl-F8	Toggle breakpoint
Ctrl-Shift-F8	New breakpoint
Ctrl-Shift-F7	New watch

### Coding in Java

Alt-Insert	Generate code
Ctrl-Shift-I	Fix all class imports
Alt-Shift-I	Fix selected class's import
Alt-Shift-F	Format selection
Alt-Shift Left/Right/Up/Down	Shift lines left/right/up/down
Ctrl-Shift-Up/D	Copy lines up/down
Ctrl/Alt-F12	Inspect members/hierarchy
Ctrl-/	Add/remove comment lines
Ctrl-E	Delete current line

## Keyboard Shortcuts & Code Templates, continued

### Refactoring

This table provides short descriptions of the refactoring operations that are available in the IDE, mostly from under the **Refactoring** menu and within the Java editor itself, when you right-click within it.

Refactoring Operation	Description
<b>Rename</b>	Enables you to change the name of a class, variable, or method to something more meaningful. In addition, it updates all source code in your project to reference the element by its new name.
<b>Introduce Variable, Constant, Field, or Method</b>	Enables you to generate a statement based on the selected code and replace that block of code with a call to the statement.
<b>Change Method Parameters</b>	Enables you to add parameters to a method and change the access modifier.
<b>Encapsulate Fields</b>	Generates a getter method and a setter method for a field and optionally updates all referencing code to access the field using the getter and setter methods.
<b>Pull Up</b>	Moves methods and fields to a class that their current class inherits from.
<b>Push Down</b>	Moves inner classes, methods, and fields to all subclasses of their current class.
<b>Move Class</b>	Moves a class to another package or into another class. In addition, all source code in your project is updated to reference the class in its new location.
<b>Copy Class</b>	Copies a class to the same or a different package.
<b>Move Inner to Outer Level</b>	Moves an inner class one level up in hierarchy.
<b>Convert Anonymous Class to Inner</b>	Converts an anonymous class to an inner class that contains a name and constructor. The anonymous inner class is replaced with a call to the new inner class.
<b>Extract Interface</b>	Creates a new interface from the selected public non-static methods in a class or interface.
<b>Extract Superclass</b>	Creates a new abstract class, changes the current class to extend the new class, and moves the selected methods and fields to the new class.
<b>Use Supertype Where Possible</b>	Changes code that references the selected class (or other type) to instead use a supertype of that type.
<b>Safely Delete</b>	Checks for references to a code element and then automatically deletes that element if no other code references it.

When typing in the Source Editor, you can generate the text in the right-column of the following list by typing the abbreviation that is listed in the left-column and then pressing Tab.

### Java Editor Code Templates

<b>En</b>	Enumeration
<b>Ex</b>	Exception
<b>Ob</b>	Object
<b>Psf</b>	public static final
<b>Psfb</b>	public static final boolean
<b>Psfi</b>	public static final int
<b>Psfs</b>	public static final String

## Java Editor Code Templates, continued

<b>St</b>	String
<b>ab</b>	abstract
<b>bo</b>	boolean
<b>br</b>	break
<b>ca</b>	catch (
<b>cl</b>	class
<b>cn</b>	continue
<b>df</b>	default:
<b>dowhile</b>	do { } while ( <i>condition</i> );
<b>eq</b>	equals
<b>ex</b>	extends
<b>fa</b>	false
<b>fi</b>	final
<b>fl</b>	float
<b>forc</b>	for (Iterator <i>it</i> = <i>collection.iterator</i> (); <i>it.hasNext</i> ();) { <i>Object elem</i> = ( <i>Object</i> ) <i>it.next</i> (); }
<b>fore</b>	for ( <i>Object elem</i> : <i>iterable</i> ) { }
<b>fori</b>	for (int <i>i</i> = 0; <i>i</i> < <i>arr.length</i> ; <i>i</i> ++) { }
<b>fy</b>	finally
<b>ie</b>	interface
<b>ifelse</b>	if ( <i>condition</i> ){}else { }
<b>im</b>	implements
<b>iof</b>	instanceof
<b>ir</b>	import
<b>le</b>	length
<b>newo</b>	<i>Object name</i> = new <i>Object</i> ( <i>args</i> );
<b>pe</b>	protected
<b>pr</b>	private
<b>psf</b>	private static final
<b>psfb</b>	private static final boolean
<b>psfi</b>	private static final int
<b>psfs</b>	private static final String
<b>pst</b>	printStackTrace();
<b>psvm</b>	public static void main(String[] <i>args</i> ){ }
<b>pu</b>	public
<b>re</b>	return
<b>serr</b>	System.err.println (" ");
<b>sout</b>	System.out.println (" ");
<b>st</b>	static
<b>sw</b>	switch (
<b>sy</b>	synchronized
<b>tds</b>	Thread.dumpStack();
<b>th</b>	throws
<b>trycatch</b>	try {} catch (Exception <i>e</i> ) {}
<b>tw</b>	throw
<b>twnew</b>	throw new
<b>wh</b>	while (
<b>whileit</b>	while ( <i>it.hasNext</i> ()) { <i>Object elem</i> = ( <i>Object</i> ) <i>it.next</i> (); }

## Mac OS Keyboard Shortcuts

In most cases, working with the IDE on the Mac is no different from working on other operating systems. Two significant differences do exist, however. Firstly, the Options window on the Mac is found under **NetBeans > Preferences**. Secondly, the About box is under **NetBeans > About**.

### Scrolling and Selecting

Keys	Action
Cmd-[	Moves the insertion point to the highlighted matching bracket. Note that this shortcut only works when the insertion point is located immediately after the opening bracket.
Cmd-Shift-[	Selects the block between a pair of brackets. Note that this shortcut only works when the insertion point is located immediately after either the opening or closing bracket.
Ctrl-G	Jumps to any specified line.
Cmd-A	Selects all text in the file.

### Code Folding

Keys	Action
Cmd-Minus (-)	Collapses the block of code in which the insertion point is currently located.
Cmd-Plus (+)	Expands the block of code which is adjacent to the insertion point.
Cmd-Shift-Minus (-)	Collapses all blocks of code in the current file.
Cmd-Shift-Plus (+)	Expands all blocks of code in the current file.

### Cutting, Copying, Pasting, and Deleting Text

Keys	Action
Cmd-Z	Undo. Reverses a series of editor actions one at a time (excluding Save).
Cmd-Y	Redo. Reverses a series of Undo commands one at a time.
Cmd-X	Cut. Deletes the current selection and places it on the clipboard.
Cmd-C	Copy. Copies the current selection to the clipboard.
Cmd-V	Paste. Pastes the contents of the clipboard at the insert point.
Backspace Delete	Deletes the current selection.
Cmd-E	Deletes the current line.
Cmd-K	Copies the word preceding the insertion point and then pastes it after the insertion point (the insertion point must be in the whitespace preceding or following a word). Press K multiple times to cycle through preceding words in succession.
Cmd-Shift-K	Copies the word following the insertion point and pastes it at the insertion point (the insertion point must be located in the whitespace preceding or following a word.) Press L multiple times to cycle through consecutive following words.

### To Change Default Settings:

1. Choose **Tools > Options** from the main menu.
2. For **code templates**, select Editor and click the Code Templates tab. Here you can also change the **expansion** key, from Tab to something else.
3. For **keyboard shortcuts**, select Keymap and choose a profile from the Profile drop-down list.

## 10 HANDY EDITOR SHORTCUTS

The following are some of the many cool NetBeans IDE 6.0 keyboard shortcuts that no programmer will be able to do without, once you know they're there. Not all of these are new in 6.0, some were there before, but deserve to be highlighted because often they're overlooked.

## 10 Handy Editor Shortcuts, continued

1. **Move/copy up/down.** Press Ctrl-Shift-Up and the current selection is copied to the lines right above the current selection. Press Alt instead of Ctrl and it is moved instead of copied. Press Down instead of Up and the lines of code will be copied below the current selection, as below:

```
private void closeButtonActionPerformed(java.  
    setVisible(false);  
    dispose();  
}  
private void closeButtonActionPerformed(java.  
    setVisible(false);  
    dispose();  
}
```

2. **Capture inner/outer syntactic element.** Each time you press Alt-Shift-Period, the selection expands to a successively wider syntactic element. For example, below one statement was selected, the key combination Alt-Shift-Period was pressed, and then the complete content of the surrounding block statement was shown to be selected. The selection expands from the current statement to surrounding block statements to the surrounding method and, from there, to the surrounding class and further. To successively narrow the selection, press Alt-Shift-Comma, instead of Alt-Shift-Period.

```
/** Creates new form About */  
public About(JFrame parent) {  
    super(parent,true);  
    initComponents();  
    pack();  
    Rectangle parentBounds = parent.getBounds();  
    Dimension size = getSize();  
    // Center in the parent  
    int x = Math.max(0, parentBounds.x + (parentB  
    int y = Math.max(0, parentBounds.y + (parentB  
    setLocation(new Point(x, y));  
}
```

3. **Generate code skeletons.** Whenever you want to generate commonly needed pieces of code, such as constructors, getters, and setters, simply click Alt-Insert, and a small popup appears with a list of items from which you can select:

```
import java.awt.*;  
"oylomprich",  
"i Generate  
"i Constructor...  
"t Getter...  
"t Delegate Method...  
"e Override Method...  
"amanegemtn",  
"aminunaler".
```

4. **Inplace rename.** If you want to change a variable, method, or other item, of which more than one are used in your code, press Ctrl-R, and you will see that all instances of the item turn blue at the same time, as shown below. Now, when you change the selected item, all the other instances change at the same time, until you press Escape, at which point the inplace rename mode comes to an end.

```
/** Creates new form About */  
public About(JFrame parent) {  
    super(parent,true);  
    initComponents();  
    pack();  
    Rectangle parentBounds = parent.ge  
    Dimension size = getSize();  
    // Center in the parent
```

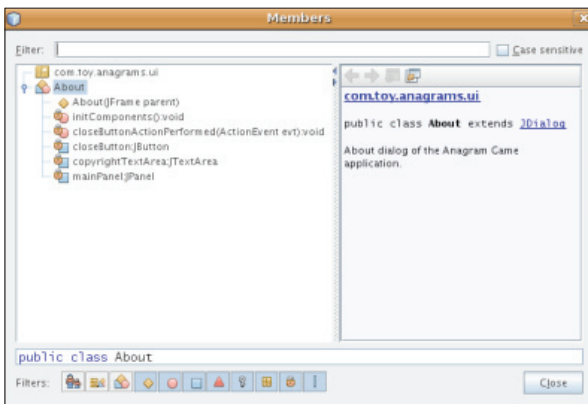
## 10 Handy Editor Shortcuts, continued

**5. Add/Remove comment lines.** Select one or more lines, press Ctrl-/ and then the selected lines are commented out, as shown below. Press the same keys again and the commented lines will no longer be commented. This was, of course, also possible in previous releases, but previously there were two different keyboard shortcuts, one for commenting and one for uncommenting. Now that they have been combined into one shortcut, you can quickly toggle between comment and uncomment, which makes this activity much faster and more efficient.

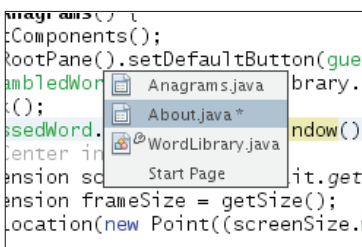
```

return word at that index in its standard
// public static String getScrambledWord(int
// return SCRAMBLED_WORD_LIST[idx];
    
```

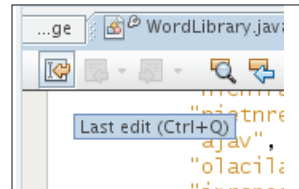
**6. Inspect members/hierarchy.** Both the members of the current class as well as its hierarchy can be displayed and then filtered. Press Alt-F12 and the ancestors of the current file are shown. On the other hand, if you press Ctrl-F12, the current file's members are displayed, as shown here:



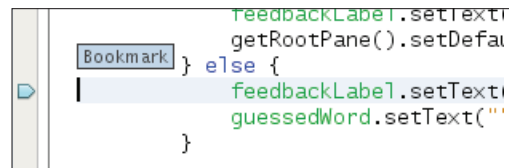
**7. Switch between documents.** When you have multiple documents open at the same time, press Ctrl and Tab, and then a small popup appears. You can scroll through the popup, which displays all the open documents, and then choose the document that you want to open:



**8. Jump to last edit.** Often, you find yourself in a situation where you have edited in one document, while you currently find yourself in a completely different document. How do you get back to the place where you were last editing your code? That is likely to be the place where you want to get back to, in order to do some more editing. Now, whenever you click Ctrl-Q, the last edited document is found, and the cursor lands on the line where the last edit took place. Alternatively, you can click the button shown below, in the top left corner of the Source Editor:



**9. Bookmarks.** When you press Ctrl-Shift-M, the current line is "bookmarked". What this means is that you can later quickly cycle back/forward to it (with Ctrl-Shift-Period and Ctrl-Shift-Comma). The bookmarked line gets a small icon in the left sidebar, as shown below, until you press Ctrl-Shift-M on the line again, to remove the bookmark:



**10. Highlight exit points.** Place the cursor on the return type and you will immediately see all exit points highlighted:

```

@Override
public boolean accept(File f) {
    if (f.isDirectory()) {
        return true;
    }
    String fileName = f.getName();
    int i = fileName.lastIndexOf('.');
    if ((i > 0) && (i < (fileName.length() - 1))) {
        String fileExt = fileName.substring(i);
        if ("txt".equalsIgnoreCase(fileExt)) {
            return true;
        }
    }
    return false;
}
    
```

## QUICK OPTIONS WINDOW OVERVIEW

The Options window lets you customize NetBeans IDE in a number of ways. Most people don't know how much can be customized there. The table below provides an overview for 6.1 only.

General	Sets the IDE-wide browser and the proxy settings.	<ul style="list-style-type: none"> <li>Web Browser</li> <li>Proxy Settings</li> </ul>
Editor	Sets the editor-specific options, specifically those relating to code folding, code completion, camel case behavior, indentation, code templates, and macros.	<ul style="list-style-type: none"> <li>Code Folding</li> <li>Code Completion</li> <li>Camel Case Behavior</li> <li>Indentation</li> <li>Code Templates</li> <li>Macros</li> </ul>



**Source URL:**

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

### Quick Options Window Overview, continued

<b>Fonts &amp; Colors</b>	Sets the fonts and colors for syntax, highlighting, annotations, and diff viewer.	<ul style="list-style-type: none"> <li>▪ Syntax</li> <li>▪ Highlighting</li> <li>▪ Annotations</li> <li>▪ Diff</li> </ul>
<b>Keymap</b>	Sets the keyboard profile to be used throughout the IDE. By default, profiles are provided for NetBeans, Eclipse, and Emacs. A legacy profile is also provided, for NetBeans 5.5 keyboard shortcuts, which were radically rewritten in NetBeans IDE 6.0.	<ul style="list-style-type: none"> <li>▪ NetBeans Profile</li> <li>▪ Eclipse Profile</li> <li>▪ Emacs Profile</li> <li>▪ NetBeans 5.5 Profile</li> </ul>
<b>Miscellaneous</b>	Sets the options for Ant processing, appearance, diffing, the Matisse GUI Builder, JavaScript, Profiler, ToDo Tasks, and Versioning.	<ul style="list-style-type: none"> <li>▪ Ant</li> <li>▪ Appearance</li> <li>▪ Diff</li> <li>▪ GUI Builder</li> <li>▪ Profiler</li> <li>▪ ToDo Tasks</li> <li>▪ JavaScript</li> <li>▪ Versioning</li> </ul>

### RESOURCES

Resource	URL
<b>NetBeans DZone Community</b>	<a href="http://netbeans.dzone.com/">http://netbeans.dzone.com/</a>
<b>NetBeans Tutorials</b>	<a href="http://www.netbeans.org/kb/index.html">http://www.netbeans.org/kb/index.html</a>
<b>NetBeans Video Tutorials</b>	<a href="http://www.netbeans.org/kb/60/screencasts.html">http://www.netbeans.org/kb/60/screencasts.html</a>
<b>NetBeans Blogs</b>	<a href="http://planetnetbeans.org/">http://planetnetbeans.org/</a>
<b>NetBeans TV</b>	<a href="http://netbeans.tv/">http://netbeans.tv/</a>
<b>NetBeans Weekly Newsletter:</b>	<a href="http://www.netbeans.org/community/news/ newsletter/latest.html">http://www.netbeans.org/community/news/ newsletter/latest.html</a>

Thanks to the following people who kindly gave of their time and expertise in reviewing this refcard: Adam Bien, Tony Kohar, Varun Nischal, Kristian Rink, and Tom Wheeler.

### ABOUT THE AUTHORS

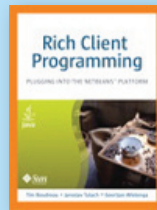


**Geertjan Wielenga** is the NetBeans technical writer responsible for documentation related to the NetBeans Java editor. He is co-author of the book *Rich Client Programming: Plugging into the NetBeans Platform*. He is known for his popular blog at <http://blogs.sun.com/geertjan>, as well as for his role as a Zone Leader at [Javalobby](http://Javalobby.com).

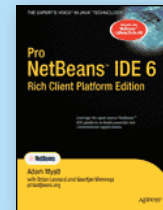


**Patrick Keegan** has been writing about the NetBeans IDE for over 9 years. In addition to writing help and tutorials, he is co-author of the *NetBeans IDE Field Guide* and has contributed to other books on NetBeans and Java.

### RECOMMENDED BOOKS



*Rich Client Programming* will help you get started with NetBeans module development, master NetBeans' key APIs, and learn proven techniques for building reliable desktop software.



*Pro NetBeans IDE 6 Rich Client Platform Edition* focuses on the new features of NetBeans 6 as well as what has changed since NetBeans 5.5, empowering you to be a more effective and productive developer.

### BUY NOW

[books.dzone.com/books/richclientprog](http://books.dzone.com/books/richclientprog)  
[books.dzone.com/books/pronetbeans](http://books.dzone.com/books/pronetbeans)

## Get More FREE Refcardz. Visit [refcardz.com](http://refcardz.com) now!

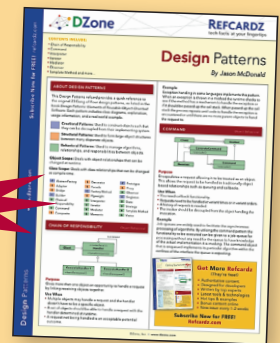
#### Upcoming Refcardz:

- Using XML in Java
- Core Mule 2
- Getting Started with Equinox and OSGi
- SOA Patterns
- Getting Started with EMF

#### Available:

- Getting Started with Hibernate Search
- Core Seam
- Essential Ruby
- Essential MySQL
- JUnit and EasyMock
- Getting Started with MyEclipse
- Spring Annotations
- Core Java
- Core CSS: Part II
- PHP
- Getting Started with JPA
- JavaServer Faces
- Core CSS: Part I
- Struts2
- Core .NET
- Very First Steps in Flex
- C#
- Groovy
- NetBeans IDE 6.1 Java Editor

Visit [refcardz.com](http://refcardz.com) for a complete listing of available Refcardz.



**Design Patterns**  
Published June 2008



DZone communities deliver over 4 million pages each month to more than 1.7 million software developers, architects and decision makers. DZone offers something for everyone, including news, tutorials, cheatsheets, blogs, feature articles, source code and more. **"DZone is a developer's dream,"** says PC Magazine.

DZone, Inc.  
1251 NW Maynard  
Cary, NC 27513  
888.678.0399  
919.678.0300  
**Refcardz Feedback Welcome**  
[refcardz@dzone.com](mailto:refcardz@dzone.com)  
**Sponsorship Opportunities**  
[sales@dzone.com](mailto:sales@dzone.com)



\$7.95