NetBeans Platform
What is it?
How do I use it?
And why should I want to?

Geertjan Wielenga
http://blogs.sun.com/geertjan
Sun Microsystems
Source Talk Tage, 27 September, 2006
Agenda

• Introduction
• 1. Why?
• 2. What?
• 3. How?
• 4. When?
• Conclusion
Open Source Java Directory

The Open Source Java Directory is maintained by Steve Mallett, creator of OSDir.com.

We welcome and encourage readers to add new product titles to the directory. All submissions go through an editorial review before being posted to the site.

You'll need to be registered with the O'Reilly Network to create or update entries.

- Not yet registered? You can do so here.
- Need to update a product entry you previously created? Login to do so here.
- Need help? Email help@oreillynet.com

What's New:

- GGZ-Java
- JTrac
- Dao-zero: implement your DAO interfaces
- DualRpc easy bidirectional RPC for client to server

Browse our directory:

<table>
<thead>
<tr>
<th>Title</th>
<th>Description</th>
<th>License</th>
<th>Update</th>
</tr>
</thead>
<tbody>
<tr>
<td>ActiveMQ</td>
<td>ActiveMQ is a fast open source JMS 1.</td>
<td>The Apache Software License</td>
<td>Nov 26, 2005</td>
</tr>
<tr>
<td>Activity Manager</td>
<td>Activity Manager is a project management tool that is simple to use, lightweight, and very efficient and customizable.</td>
<td>BSD License (BSD)</td>
<td>Apr 4, 2011</td>
</tr>
<tr>
<td>Application</td>
<td>Description</td>
<td>License</td>
<td>Date</td>
</tr>
<tr>
<td>-------------</td>
<td>-------------</td>
<td>------------</td>
<td>------------</td>
</tr>
<tr>
<td>IBM Jikes</td>
<td>Jikes is a compiler that translates Java source files as defined in The Java Language Specification into the bytecoded instruction set and binary format defined in The Java Virtual Machine Specification.</td>
<td>IBM Public License</td>
<td>Sep 18, 2002</td>
</tr>
<tr>
<td>InfoSapient</td>
<td>InfoSapient is an Open Source, Common License 0.5, Java 1.2 and above, program for the easy expression, execution and maintenance of business rules within a company.</td>
<td>IBM Public License</td>
<td>Oct 8, 2002</td>
</tr>
<tr>
<td>iSQL</td>
<td>iSQL(IndependentSQL)-Viewer is a JDBC 2.0-compliant application that is designed to exploit JDBC Features for all compliant drivers.</td>
<td>Mozilla Public License (MPL)</td>
<td>Oct 8, 2002</td>
</tr>
<tr>
<td>iText</td>
<td>iText is a library that allows you to generate PDF files on the fly.</td>
<td>GNU Lesser General Public License (LGPL)</td>
<td>Feb 14, 2002</td>
</tr>
<tr>
<td>ITracker: Java Issue Tracking Software</td>
<td>ITracker is a Java J2EE issue tracking system designed to support multiple projects with independent user bases.</td>
<td>GNU General Public License (GPL)</td>
<td>Jun 5, 2002</td>
</tr>
<tr>
<td>JacORB</td>
<td>Java implementation of the OMG’s CORBA standard.</td>
<td>GNU Lesser General Public License (LGPL)</td>
<td>Aug 6, 2002</td>
</tr>
</tbody>
</table>
Welcome
Download
Programming
JFugue
JFugue
JavaDoc
Interesting
Topics
Projects

Latest News
Coming Soon: JFugue 3.0! (For now, please enjoy JFugue 2.1)

JFugue is a Java API for music programming. It makes music programming incredibly easy, and it's great for applications in which music is generated at run-time: algorithmic or evolutionary music, music editors, jazz improvisers, and more!
Main Features

- Music is easy to program, or to generate, with Music Strings
- Patterns allow musical segments to be added and recombinined
- Pattern Transformers permit dynamically changing a pattern of music
- Music can be played at runtime, or saved in MIDI files
- Lets you play with music without having to know the guts of MIDI

To program music, you use music strings to specify the music you want to play. To play a C note, you basically say \texttt{play("C")};. You can use JFugue's Music Strings to specify additional effects - instrument changes, multiple voices, tempo, and MIDI controller events. You never have to deal with low-level MIDI messages to use JFugue.
play():

    Player player = new Player();
    Pattern pattern = new Pattern(string);
    player.play(pattern);

save():

    Player player = new Player();
    Pattern pattern = new Pattern(string);
    player.save(pattern, "Myfile.midi");
string = C5q D5q E5q C5q
I[Cello] E5q F5q G5h
I[Trumpet] G5i A5i G5i F5i
E5q C5q I[Piano] C5q G4q
C5h .C5q D5q E5q C5q
I[Cello] E5q F5q G5h
I[Trumpet] G5i A5i G5i F5i
E5q C5q I[Piano] C5q G4q
C5h C5q D5q E5q C5q
I[Cello] E5q F5q G5h
I[Trumpet] G5i A5i G5i F5i
E5q C5q I[Piano] C5q G4q
C5h
Demo!
Join the team!

https://nbjfuguesupport.dev.java.net/
SysUpTime: Network Management System
Nokia NetAct: Mobile Network Manager
Fiorano Studio: Lego Blocks for SOA
Streamsim IDE: Oil Flow Modeling
NetBeans IDE 5.0: Java Development
How: Does NetBeans IDE Help?

**Before** NetBeans IDE 5.0:
- *No* specific user interface support in NetBeans IDE
- *Little/no* documentation

**Since** NetBeans IDE 5.0:
- Wizards
- Templates
- Specific enhancements/support for Platform Apps
- A *lot* of API documentation, tutorials, blogs, samples
How: What's the Process?

1) Project templates: Basic source structure
2) File templates: Skeleton API implementations
3) Getting further:
   • Matisse!
   • Highly Advanced Source Editor
   • NetBeans API Javadoc
   • Tutorials
4) Menu items: Build, install, test, and debug...
5) User interface support: Branding & Distribution
Step 1: Project Templates

- Module Suite Project
  > Application structure.

- Module Project
  > Wrap functionality.

- Library Wrapper Module Project
  > Wrap external JAR files.
Step 2: NetBeans API Templates

- **Action wizard**
  - menu items, pop-up menus, toolbar buttons, keyboard shortcuts

- **Window Component wizard**
  - GUI custom components (Matisse!)
Step 2: NetBeans API Templates (Ct'd)

- **File Type** wizard
  > recognizing new file types

- **Wizard** wizard
  > wizard for generating wizard skeleton...

- **Project Template** wizard
  > bundles samples and templates

- **J2SE Library Descriptor** wizard
  > makes new JARs available to users in the IDE
Step 2: NetBeans API Templates (Ct'd)

• Options Panel **wizard**
  > extend the Options window (give your users options)

• JavaHelp Help Set **wizard**
  > wizard for setting up JavaHelp help set

• Update Center **wizard**
  > support your distribution process

• Module Installer **wizard**
  > _before_ the application begins, initialization functions
Step 3: Develop the Application

- Get to know the NetBeans APIs
- Use the NetBeans Source Editor
- Read the NetBeans API Javadoc

```java
public class HtmlBrowser extends JPanel {

    // Object that provides viewer for HTML pages.

    // If all you want to do is show some URL in the IDE's normal way, this is
    // overkill. Just use HtmlBrowser.URLDisplayer.showURL(java.net.URL)
    // instead. Using HtmlBrowser is appropriate mainly if you want to embed a
    // web browser in some other GUI component (if the user has selected an
    // external browser, this will fall back to a simple Swing renderer). Similarly
    // Impl (coming from a Factory) is the lower-level renderer itself (sans toolbar). 

    Summary: for client use, try URLDisplayer.showURL, or for more control or 
    where embedding is needed, create an HtmlBrowser. For provider use, create
```
Step 3: Develop the Application (Ct'd)

- [http://platform.netbeans.org/tutorials](http://platform.netbeans.org/tutorials)
  - Quick Start Guide
  - FAQ
  - Tutorials
Step 3: Develop the Application (Ct'd)

• Matisse! Matisse! Matisse!
Step 4: Build, Deploy, Test, & Debug

• Build Project
• Run Project
• Debug Project
• Create NBM File
• Install Project
• etc...
• etc...
Step 5: Distribution...

Advanced Options

Options
- Options
- Editing
- IDE Configuration
  - Look and Feel
  - Server and External Tool Settings
- System
  - File Types
  - Object Types
  - System Settings
  - Autoupdate Types
- Geertjan's Update Center
  - NetBeans Update Center
  - NetBeans Hotfix Update Center
  - Third-party Update Center
  - NetBeans Update Center Beta
- Update Center
- Suggestions Framework
- Print Settings
- Testing
  - JUnit Module Settings

Properties
- Server URL: http://blogs.sun.com/roller/resources/geertjan/updates_geertjan.xml
- Enabled: checked
- Time of last check: 1/14/06 7:26 PM
- Identifying Name: Geertjan's Update Center

Available Updates and New Modules:
- Editor Extension Samples
  - Actions API Sample (CookieAction)
  - Loaders API Sample
  - Palette API Sample (Simple)
  - Palette API Sample (Complex)
  - Text API Sample
  - Editor Module Library API Sample
  - CodeFoldingSupport
  - MultiView API Sample
- Miscellaneous Extension Samples
  - Actions API Sample (CallableSystemAction)
  - Options API Sample
  - JavaHelp API Sample
  - Matisse in Action Samples
  - Customers In Matisse Sample
- Fully-Functioning NetBeans IDE Extensions
  - HelpForJavaKeywords

Server URL
URL of Autoupdate Server
...but first... branding...
ZIP file generation (plus launcher)
...or a JNLP application!

- JNLP applications can be delivered and started over the web.
- With one menu item in the IDE, you can package your application as a JNLP application, for easy distribution to your end users.
Show me! Demo...

1. Create a rich-client application skeleton (the core).
2. Bundle the JFugue API.
3. Add module for functionality (with Matisse).
   - JTextArea
   - JButton: Play
   - JButton: Save
4. Call “Play” and “Save” methods in the JFugue API.
5. Create a distribution.
Summary

• Developing on the NetBeans Platform since 5.0
  > Wizards and templates
  > Matisse for intuitive GUI design
  > NetBeans API JavaDoc
  > Samples and Tutorials

• Need help?
  > dev@openide.netbeans.org
  > http://blogs.sun.com/geertjan
Questions & Answers
jdkhome="C:\Program Files\Java\jdk1.5.0_06"