Getting Started with the NetBeans Platform

Geertjan Wielenga
http://blogs.sun.com/geertjan
Agenda

- **Goals**
  1. Generic Desktop Framework
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit
- Demo: Porting to NetBeans Platform
- Resources
Goals

- Meet the NetBeans Platform
- Understand its problem domain
- Get an overview of the main APIs
- See a demo
- Learn what to do next
Agenda

- Goals
  1. **Generic Desktop Framework**
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit
- Demo: Porting to NetBeans Platform
- Resources
NetBeans Platform
Nuance Voice-XML
Fiorano Studio
Nokia: Mobile Network
Sketsa SVG Editor
VisualVM
AIOTrade
NetBeans IDE
MC4J JMX Console
JFugue Music Notepad
StudioSL: Oil Flow
Agenda

• Goals
  1. Generic Desktop Framework
  2. **Infrastructural Plumbing**
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit

• Demo: Porting to NetBeans Platform

• Resources
Maybe this is your code:
Maybe this is you:
This should be you...

domain expert knowledge
Application “Plumbing”

● Windowing/docking system
● Architecture
● Lifecycle management
● Persistence
● Data management
● Consistent look & feel
● Distribution/update mechanism
Demo
Agenda

• Goals
  1. Generic Desktop Framework
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit

• Demo: Porting to NetBeans Platform
• Resources
Javadoc

NetBeans API List

Current Development Version

10 Mar 2008

This document provides a list of NetBeans APIs with a short description of what they are used for, and a table describing different types of interfaces (see What is an API? to understand why we list DTDs, file formats, etc.) and with a stability category (stable and official, under development, deprecated, friend or private; see API life-cycle for more info). The aim is to provide as detailed a definition of NetBeans module external interfaces as possible and give other developers a chance to decide whether they want to depend on a particular API or not.

Some of these APIs are part of the NetBeans Platform (for example lookup, loaders, utilities, nodes, explorer, window systems, multiview, etc.), some of them are specific to NetBeans IDE (projects, javacore, diff, etc.) and some are not included in the release at all and are just provided for download (usually via autoupdate). Basically when building an application based on NetBeans one is free to choose the set of modules and their APIs to satisfy one’s needs.

This is a list of APIs for development version, if you want to see a list of APIs for a particular version, you may want to go to:

- 6.0 - JavaDoc as released for NetBeans IDE 6.0
Libraries

• Window System API
• Actions API
• Options Window API
• Many Editor APIs
• Visual Library API
• ...and many more
Ways to get started

● Javadoc
● Tutorials
● Samples
● Blog entries

Tip: Use them outside the NetBeans Platform
Agenda

• Goals
  1. Generic Desktop Framework
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit
• Demo: Porting to NetBeans Platform
• Resources
In addition to...

- JLabel
- JTextField
- JTextArea
- JTable
- ...etc...
TreeTableView
Visual Widgets
More Visual Widgets
More Visual Widgets
Drop Down Button Factory

```xml
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE filesystem PUBLIC "filesystem 1.1/EN" "http://"
<filesystem>

  <folder name="Actions">
    <folder name="MyCategory">
      <file name="org-bla-bla-bla-MyOneAction.instance"/>
      <file name="org-bla-bla-bla-MyTwoAction.instance"/>
    </folder>
    <folder name="MyOtherCategory">
      <file name="org-bla-bla-bla-MyOtherAction.instance"/>
    </folder>
  </folder>
</filesystem>
```
Tabbed Pane Factory
Agenda

● Goals
  1. Generic Desktop Framework
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. **NetBeans Platform Toolkit**

● Demo: Porting to NetBeans Platform

● Resources
Project wizards

Steps
1. Choose Project
2. ...

Choose Project

Categories:
- Java
  - NetBeans Modules
  - Samples

Projects:
- Module
- Module Suite
- Library Wrapper Module
- NetBeans Platform Application

Description:
Creates a new empty NetBeans Platform Application. This is an application skeleton, without any modules. It is the starting point for your own rich-client applications.
NetBeans Platform
Outline view
Project wizards

Steps
1. Choose Project
2. ...

Choose Project
Categories:
- Java
- NetBeans Modules
- Samples

Projects:
- Module
- Module Suite
- Library Wrapper Module
- NetBeans Platform Application

Description:
Creates a new empty NetBeans Module.
File wizards

Steps
1. Choose File Type
2. ...

Choose File Type
Project: VisualVM-JConsole

Categories:
- Module Development
  - Java
  - Swing GUI Forms
  - AWT GUI Forms
  - JUnit
  - XML
  - Other

File Types:
- Java SE Library Descriptor
- Language Support
- Action
- JavaHelp Help Set
- File Type
- Module Installer
- Options Panel
- Project Template
- Update Center
- Window Component
- Wizard

Description:
Use this template to add a new class library to the Library Manager of the user's IDE. Select a library you have already defined in your own IDE. You may include associated sources and/or Javadoc.
File wizards

- Actions API
- Datasystems API
- Module Install class
- Options Dialog API
- TopComponent class
- Wizard Descriptor class
Configuration
While developing...

...enhance the IDE
Dependencies
Dependencies

![Image of Add Module Dependency window]

- Filter: call
- Show Non-API Modules
- Module:
  - JPDA Debugger API
  - Task List API
  - Utilities API
  - MultiView Windows
- Description:
  - JPDA Debugger API
- Matching Filter Contents:
  - org.netbeans.api.debugger.jpda.CallStackFrame
  - org.netbeans.spi.debugger.jpda.SmartSteppingCallback

[Buttons: Show javadoc, OK, Cancel, Help]
## Contextual menus

<table>
<thead>
<tr>
<th>Menu</th>
<th>Submenu</th>
</tr>
</thead>
<tbody>
<tr>
<td>New</td>
<td></td>
</tr>
<tr>
<td>Build</td>
<td></td>
</tr>
<tr>
<td>Clean and Build</td>
<td></td>
</tr>
<tr>
<td>Clean</td>
<td></td>
</tr>
<tr>
<td>Run</td>
<td></td>
</tr>
<tr>
<td>Debug</td>
<td></td>
</tr>
<tr>
<td>Profile</td>
<td></td>
</tr>
<tr>
<td>Install/Reload in Target Platform</td>
<td></td>
</tr>
<tr>
<td>Install/Reload in Development IDE</td>
<td></td>
</tr>
<tr>
<td>Create NBM</td>
<td></td>
</tr>
<tr>
<td>Generate Javadoc</td>
<td></td>
</tr>
<tr>
<td>Generate Architecture Description</td>
<td></td>
</tr>
<tr>
<td>Set as Main Project</td>
<td></td>
</tr>
<tr>
<td>Open Required Projects</td>
<td></td>
</tr>
<tr>
<td>Close</td>
<td></td>
</tr>
<tr>
<td>Rename...</td>
<td></td>
</tr>
<tr>
<td>Move...</td>
<td></td>
</tr>
<tr>
<td>Copy...</td>
<td></td>
</tr>
<tr>
<td>Delete</td>
<td>Delete</td>
</tr>
<tr>
<td>Find...</td>
<td>Ctrl+F</td>
</tr>
<tr>
<td>CVS</td>
<td></td>
</tr>
<tr>
<td>Local History</td>
<td></td>
</tr>
<tr>
<td>Properties</td>
<td></td>
</tr>
</tbody>
</table>
Contextual menus

<table>
<thead>
<tr>
<th>New</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build All</td>
</tr>
<tr>
<td>Clean and Build All</td>
</tr>
<tr>
<td>Clean All</td>
</tr>
<tr>
<td>Run</td>
</tr>
<tr>
<td>Debug</td>
</tr>
<tr>
<td>Profile</td>
</tr>
<tr>
<td>Build ZIP Distribution</td>
</tr>
<tr>
<td>Build JNLP Application</td>
</tr>
<tr>
<td>Run JNLP Application</td>
</tr>
<tr>
<td>Debug JNLP Application</td>
</tr>
<tr>
<td>Build Mac OS X Application</td>
</tr>
<tr>
<td>Create NBM s</td>
</tr>
<tr>
<td>Set as Main Project</td>
</tr>
<tr>
<td>Open Required Projects</td>
</tr>
<tr>
<td>Close</td>
</tr>
<tr>
<td>Rename...</td>
</tr>
<tr>
<td>Move...</td>
</tr>
<tr>
<td>Delete</td>
</tr>
<tr>
<td>Delete</td>
</tr>
<tr>
<td>Find...</td>
</tr>
<tr>
<td>Ctrl+F</td>
</tr>
<tr>
<td>CVS</td>
</tr>
<tr>
<td>Local History</td>
</tr>
<tr>
<td>Properties</td>
</tr>
</tbody>
</table>
Samples

Steps
1. Choose Project
2. ...

Choose Project
Categories:
- Java
- NetBeans Modules
- Samples
  - Java
  - NetBeans Modules

Projects:
- FeedReader
- Paint Application

Description:
Illustrates how to use the NetBeans Platform to create your own application. See the FeedReader Tutorial for details.
Agenda

- Goals
  1. Generic Desktop Framework
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit
- Demo: Porting to the NetBeans Platform
- Resources
Demo Outline

1. Start with a simple application.
2. Compare what it is to what the NetBeans Platform has.
3. Create a new plugin project.
4. Move the useful parts of our old application to our new plugin.
5. Run it.
Demo
Porting Guide

NetBeans Platform Porting Tutorial

This tutorial demonstrates how to port a simple Swing application to the NetBeans Platform. Though the sample is simple, the basic concepts of "porting" an application to the NetBeans Platform will become clear. At the end, some general principles will be identified, based on the steps taken in the tutorial. Hopefully, they will be useful to you when porting your own Swing applications to the NetBeans Platform.

Before beginning this procedure, it makes sense to ask why one would want to do so in the first place. A typical Swing application consists of a user interface layer on top of a general framework. The general framework normally provides features dealing with an application's infrastructure, such as an application's menu bar, windowing system (also known as "Docking System"), and lifecycle management. Typically this framework is reused by many applications within the same company. The NetBeans Platform exists specifically to cater to these infrastructural concerns. You do not need to create these on your own for your own Swing applications. You can simply move the useful parts of your own application to the NetBeans Platform and then, from that point onwards, the NetBeans Platform will be the new underlying 'plumbing' layer of your application. You can then focus on the more interesting parts of your application, specifically, the user interface. This will speed up your development process and give you a consistent basis for all your applications.

In this tutorial, we will begin with the Anagram Game, which is a standard Swing application sample that is distributed with NetBeans IDE. We will, step by step, move it to the NetBeans Platform and then see the advantages and disadvantages of doing so.

Contents

- Installing the Software
- Getting the Anagram Game
- Levels of Compliance
  - Creating the Module Project Source Structure
  - Porting Level 0: Launchable
  - Porting Level 1: Integration
  - Porting Level 2: Use Case Support
  - Porting Level 3: Aligned
- Next Steps

For more information on working with NetBeans modules, see the NetBeans Development Project home on the NetBeans website. If you have questions, visit the NetBeans Developer FAQ or use the feedback link at the end of this page.

http://platform.netbeans.org/tutorials/60/nbm-porting-basic.html
Agenda

- Goals
  1. Generic Desktop Framework
  2. Infrastructural Plumbing
  3. Collection of Libraries
  4. Swing Extensions
  5. NetBeans Platform Toolkit
- Demo: Porting to the NetBeans Platform
- Resources
Resources

http://www.netbeans.org/kb/articles/books.html
NetBeans Modules and Rich-Client Applications Learning Trail

What is a NetBeans Module?
A NetBeans module, also known as 'plugin', is a group of Java classes that provides an application with a specific feature. By developing your own NetBeans modules, you can extend an application's functionality with new features. For example, you can write modules that make your favorite cutting-edge technologies available to the NetBeans IDE. Alternatively, you might miss some low-level functionality in the Source Editor and create a module to provide it. On a higher level, you can use the core of NetBeans as a platform on top of which you develop rich-client Swing applications, out of NetBeans modules. You can save a lot of development time by reusing features readily available in the platform.

This page provides links to some of the NetBeans documents and resources that can help you learn about NetBeans modules and rich-client applications with NetBeans IDE 6.0.

Getting Started
- NetBeans Module Quick Start
- NetBeans Platform Quick Start
- NetBeans Platform Porting Tutorial

Tutorials, Guides, and Demos
Applications Built on the NetBeans Platform
- Paint Application Tutorial
- Feed Reader Tutorial

NetBeans APIs for Selection Management
- Part 1: Selection Management Tutorial
- Part 2: Selection Management

platform
- NetBeans Platform Resources
  - NetBeans API Javadoc
  - Tutorials
  - Articles
  - Guidelines
  - FAQs
  - Use Cases
  - Plugin Portal
  - Mailing Lists

Tell Me How To...
- ...add a toolbar?
- ...integrate a new file type?
- ...part a Swing application?
- ...work with nodes?
- ...extend an editor?
- ...do something else?

Get Tips From The Experts
- Sardip Chitale
- Tom Wheeler
Welcome to the NetBeans Platform Certified Training

The NetBeans Platform is a mature open sourced Swing framework for building general rich client applications. The platform is a "generic application", that is, a runtime which can be used to develop desktop applications. NetBeans IDE is one example of this type of application, but there are many others.

One of the key distinctions of software built upon the NetBeans Platform is modularity: reuse in the large. Such software is designed as logical sets of macro-components which integrate through well-defined API contracts. Writing modular applications brings some enhancements to programming in standard Java, particularly in the ability to have Java classes which are only public to other classes within the archive they reside in. This has a number of beneficial effects on development: in particular, the ability to develop cleaner, simpler APIs by being able to fully conceal implementation from foreign code, while retaining type-safety.

This course covers developing on the NetBeans Platform from the ground-up. It covers two days in total. On the first day, we start with non-GUI applications: simply making use of the module system, which is the core runtime of the NetBeans Platform. Next, we cover the basic development patterns and commonly used APIs. The second day is dedicated to a tour of the reusable existing functionality built on the NetBeans Platform, descriptions of how to empower it further, and examples of real software applications consisting of multiple, decoupled modules.

Finally, after the two days outlined above, you will develop a functioning project consisting of one or more modules. Or you will extend an already existing project with new functionality.
NetBeans Certified Training: Linz, Austria

From 9 to 10 November, 2007, the Johannes Kepler Universitat Linz was the first to be introduced to the NetBeans Certified Engineering course. Over the two days, the full program was followed.

Here is a photo of the whole group, with the two trainers, Jaroslav Tulach and Geertjan Wielenga:

![Group Photo](image)

The course ended with the following assignments and suggestions:

- [LinzCourseExcercises](#)
- [Link to Tutorials You Can Write](#)
References

- dev@openide.netbeans.org
- users@edu.netbeans.org
- http://platform.netbeans.org
- http://plugins.netbeans.org/PluginPortal/
Summary

- Many on the NetBeans Platform
- NetBeans Platform gives plumbing
- Many APIs and libraries to help you
- NetBeans IDE provides tools
- Getting started is easy
- Mail list, tutorials, and training course