Using NetBeans IDE for Desktop Development

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

- Goals
- Design: Matisse GUI Builder
- Medium Applications: JSR-296 Tooling
- Large Applications: NetBeans Platform
- Conclusion
Goals

● Show that NetBeans IDE is the one-stop shop for all Swing desktop needs

● Ready out of the box, for applications of all sizes:
  ● Demo 1: Matisse GUI Builder
  ● Demo 2: Tooling for JSR-296
  ● Demo 3: NetBeans Platform
Agenda

- Goals
- **Design: Matisse GUI Builder**
- Medium Applications: JSR-296 Tooling
- Large Applications: NetBeans Platform
- Conclusion
GroupLayout

- Part of JDK 6
- Principles:
  - Independent dimensions
  - Hierarchical groups
- Designed to suit GUI Builder needs
“Matisse” GUI Builder

- Professional quality layout easily done
- Intuitive drag & drop interface
- Guidelines
- Resize, align, optimal spacing
- Automatically resizes when localized
Demo
Agenda

- Goals
- Design: Matisse GUI Builder
- Medium Applications: JSR-296 Tooling
- Large Applications: NetBeans Platform
- Conclusion
Motivation for JSR-296

```java
public static void main(String args[]) {
   // good luck!
}
```
JSR-296 Features

- Lifecycle support
- Resources
- Actions
- Tasks
- Session state
Lifecycle Support
(On your own)

import javax.swing.*;

public class HelloWorldSwing {

    public static void main(String[] args) {
        JFrame frame = new JFrame("HelloWorldSwing");
        final JLabel label = new JLabel("Hello World");
        frame.getContentPane().add(label);
        frame.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
        frame.pack();
        frame.setVisible(true);
    }
}


public class MyApp extends SingleFrameApplication {

   @Override protected void startup() {
      JLabel label = new JLabel("Hello World");
      show(label);
   }

   public static void main(String[] args) {
      Application.launch(MyApp.class, args);
   }
}
Demo of JSR-296

- Show all the other JSR-296 features in action.
- Integrate Flickr into a Swing desktop application based on JSR-296 using NetBeans tooling.
Agenda

● Goals
● Design: Matisse GUI Builder
● Medium Applications: JSR-296 Tooling
● Large Applications: NetBeans Platform
● Conclusion
NetBeans Platform

What is it?

1. Generic Desktop Framework
2. Infrastructural Plumbing
3. Collection of Libraries
4. NetBeans Platform Toolkit
Agenda

• Goals

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

• Goals
  1. Generic Desktop Framework
  2. *Infrastructural Plumbing*
  3. Collection of Libraries
  4. NetBeans Platform Toolkit
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. NetBeans Platform Toolkit
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, multitview, 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. NetBeans Platform Toolkit
Project wizards

Steps
1. Choose Project
2. ...

Choose Project
Categories:
- java
- NetBeans Modules
- Samples

Projects:
- Module
- Module Suite
- Library Wrapper Module

Description:
Creates a new empty NetBeans Module Suite.
NetBeans Platform
Outline view

- LayerRegistrationAction
  - NetBeansPlatform
    - Modules
    - Important Files
  - plugins
    - Modules
      - VisualVM-JConsole
      - VisualVM-MBeans
    - Important Files
  - VisualVM
    - Modules
      - VisualVM-AppUI
      - VisualVM-Core
      - VisualVM-Startup
    - Important Files
Project wizards

Steps
1. Choose Project
2. ...

Choose Project

Categories:
- java
- NetBeans Modules
- Samples

Projects:
- Module
- Module Suite
- Library Wrapper Module

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
```
String name = names[i];
JCheckBoxMenuItem item = new JCheckBoxMenuItem(name);
item.setName(name);
item.addActionListener(menuItemListener);
buttonGroup.addItem(item);
popup.addItem(item);
}

dropDownButton = DropDownButtonFactory.createDropDownButton(new ImageIcon(
    new BufferedImage(32, 32, BufferedImage.TYPE_BYTE_ARGB)),
    "Insert Layer Registration"
);
dropDownButton.addItemListener(new ItemListener() {
    public void itemStateChanged(ItemEvent e) {
        if (e.getStateChange() == ItemEvent.SELECTED) {
            /* show popup menu on dropdown button at position */
            popup.show(dropDownButton, 0, dropDownButton
        }
    }
```

Editor
While developing...

...enhance the IDE
Dependencies

Categories:
- Sources
- Libraries
  - Display
  - API Versioning
- Build
  - Compiling
  - Packaging

Java Platform: JDK 1.6 (Default)
NetBeans Platform: NetBeans IDE 6.0.1 ...

Module Dependencies:
- Actions APIs
- Editor Library 2
- File System API
- Nodes API
- Text API
- UI Utilities API
- Utilities API
- Window System API

Required Tokens:

Add... Remove... Edit...
Dependencies

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

Show javadoc

OK  Cancel  Help
Contextual menus

- New
- Build
  - Clean and Build
  - Clean
- Run
  - Debug
  - Profile
- Install/Reload in Target Platform
  - Install/Reload in Development IDE
  - Create NBM
- Generate Javadoc
  - Generate Architecture Description
- Set as Main Project
- Open Required Projects
- Close
- Rename...
  - Move...
  - Copy...
  - Delete
- Find...
  - Ctrl+F
- CVS
- Local History
- Properties
## 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.
Demo: Porting
Porting 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.
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
● Design: Matisse GUI Builder
● Medium Applications: JSR-296 Tooling
● Large Applications: NetBeans Platform
● Conclusion
Resources

- http://weblogs.java.net/blog/tpavek
- 
- 
- 
- 
- 
-
Conclusion

● Ready out of the box, for applications of all sizes:
  ● Demo 1: Matisse GUI Builder
  ● Demo 2: Tooling for JSR-296
  ● Demo 3: NetBeans Platform

● Showed that NetBeans IDE is the one-stop shop for all Swing desktop needs