

# **Building Professional Desktop Applications on the NetBeans Platform**

**Tom Wheeler**  
Object Computing Inc.

# Who Am I?

- Principal Software Engineer at OCI
- Active in NB community

# What You'll Learn...

- What is the platform
- Why you should use the platform
- How to use the platform
- Where to learn more

# What is the Platform?

- Pure Java
- Open source
- A great way to write desktop apps in Java
- Subject of a great new book

# Some Good Examples...

- Here are some platform apps...

# AIOTrade



# Music Notepad

The screenshot displays the JFugue Music NotePad application window. The title bar reads "JFugue Music NotePad". The menu bar includes "File", "Tools", "Window", and "Help". A toolbar with various icons is located below the menu bar. The main workspace shows two staves of musical notation in 4/4 time, with a treble clef and a key signature of one flat (B-flat). The first staff contains a sequence of notes: G4, A4, B4, C5, B4, A4, G4, F4, E4, D4, C4. The second staff contains a sequence of notes: G4, A4, B4, C5, B4, A4, G4, F4, E4, D4, C4, followed by a whole rest. To the right of the main workspace are two panels: "JFugue Commands" and "Instruments Window". The "JFugue Commands" panel lists several commands: F6s, G6s, A6q, D6q, G6q, and C5w. The "Instruments Window" panel shows a list of instruments under the "Piano" category: Acoustic Grand Piano, Bright Acoustic Piano, Electric Grand Piano, Honky-tonk Piano, Electric Piano 1, Electric Piano 2, and Harpsichord. At the bottom of the window, a status bar displays the message "Successfully played!".

# Mass Toolkit


The screenshot displays the Mass Toolkit software interface. The main window shows a "Mass Case MC\_102 Payload Loading Diagram" of an aircraft fuselage. The diagram illustrates the distribution of mass along the fuselage, with green bars representing fuel and red bars representing payload. Key components labeled include "W.W." (Wing Well), "WING BOX", and "WHEEL WELL". The diagram is oriented with "Y" on the vertical axis and "X" on the horizontal axis. Below the diagram are "Solve" and "ClearSolution" buttons. A green status message at the bottom right of the diagram area reads "Mass Case is SOLVED".

The left sidebar shows a project tree for "MassTKProject\_8" with the following structure:

- Fuel\_Management
  - FM\_1
  - FT\_1
  - FT\_10
  - FT\_2
  - FT\_20
- Mass\_Cases
  - MC\_101
  - MC\_102
- Payload\_Cases
- Payload\_Tables
- ConMass\_Set\_1

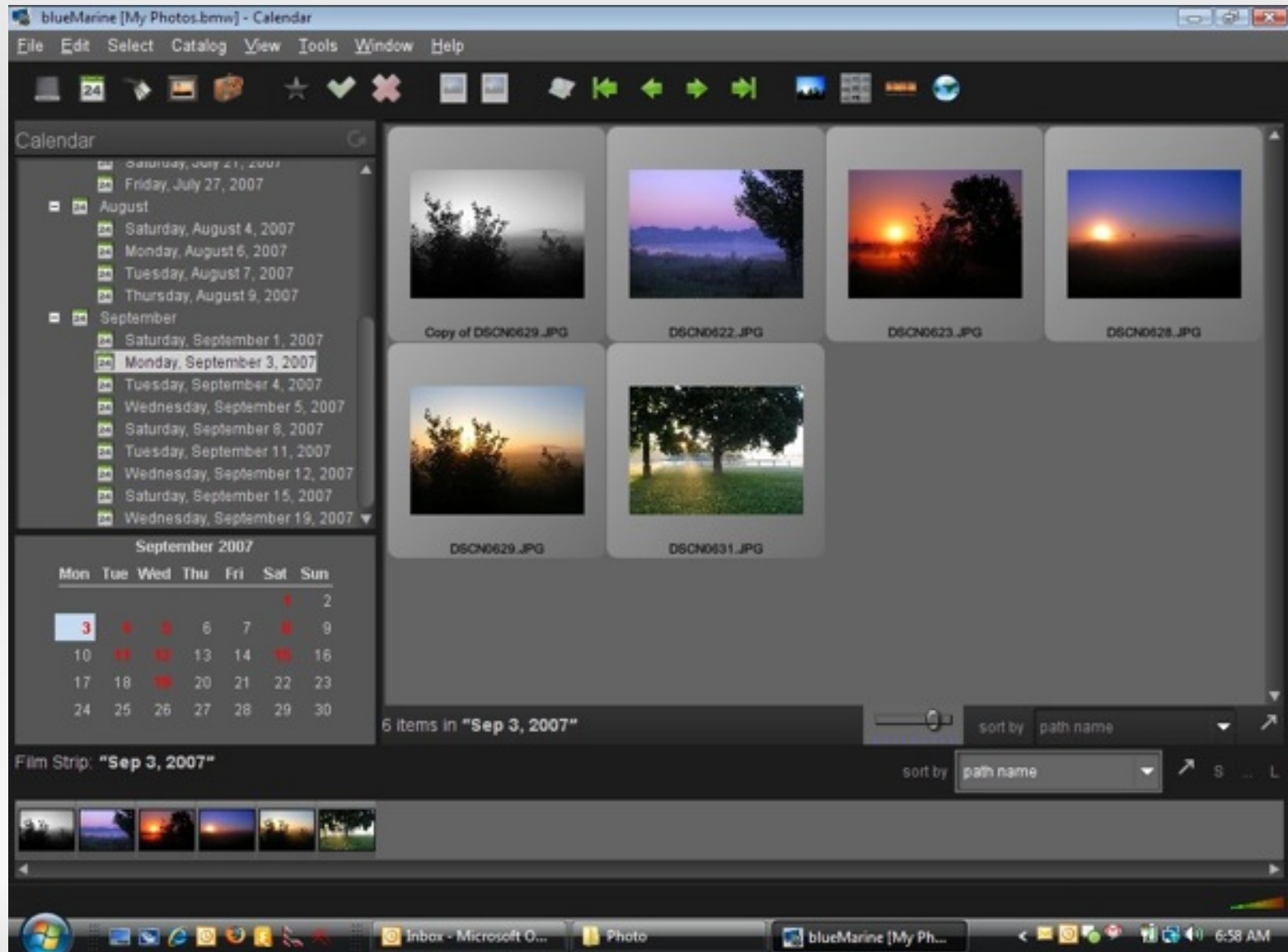
The top toolbar includes icons for file operations and a memory usage indicator showing "68.1/94.3MB". The Log Panel at the bottom displays the following activity:

```
06/04/07 13:47:38 : Opening Fuel Management Editor for FM_1
06/04/07 13:47:49 : Opening Fuel Table Editor for FT_10
06/04/07 13:48:11 : Opening Empty Vehicle Editor for ConMass_Set_1
06/04/07 13:53:39 : Queuing solution task for mass case MC_102
06/04/07 13:53:41 : Finished solving mass case MC_102
```

Boeing Proprietary 



# blueMarine



# Other Platforms

- JSR-296
- Eclipse RCP
- Spring Rich Client Platform
- JIDE Desktop App Framework

# Everything is a Platform

- How does the IDE relate to platform?

# Why Use a Framework?

- Frameworks are widespread for Web apps
  - But seldom used for Swing
  - There's no good reason for this
- The business logic is what counts

# Why Use the Platform?

- Saves time and money
- Encourages code reuse
- Professional, consistent appearance

# Platform Features

- Modules with dependency management
- AutoUpdate Feature
- Declarative UI
- Windowing system
- Nodes, Explorer and PropertySheet APIs

# Platform Features, Part II

- Integrated JavaHelp
- Flexible Filesystem implementation
- Deploy via JNLP
- Build harness (Ant scripts)
- Custom File Type/Project Types

# Platform Features, Part III

- Wizard framework
- Flexible filesystem implementation
- Graph Library (AKA Visual Library)
- Palette API
- Many handy utility classes
- Can use IDE and contrib modules
- Branding



# Modules

- Modules
  - One or more classes
  - Build scripts / configuration files
  - Can express dependencies on other modules
  - Two main types

# Suites

- Suites
  - A configuration item for containing modules
  - Contains
    - List of modules
    - Platform info
    - Build scripts and configuration files
    - Branding

# Harness

- You won't have to write build scripts
  - At least for the simple cases
- The harness is a set of Ant build scripts

# How to Design Modules

- Public class visibility between modules
- Granularity means future flexibility
- Modules should be as small as possible

# Porting to the Platform

- Keep it simple (at first)
  - Get rid of your `main()` method
  - Get rid of frames and dialogs
  - Focus on Actions and TopComponents
- Integrate with other NB APIs later

# Challenges

- Learning curve
  - Significant, but definitely worth the effort
- Testing
  - Testing platform apps can be tough
  - XTest is your friend!
    - Jemmy, Jelly Tools, etc.

# Get Started!

- Book: Rich Client Programming
  - Plugging into the NetBeans Platform
- Tutorials
- Wiki / FAQ
- Mailing lists
- Get involved
  - Share what you learn
  - Be a part of the community!

# Thanks for Listening

Any questions?