Building Performance Swing Applications

NetBeans 3.5 Case Study

Trung Duc Tran
Engineering Manager

Petr Nejedly
Member of Technical Staff

Sun Microsystems, Inc.

Presentation Goal

The techniques used to improve the UI responsiveness of NetBeans 3.5

Things they don't tell you in performance textbooks

BEGINNING

About NetBeans

- large IDE application written using Java/Swing
- runs on many OSes
- platform to build desktop apps
- very modular extensible architecture
- NetBeans 3.5 was released this Monday, June 9
- http://www.netbeans.org/

BEGINNING

Speakers' Qualifications

- Petr Nejedly has been working on performance of NetBeans for the last two years
- Trung Duc Tran manages Sun's NetBeans Platform engineering team, he led the performance action team during NetBeans 3.5 release cycle

BEGINNING

Agenda

- The problem
- What is UI responsiveness
- How to measure/profile responsiveness
- How to improve responsiveness
- Summary
- Q&A

BEGINNING

The Problem

- “nice features but too slow”
- “sluggish, can't keep up with me typing”
- “memory hog”
- “I need to buy a new machine to be able to use it”

UI Responsiveness != Raw Performance

- Performance: raw numbers
- UI responsiveness: perception, timely feedback
- Example: splash screen during startup
  - the app starts more slowly
  - but better user perception

MIDDLE

Design Principles for Responsive UI

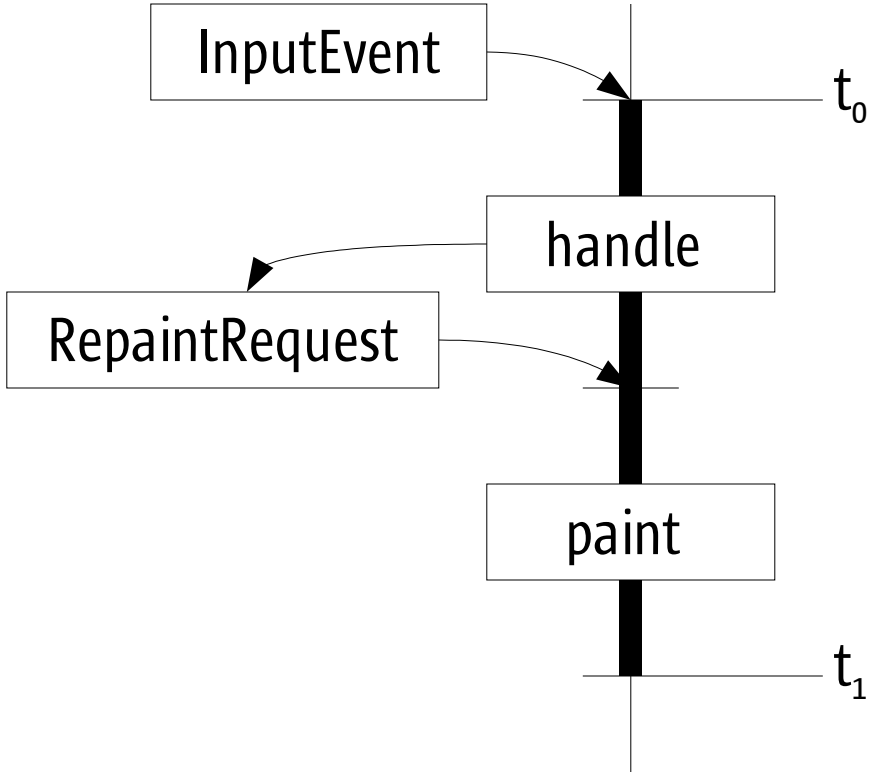
- timely feedback
- acknowledge user inputs immediately
- three time constants:
  - 100 msec: eye-hand coordination tasks (mouse, typing)
  - 1 sec: opening a new window/dialog, reformatting source files,...
  - 10 sec: unbroken concentration on a task, “unit task”, progress indicator is necessary
- Reference:
  Jeff Johnson:“GUI Bloopers” book

MIDDLE

Measure/Profile Responsiveness

- needs accuracy of 10 msec
  System.currentTimeMillis() is fine
- start: mouse/keyboard event
- stop: the last paint event
- technique:
  patching AWT/Swing code
  application code remains unchanged
- profiler start/stop API

The Event Flow




```
> java -Xbootclasspath/p:/path/to/uiresp.jar Main
```

Patch download: http://performance.netbeans.org/

Demo

Identified Problems in NetBeans

- slow main menus, context menus
- slow expansion, navigation in the explorer tree
- the missing wait cursor, no progress indicator
- frequently used dialogs are slow
- opening files is slow
- temporary “freeze” during typing
- first use is especially slow

http://performance.netbeans.org/responsiveness/issues.html

MIDDLE

Techniques Used in NetBeans 3.5

- standard performance tuning techniques
- automatic wait cursor
- warm-up
- lazy initialization
- progressive rendering
- UI redesign

MIDDLE

Automatic Wait Cursor

- all actions are called through the ActionManager
- easy to wrap action invocation inside show/hideWaitCursor()

```
try {
    MouseCursorUtils.showWaitCursor();
    ...
} finally {
    MouseCursorUtils.hideWaitCursor();
}
```

- consistency

MIDDLE

Warm-up

- first use is slow (loading classes,...)
- prepare in advance for expected user tasks
- example: context menu in the editor is computed in the background and cached when a file is opened

Lazy Initialization

- wizard panels used to be init'ed before the wizard is shown -> slow
- to show the wizard we only need the first panel
- the second panel is created when the user clicks “Next”

MIDDLE

Progressive Rendering

- show the most important information first
- the secondary information can be filled in later

What's if it's too hard?

- redesign the UI

Demo

Summary

- perceived performance != raw performance
- timely feedback
- three time constants: 0.1 sec / 1 sec / 10 sec
- measuring/profiling UI response time
- techniques used in NetBeans 3.5
- http://performance.netbeans.org

END

What Users Say

"NetBeans in the past has operated pretty slow, enough to frustrate even the biggest fan. [....] In my personal testing of 3.5, NetBeans performs as fast as any native application."

http://dmartin.org/blog/Java/?permalink=Netbeans_3_5_Beta1_is_released!.html

END

If You Only Remember One Thing...

UI applications should be optimized for human users, not for machines.

END

URLs

- NetBeans open source project
  http://www.netbeans.org/
- NetBeans architecture and APIs
  http://openide.netbeans.org/
- NetBeans performance HOWTOs, guidelines
  http://performance.netbeans.org/

END

Q&A

JavaOne℠

Sun's 2003 Worldwide Java Developer Conference℠

java.sun.com/javaone/sf