Comp Sci 108 - GUls in Java

David Winslow

Duke CS Department

February 19, 2007
Event-driven programming

Java GUI Layout

Working in Groups
Procedural programming: do things in a sequence that the programmer specifies.

**Benefits** Easy to program; easy to think about

**Drawbacks** Doesn’t allow users much freedom for complex or highly configurable tasks
Event-driven Programming

**Main idea:** respond to things as they happen rather than requiring a particular order

**Handler** a program module that responds to events

**Main loop** a central module that detects the events and notifies the appropriate handler(s)
**Swing vs. AWT**

**AWT** (the Abstract Window Toolkit)
- Original GUI API for Java
- Written by an intern
- Thin wrapper over underlying OS; uses native style

**Swing**
- Quickly replaced AWT
- Provides 'pure Java' GUI; swappable 'look n’ feel’
GUI programming lends itself to ugly code.

- `addButton(OPEN); addButton(CLOSE);`  
- `addButton(EDIT);`  
- `switch (buttonEvent.getType()){ /*...*/ }`  
- `new Window(400, 600);`
Guidelines for cleaning up your GUI code.

- Use containers (lists, etc) wherever possible.
- Extract constants early; read them in from an external file to make translating the code easy later on.
- Give a lot of thought to modularizing your GUI code (a typical window builder might look like this:)

```java
void buildWindow()
{
    makeMenu();
    makeToolBar();
    makeStatusBar();
    makeMainView();
}
```
GUI Layout

Layout is done in a tree model. It is best to allow Swing to calculate the actual pixel dimensions for you.

**Widgets** are individual GUI elements such as buttons, sliders, and checkboxes.

**Components** are elements of the GUI, including individual widgets, but also including sets of widgets like a spinbox or a tabbed pane.

**Layouts** are different sets of rules for determining the spacing of the widgets.
Working With Swing

- Keep the documentation for `javax.swing` handy while coding.
- Plenty of widgets provided in the library: buttons, checkboxes, sliders, spinboxes, dropdowns, etc.
- Use Layouts wisely! `BorderLayout`, `FlowLayout`, `GridLayout`, others.
- Use Java’s `.properties` files to set the text in your menus so that they are easy to change later.
Upcoming Projects

Coordinating with other programmers will be essential for the rest of the course.

- In the next project, two-person teams
- Larger groups after that
Coordinating a team is a matter of policy; there is no One True Way to keep a team in sync.

▶ Write high-level documentation as a group and before you start coding.

▶ Design your program such that it is divided into clearly separate subsystems with well-defined means of interaction; assign portions of the group to each subsystem.

▶ Keep your code-level documentation up-to-date and clear so that the original author is not the only one who can fix your code.
Tools to Help

General Coordination (email, wiki) Make sure that when (not if) questions come up, they can be discussed and resolved quickly.

Source Control (CVS, Subversion) Provides a central repository for your project to ease problems that arise with multiple coders modifying the same set of source files.

Documentation Generators (doxygen, javadoc) Create nice-looking, easily browsable API documentation for your project. This is not a replacement for high-level documentation, but it can make working with other peoples’ source easier. These tools can also perform checks such as finding undocumented functions.