CS1007: Object Oriented Design and Programming in Java

Lecture #16

Nov 22

Shlomo Hershkop
shlomo@cs.columbia.edu

Outline

- Java beans
- Applets

- Reading: finish chapter 7, starting 8

Announcements

- 4 more lectures after today
- 1 more homework after today (no actual code needed).
- Plan
- No class Thursday (or office hours).

Feedback

- Questions on concepts
  - Please stop me and I can give more examples
- Generic example
  - Will review some more
- Homework help
  - Will try to help later in class

Generics

- Generally when you manipulate a group of objects, in order for the compiler to be aware of the manipulations (error checking) you need to explicitly cast the general objects you are working with.

```
List myints = new LinkedList();
myints.add(new Integer(3));
Integer x = (Integer)myints.iterator().next();
```

Errors runtime

- When manipulating objects and casting, if you make a mistake on what you think is in a collection, will throw errors.

Idea

- Allow you to tell the compiler what you are thinking…

```
List<Integer> myints = new
   LinkedList<Integer>();
//adding same
Integer x = myints.iterator().next();
```

Another advantage

- Allows you to program algorithms which don't work in the dark
- Allows you to setup constraints on the objects you allow to be passed to your methods

wildcarding

- Allows you specify general types and bounded general types in your algorithms

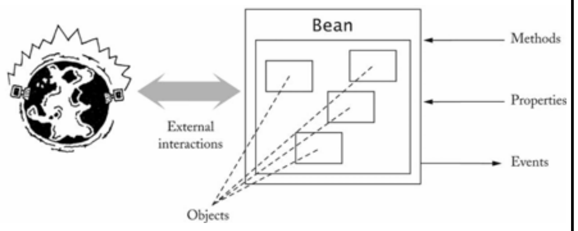
Map <String, ? extends employee>

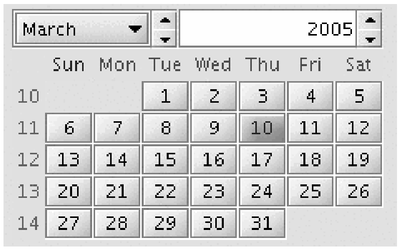
Confused?

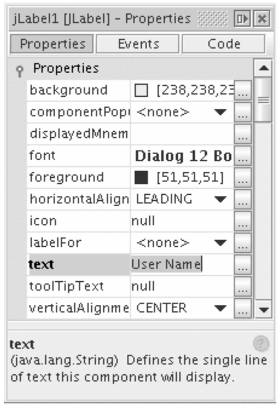
- Please review book
- Please review last lecture
- Email myself/TAs
- Can ask other students
- Use internet (not responsible for anything you happen to dig up there).

Introducing Java Beans

- Java component model
- Bean has
  - methods (just like classes)
  - properties
  - events







Properties

background	[238,238,23
componentPopı	<none>
displayedMnem	
font	Dialog 12 Bo
foreground	[51,51,51]
horizontalAlign	LEADING
icon	null
labelFor	<none>
text	User Name
toolTipText	null
verticalAlignme	CENTER

text
(java.lang.String) Defines the single line of text this component will display.

Property sheet

Façade class

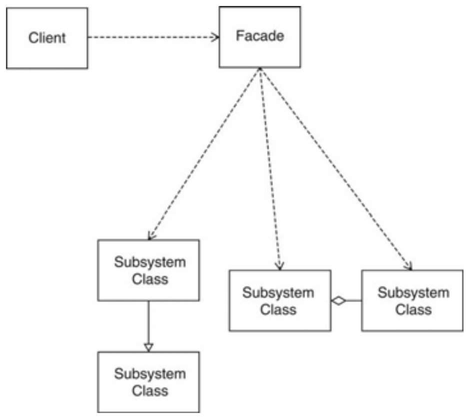
- Bean usually composed of multiple classes
- One class nominated as facade class
- Clients use only facade class methods

What kind of pattern can we extract?

- A subsystem consists of multiple classes, making it complicated for clients to use
- Implementor may want to change subsystem classes
- Want to give a coherent entry point

How JAVABEAN does it

- Define a facade class that exposes all capabilities of the subsystem as methods
- The facade methods delegate requests to the subsystem classes
- The subsystem classes do not know about the facade class



Bean Properties

- Property = value that you can get and/or set
- Most properties are get-and-set
- Can also have get-only and set-only
- Property not the same as instance field
- Setter can set fields, then call repaint
- Getter can query database

Syntax

- Not Java :-(
- C#, JavaScript, Visual Basic
- b.propertyName = value
  calls setter
- variable = b.propertyName
  calls getter

Conventions

- property = pair of methods

```
public X getPropertyName()
public void setPropertyName(X newValue)
```

- Replace propertyName with actual name

```
(e.g. getColor/setColor)
```

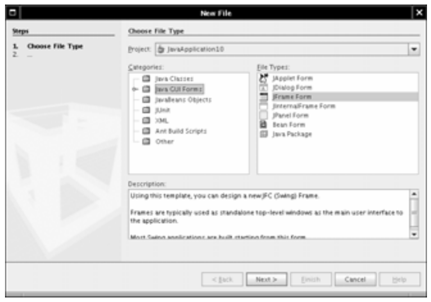
- Exception for boolean properties:

```
public boolean isPropertyName()
```

- Decapitalization hokus-pokus:

```
getColor -> color
getURL -> URL
```

Builder tool



Project: JavaApplication10

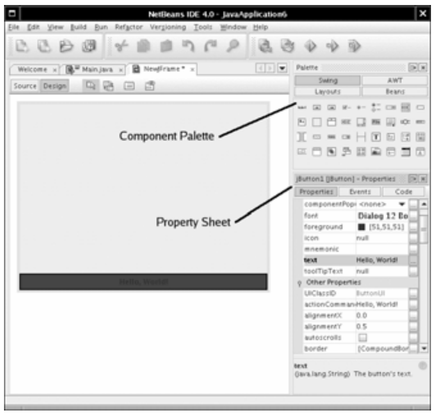
Categories: Java Classes, Java GUI Forms, JavaBeans Objects, JUnit, XML, Ant Build Scripts, Other

File Types: JApplet Form, JDialog Form, JFrame Form, JInternalFrame Form, JPanel Form, Bean Form, Java Package

Using this template, you can design a new JFC (Swing) Frame.

Frames are typically used as standalone top-level windows as the main user interface to the application.

Most Swing applications are built starting from this form.



componentPopu	<none>
font	Dialog 12 Bo
foreground	■ [51,51,51]
icon	null
mnemonic	
text	Hello, World!
toolTipText	null
UIClassID	ButtonUI
actionComman	Hello, World!
alignmentX	0.0
alignmentY	0.5
autoscrolls	☐
border	[CompoundBor

text
(java.lang.String) The button's text.

Packaging

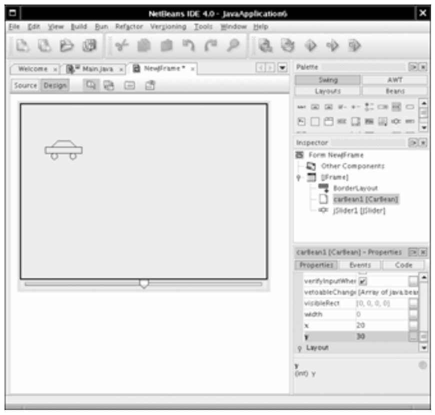
- Compile bean classes
Ch7/carbean/CarBean.java
- Create manifest file
Ch7/carbean/CarBean.mf
- Run JAR tool:
- jar cvfm CarBean.jar CarBean.mf *.class
- Import JAR file into builder environment

Composing Bean

- Make new frame
- Add car bean, slider to frame
- Edit stateChanged event of slider
- Add handler code

carBean1.setX(jSlider1.getValue());

- Compile and run
- Move slider: the car moves



verifyInputWhen	✓
vetoableChange	[Array of java.bea
visibleRect	[0, 0, 0, 0]
width	0
x	20
y	30

y
(int) y

Framework

- Set of cooperating classes
- Structures the essential mechanisms of a problem domain
- Example: Swing is a GUI framework
- Framework != design pattern
- Typical framework uses multiple design patterns

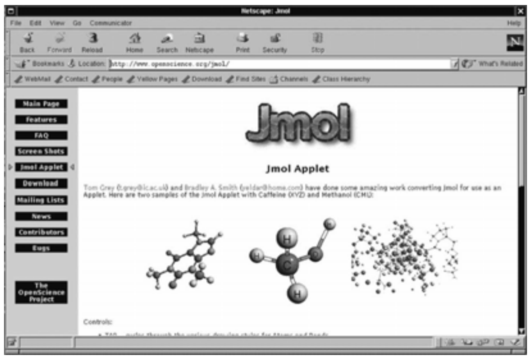
Application framework

- Implements services common to a type of applications
- Programmer forms subclasses of framework classes
- Result is an application
- Inversion of control: framework controls execution flow

Applet

- Applet: Java program that runs in a web browser
- Programmer forms subclass of Applet or JApplet
- Overwrites
  - init/destroy
  - start/stop
  - paint

Openscience.org/jmol


Netscape: Jmol
File Edit View Go Communicator Help
Back Forward Reload Home Search Netscape Print Security Stop
Bookmarks Location: http://www.openscience.org/jmol/ What's Related
WebMail Contact People Yellow Pages Download Find Sites Channels Class Hierarchy
Main Page
Features
FAQ
Screen Shots
Jmol Applet
Download
Mailing Lists
News
Contributors
Bugs
The OpenScience Project
Jmol
Controls:

Jmol Applet

Tom Grey (t.grey@ic.ac.uk) and Bradley A. Smith (yeldar@home.com) have done some amazing work converting Jmol for use as an Applet. Here are two samples of the Jmol Applet with Caffeine (XYZ) and Methanol (CML):

Applets

- Interacts with ambient browser

getParameter
showDocument

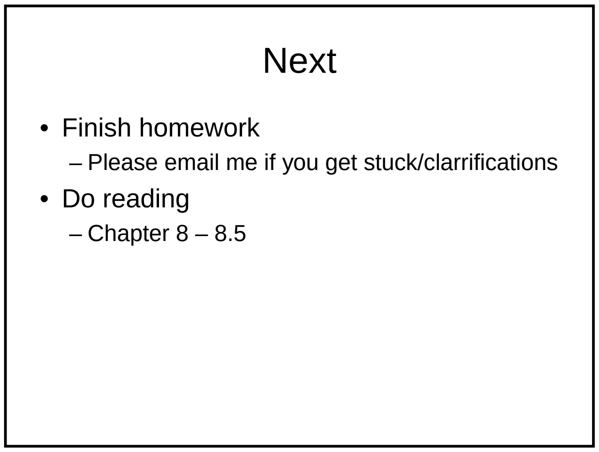
- HTML page contains applet tag and parameters

```
<applet code="BannerApplet.class"
  width="300" height="100">
   <param name="message" value="Hello,
  World!"/>
   <param name="fontname" value="Serif"/>
   <param name="fontsize" value="64"/>
   <param name="delay" value="10"/>
</applet>
```

Example

- Shows scrolling banner
- init reads parameters
- start/stop start and stop timer
- paint paints the applet surface

Ch8/applet/BannerApplet.java



Next

- Finish homework
  - Please email me if you get stuck/clarrifications
- Do reading
  - Chapter 8 – 8.5