CPSC 324
Topics in Java Programming

Lecture 14

Today …

• Go over Quiz 5
• Midterm notes
• Go over Lab 3
• JComboBox and JList
• Lab 4 & 5
• Readings:
  – Core Ch. 9 pages 396 – 399
  – Java Tutorial: JList
Quiz 5

public class Account {
    private double balance;
    public Account(double theBalance) {
        balance = theBalance;
    }
    public double getBalance() {
        return balance;
    }
}

public class Savings extends Account {
    private double monthlyFees;
    public double getBalance() {
        // ...
    }
}

1. Will these classes compile? If not, what are the error(s)?

    There is no constructor defined in Savings
    So the compiler will add a default one:
    public Savings() {
        super();
    }
    But, there isn’t a no-arg constructor in Account!

Quiz 5

    Is there another way to do this?

public class Account {
    private double balance;
    public Account(double theBalance) {
        balance = theBalance;
    }
    public double getBalance() {
        return balance;
    }
}

public class Savings extends Account {
    private double monthlyFees;
    public double getBalance() {
        // ...
    }
}

1. Fill in the two constructors. One using this(...). Default fee is $10.
Quiz 5

public class Account {
    private double balance;
    public Account(double theBalance) {
        balance = theBalance;
    }
    public double getBalance() {
        return balance;
    }
}

public class Savings extends Account {
    private double monthlyFees;
    public double getBalance() {
        // public Savings(double b) {
        super(b);
        // monthlyFee = 10.00;
        // }
        // }
        return balance;
    }
}

1. Fill in the two constructors. One using this(...). Default fee is $10.

Midterm

• There will be 10 questions
• Open book / open notes
• Closed computer
• Be sure you understand:
  – Answers to exercises
  – Answers to quizzes
  – The labs
  – Answers to questions on slides
• Be sure you have done the reading assignments
• Worth 20% of your grade
Topics we’ve covered

• Virtual Machines
  – Differences and advantages over traditional compilation

• Basic class organization
  – Methods, Fields/Variables, the main method
  – Public, private, protected

• Basic language features
  – Primitive types
  – Arrays and reference types (objects)
  – Boxing, unboxing
  – Variables (declaration and initialization)

Topics we’ve covered

• Basic Language Features (cont.)
  – Constants (final)
  – Casts
  – Control-flow (if-then-else, switch, loops)
  – Static vs. non-static methods and variables
  – Strings
  –Enumerations

• Input/Output
  – System.out.print / println
  – Scanner
Topics we’ve covered
• Exceptions
  – Passing on exceptions (throws)
  – Try catch blocks
• Garbage collection (the heap)
• Pass by value
• Inheritance
  – Extends
  – Overloading
  – Overriding

Topics we’ve covered
• Inheritance (cont.)
  – The ‘this’ and ‘super’ keywords
  – Constructors
  – The Object class (briefly)
  – Dynamic binding
  – Abstract classes and interfaces
• GUIs
  – Listener (Observer) pattern
  – Frames, panels, text fields, buttons, labels, dialogs, applets
  – Layout managers
• Inner classes
Midterm focus

Everything is fair game, but …

– Basic syntax
  • methods, loops, if-then-else, ints, strings, booleans, arrays
  • Static vs. non static methods and variables
  • Classes, main method
  • Exception handling
– References & garbage collection
– Inheritance
  • Extends, overloading, overriding
  • Dynamic binding
  • Constructor calls
  • Super and this
– Listener pattern (button events)
– Basic GUI stuff: panels, buttons, labels
– Inner classes (including local and anonymous)

Lab 3

public class Lab3 extends JApplet implements ActionListener {

private JButton setTextButton;
private JTextField textField;
private JFrame frame;

private void createGUI() {
    ...
}

public void actionPerformed(ActionEvent event) {
    ...
}

public void init() {
    ...
}
}
Lab 3

JComboBox

- Useful when there is:
  - One thing to be selected
  - Out of many possible choices
- Also known as a drop-down list
  - When it isn’t “editable”
  - setEditable(boolean)
- Useful methods:
  - JComboBox comboBox = new JComboBox();
  - Object getSelectedItem() // get currently selected item
  - addItem(Object) // add an item
  - addActionListener(...) // selection events
**JList**

- Lists are a bit more involved
  - Especially when they can be edited
  - We'll focus on non-editable lists
  - Nice for displaying a list of items that can be selected

- Lists have ListModels to manage their elements
  - ListModel is a simple interface
  - DefaultListModel is the default (simple to use since it takes care of all the methods)
  - … a lot like a regular list data structure

- Lists also have SelectModels to manage selections

---

**JList**

- Creating and initializing JList objects

```java
JList list = new JList();
list.setSelectionModel(ListSelectionModel.SINGLE_SELECTION);
list.setLayoutOrientation(JList.HORIZONTAL);  
JSObjectPane listScroller = new JSObjectPane(list);
listScroller.setPreferredSize(new Dimension(x, y));

etc.
```
JList

- Selection Models

```java
list.setSelectionModel(ListSelectionModel.SINGLE_SELECTION);
```

- SINGLE_SELECTION

- SINGLE_INTERVAL_SELECTION

- MULTIPLE_INTERVAL_SELECTION

JList

- Selection Models

```java
list.setLayoutOrientation(JList.VERTICAL);
```

- VERTICAL

- HORIZONTAL_WRAP

- VERTICAL_WRAP
JList

- Creating a model

```java
DefaultListModel model = new DefaultListModel();
model.addElement("Arlo"); // any object
model.addElement("Cosmo");
model.addElement("Elmo");
...
list.setModel(model); // set the list's model
...
model.addElement("Hugo");
model.removeAt(index); // 0 to n-1
model.insertElementAt("Milo", index);
Object obj = model.getElementAt(index);
```

JList

- JLists use a different listener interface
  
  - ListSelectionListener instead of ActionListener
    
    ```java
    list.addListSelectionListener(...);
    ```
  
  - Must implement valueChanged method
    
    ```java
    public void valueChanged(ListSelectionEvent event) {
        if(!event.getValueIsAdjusting()) {
            int index = list.getSelectedIndex();
            // do something with index (could be < 0)
            // alt: Object obj = list.getSelectedValue();
        }
    }
    ```
JComboBox and JList

- We’ll use these in the next assignment
  - GUI for our TodoPlanner
- You do not need to know these for the exam …

Lab 4 & 5

- Finish up Lab 4 (if you haven’t already)
  1. Remove the text field
  2. Add a JComboBox
  3. Use the value in the combo box to display in the dialog
     Note that combo boxes use action listeners
  4. Make your action listener an anonymous inner class!
  5. Play with editable and un-editable combo boxes (if time)
- Lab 5:
  - Same as above, but use a JList
  - Note: Don’t need the selection listener here