Swing: Menus, Window Events

15 Feb 2010
CMPT166
Dr. Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

Quiz 3: 20pts, 12min

- In Java, what does final mean when applied to: [4]
  - (a) attributes, (b) methods, (c) classes?
- Describe and contrast abstract superclasses vs. interfaces. Describe an example of each. [6]
- Interpret the following set of statements into class diagrams and into Java declarations: [10]
  - Every order has a customer, a product, and a payment.
  - There are two kinds of payment: credit card and cash.
  - All payments have an amount.
  - Every order knows how to fill itself.

Quiz 3: answers #1

- In Java, what does final mean when applied to: [4]
  - (a) attributes:
  - constant: can't change value
  - (b) methods:
  - subclasses cannot override/redefine
  - (c) classes:
  - cannot inherit / create subclass

Quiz 3: answers #2

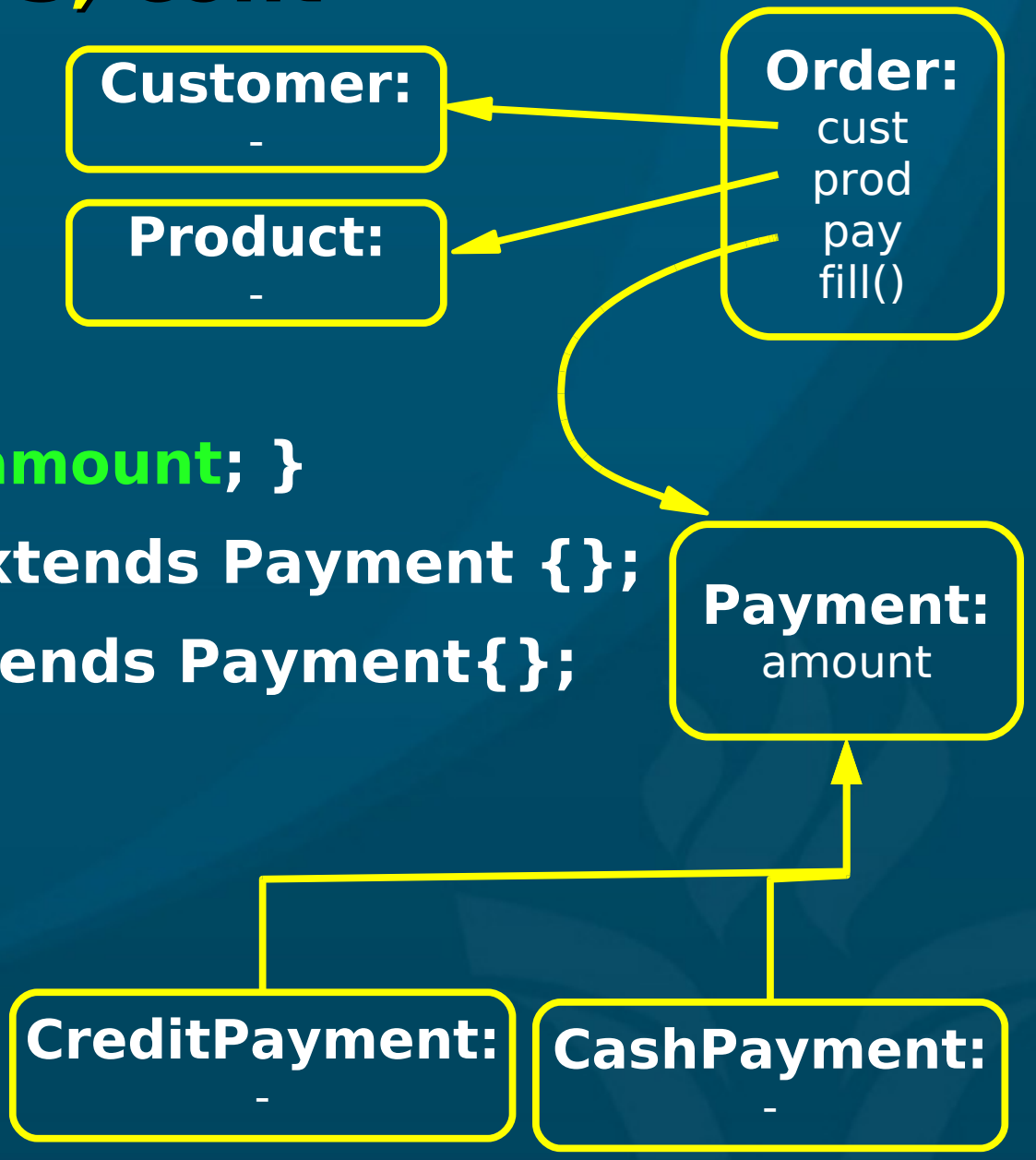
- Describe and contrast abstract superclasses vs. interfaces. Describe an example of each. [6]
  - Superclass: defines identity:
    what kind of object it is
  - e.g., Histogram extends JFrame:
    a Histogram object is a JFrame window
  - Java only allows single inheritance
  - Interface: defines capability/functionality:
    what it knows how to do
  - e.g., Histogram implements ActionListener:
    knows how to handle events

Quiz 3: answers #3

- Interpret the following set of statements into class diagrams and into Java declarations: **[10]**
  - Every order has a customer, a product, and a payment.
  - There are two kinds of payment: credit card and cash.
  - All payments have an amount.
  - Every order knows how to fill itself.

Quiz3: answers #3, cont

- class Customer {}
- class Product {}
- class Payment { float amount; }
- class CreditPayment extends Payment {};
- class CashPayment extends Payment{};
- class Order {
  - Customer cust;
  - Product prod;
  - Payment pay;
  - void fill() {}; }



Menus: JMenuBar

- A JMenuBar is a top-level container for menus:
  - **JMenuBar bar = new JMenuBar();**
- You can either add() it to the window and use the panel's regular layout manager:
  - **add(bar); // in constructor of window**
- Or use the JFrame's setJMenuBar() method to lay it out at the top of the window:
  - **setJMenuBar(bar);**
- You may have multiple menubars per window
- Each menubar may contain menus and items

Menus: JMenu and JMenuItem

- A JMenu represents one menu (e.g., “File”)
  - **JMenu fileMenu = new JMenu();**
  - **bar.add(fileMenu);**
- Contains menu items: JMenuItem
  - **JMenuItem saveItem =**
    new JMenuItem(“Save”);
  - **fileMenu.add(saveItem);**
- Attach a handler to the menu item:
  - **saveItem.addActionListener(handler);**
- JMenu is itself a subclass of JMenuItem: allows nested submenus

Window events

- We have seen: ActionEvent (button, menu)
  - also InputEvent (KeyEvent, MouseEvent)
- A WindowEvent is sent when the window interacts with the OS windowing system:
  - opening, closing, iconifying, activating
- A JFrame can register a window listener to handle these events:
    - myJFrame.setWindowListener(winevents);
- This handler must implement the WindowListener interface

Window listeners

- Implementing WindowListener means providing:
  - class WinEvents implements WindowListener {
    public void windowOpened(WindowEvent e);
  - Also windowClosing, windowClosed, windowIconified, windowDeiconified, windowActivated, windowDeactivated
- Closing: once the close button is clicked
- Closed: after the window is done
- Activated: usually when a window is clicked in
  - Only one window may be active at a time

WindowAdapter class

- Implementing the WindowListener interface means needing to implement all its methods, even if you don't need them
- WindowAdapter is an abstract superclass that implements WindowListener and provides default blank bodies for the methods
- Subclass WindowAdapter and override just the ones you need:
  - class WinEvents extends WindowAdapter {

    public void windowClosed(WindowEvent e) {