Semester Review

7 April 2011
CMPT166
Sean Ho
Trinity Western University

TRINITY WESTERN UNIVERSITY

CMPT166 on one slide

- Java:
  - Basics: types, control flow, access modifiers
    - Packages, arrays, exceptions
  - I/O: console vs. file, text vs. object, network
  - Swing and multi-threading
  - Android (concepts)
- OO Design:
  - Inheritance and polymorphism; interfaces
  - UML: CRC/class, use-case, sequence
  - Design patterns

Java basics (ch1-3, 6, 9-10)

- Java system: JDK/JRE, bytecode, VM
  - 8 primitive types, String, Math library
  - Style: naming, Javadoc comments
- if/else, switch, while, for(;;), break/continue
- Packages, public/private/protected
- Arrays: declare → allocate → create items
- Exceptions: try/throw/catch
  - Catch-or-declare rule
  - Subclassing Exception, adding auxiliary data

I/O (ch10, 19)

- Console: System.out/in/err
- File: File
- Network: Socket.getInputStream()
  - TCP vs. UDP, client-server, multithread server
- Text in: Scanner: next(), nextLine(), ...
- Text out: PrintWriter: print(), printf(), ...
- Obj in: (File/Object)InputStream, readObj
- Obj out: (File/Object)OutputStream, writeObj
  - Serializable interface, transient keyword

Swing (ch17-18)

- Event programming model:
  - main() → JFrame subclass constructor
    → create/layout widgets
  - Events → event listeners (w/anon inner classes)
- Widgets: JPanel, JButton, JLabel, JTextBox, ...
  - Action cmd: .setText, .getActionCommand()
  - Menus: MenuBar, Menu, MenuItem
- Layout managers
- Drawing: paint[Comp](), shape, colour, clip
- Thread model: event-dispatch, SwingWorker

Android

- What is it, history, component architecture
  - compare vs. iPhone OS
- Activity, Service, BroadcastReceiver, ContentProvider
- Activity lifecycle: active, paused, stop, dead
- Views (UI widgets)
- XML configuration of views
- Resources: layouts, strings, drawables
- Event listeners for buttons: OnClickListener

OO concepts (ch4-5, 7-8, 13)

- Creating classes: attributes/methods
  - private → (package) → protected → public
  - Constructors, calling self(), copy constructor
  - static: class methods/attribs; static import
- Inheritance: overriding, polymorphism
  - OO design: UML class diagram, use-case diag
    - "has a" / "is a" / "is a kind of" / "knows how to"
    - Subclass, assoc/aggr/comp, multiplicity
  - Abstract, (mthd, cls) final (attrib, mthd, cls)
  - Interfaces

UML

- Use-case diagram: requirements
  - Specifying the bounds of the system
  - Actors, use cases
  - Basic flow, alternate flows
- Component diagram: Class-Resp.-Collab.
- Sequence diagram: messages, flows
- Class diagram:
  - Inherit., assoc., aggregation, composition
  - Direction of dependency; multiplicity

Design patterns

- Creational:
  - Factory Method, Abstract Factory, Builder, Prototype, Singleton
- Structural:
  - Adapter, Bridge, Composite, Decorator, Facade, Flyweight, Proxy
- Behavioural:
  - Chain of Responsibility, Command, Interpreter, Iterator, Mediator, Memento, Observer