Modules in Java

- Classes that are related should be grouped together, may even share access to otherwise private methods/instance variables
  - Java uses package: one directory of related classes
  - private, public, protected and the default: package access
  - to compile, the CLASSPATH must be set properly
    - `setenv CLASSPATH `pwd`:`
    - `javac dirName/App.java`
    - `java dirName.App`

- Can also create anonymous (package access) classes
  - on-the-fly creation of unnamed classes
  - useful for AWT event listeners, commands, etc., see Anagram and Pixmap code examples
Interface and Classes

- **Java is single inheritance, in GUI world this can be a problem**
  - must extend Frame, Component, Composite, etc.
  - sometimes need additional functionality
- **a Java interface is similar to a C++ abstract base class**
  - classes can implement many interfaces
  - all methods in each interface MUST be implemented
  - see Comparable in Anaword, Iterator/Enumerator
- **sometimes awkward necessity of implementing all classes, even when not interested in them**
  - use an Adapter class (anonymous) that provides empty method implementations --- see WindowAdapter example
- **Anonymous classes can cause recompilation problems**
  - `rm *$*.class`
AWT concepts

- **Object < Component < Container < Window < Frame**
  - the component/container pair is an example of the Composite pattern: Container has Components, including other Containers (which are Components)
  - containers have layout managers that control how widgets are added/appear
  - Panel is the simplest, most useful container
    - defaults to Flowlayout, add widgets (other panels!)
  - Canvas is a drawable component
  - Frames is a top-level window (which is a container)
    - has a title, defaults to BorderLayout
General Gui/AWT guidelines

● Keep the GUI and Application separate, use a Controller class to mediate between widgets in the GUI
  ➤ see AnaGui and PixGui for examples
● use Command pattern (even if no explicit Command classes)
  ➤ isolate Action events on a class-per-action basis, avoid if-else chain to identify source of action
● anonymous classes, or package/private classes associated with the GUI are useful
  ➤ see PixGui and AnaGui, note that default access is package
● key member functions
  ➤ pack(), show(), validate()
● understand the repaint(), paint(), update() protocol
  ➤ call repaint() directly