Student Originated Software Object-Oriented Programming
Workshop 1, Fall, 2001-2

The objectives of this workshop are to:

1. introduce our programming support staff, Steve Dublin and Issac Overcast;
2. develop a working relationship with your programming partner;
3. learn how to use Visual Age Java in the ACC, save your programs to the program repository, and develop a method for sharing files with your partner;
4. learn how to write a simple Java program (everybody's favorite "helloWorld"), to make changes to a more interesting program "clicking ball", and to access the class library documentation;
5. get the information you need to complete the programming assignment (due Monday) – reverse engineer a "CRC" design for the clicking ball.

Isaac will explain how to access VAJ in the ACC, and how to get your own copy. Judy will explain the terms Repository, Workspace and Workbench, and give you a guided tour of the Workbench.

To do this work, you will need to use some of the methods provided by Java. To find out the appropriate interfaces to those methods, see the Java docs. www…..

On your own, you might want to read (below briefly) about the VAJ IDE, and look at the VAJ documentation on the IDE or Stanchfield & Mauny's *VisualAge for Java* (in the bookstore). See also Sun's java tutorial Object-Oriented Programming Concepts at http://java.sun.com/docs/books/tutorial/java/index.html and the API documentation for the classes below at http://java.sun.com/j2se/1.3/docs/api/index.html :

java.applet.Applet
java.awt.Graphics
java.awt.Color
java.awt.event.MouseListener
java.awt.event.MouseEvent

Things to remember:

1. This is a "closed lab" (as will be most of the Tuesday labs). That means that there is a series of activities we want you to complete here, in the lab. We may want to debrief those activities before you leave. We may ask you to present your solution(s). We may ask you to help other students (if you finish early), or have other suggestions for what to do.
2. The repository contains source code for all versions of all Java language constructs that you develop. It tracks methods, classes, interfaces, and packages, as well as their relationships to each other. Only if you do not have the source, does byte code appear in the repository.
3. The workspace is a user's "current slice" of the repository. It is non-visual.
4. The Workbench provides a visual representation of the workspace.
5. Several programmers can work from the same repository (simultaneously), though we probably won't do this for a while.
6. In VAJ, the method (not the file) is the unit of compilation and version control. (Full source editing is available in VAJ 3.5 – but please try method-level interaction first.) Methods are incrementally compiled (into bytecode) every time you save a change to any piece of your VAJ code.

Activities:

1. With your partner, get on to VAJ, and decide in whose account will reside the repository for your work. Set the appropriate permissions so that each person will be able to access the joint repository.

2. **HelloWorld** Application. The purpose of this exercise is to familiarize you with the VAJ Integrated Development Environment (IDE).
   - Use the Create Class SmartGuide (icon "C"). On the first page of the dialog, specify the project, package and classname (respectively): My First Project, my.first.pkg, and HelloWorld. Leave the superclass as java.lang.Object, and do NOT check **Browse the class when finished** or **Compose the class visually**. On the second page, select the public modifier, and tell VAJ to create stubs for **methods that must be implemented**, constructors from superclass, and **main().** Click **finish**.
   - Expand the tree on the left side of the Workbench window to see the main() method inside the HelloWorld class. Select the main() method and edit it to add the following code:

     System.out.println("Hello, World!");

     (The above Java statement is a message to the println method (of class **System** and its data member **out**), with a string as a parameter.) You might also want to document your code.
   - Right click on the pop-up menu in the source pane, and click **Save**. This will save your method (and check syntax of your method). When all is well, a runner icon appears next to the HelloWorld Class.
   - Run your HelloWorld application by highlighting HelloWorld (class) or main (method) in HelloWorld.
   - Export your HelloWorld class to a file folder, and look at what files are created, and what is in those. This file folder has what you need to take this Java program with you. Be sure you understand what you have done before going to the next step.

3. **ClickMe** Applet. Import the **ClickMe** and **Spot** Classes from the SOS share. ClickMe is an applet that creates a window and makes a colored spot appear when and where you click the mouse. Try it! Then read the code and make sure you understand (in general) what the code does and the interaction between the components. You might try drawing a sequence diagram.

   Note differences between this applet and the HelloWorld application.

   - Make the following changes to the **ClickMe** Applet. To do this, you will have to go on a treasure hunt for the methods you will need.
     - Change the color of the ball to green, then to purple.
     - Change the shape of the ball (haha) to a square, then to a rectangle.
     - Instead of having a geometric shape appear when you click, have ClickMe materialize your names, in purple or some distinctively personal color.

4. Work on your assignment for Monday – write up the CRC cards for ClickMe.

5. If you still have time, you can:
   - Start working on BallWorld (Chapter 6, Budd).
   - Explore the VAJ **Scrapbook** window. Here you can organize, develop and test ideas for your Java programs and experiment with code fragments without specifying a containing class. It will be a lot easier to figure out how this works with your partner than alone.