Event Handling

If possible, try at least one exercise with separate classes as event handlers, at least one where the main Applet is the event handler, and at least one where you use an inner class for the event handler. You might want to try three (or four, if you try anonymous inner classes also) of problem 1, instead of moving directly on to problems 2 and later. My solution set has four alternative solutions to problem 1 (separate classes, Applet implementing interface, named inner classes, anonymous inner classes).

1. Make an applet whose background color changes from red to blue whenever the user presses the mouse. You can use setBackground(Color.RED) and setBackground(Color.BLUE) to change the background color, and if you want to toggle back and forth, you can use getBackground() to find the current background color. Remember to use mousePressed, not mouseClicked (mouseClicked fires only when the press and release are in the same place, so if you jiggle the mouse too much when clicking, mouseClicked does not fire).

You will almost definitely find this easier if you start by copying my circle-drawing example and editing it. The best approach is either the interface version or the inner-class version, but you can try multiple different approaches. My solution set has four solutions to this problem: separate classes, interfaces, named inner classes, and anonymous inner classes.

Also, since the size of the applet doesn't matter too much for the first two examples, you might want to try the Eclipse shortcut of skipping the HTML file and just right-clicking inside the applet and doing "Run As ... Java Applet".

Finally, note that Eclipse can greatly simplify writing the methods needed for an interface. If your code says "public class MyApplet extends Applet implements MouseListener", you can right-click in the code, go to the "Source" submenu, and choose "Override/Implement Methods...". The methods from the interface will be selected automatically and inserted when you press OK. If you use an inner class instead, you can use the same trick, but just interactively choose the methods of interest for Eclipse to insert.

2. Make an applet that prints "a key was hit" whenever the user presses a key on the keyboard. Put the printout in the Java console.

3. Make an applet that is red when the mouse is on the left side of the applet, and blue when the mouse is on the right side.

4. Make an applet that draws a picture of Bill Gates. Switch the picture to Larry Ellison whenever the user clicks the mouse. (Note: call the applet's repaint() method to tell it to reinvoke paint).