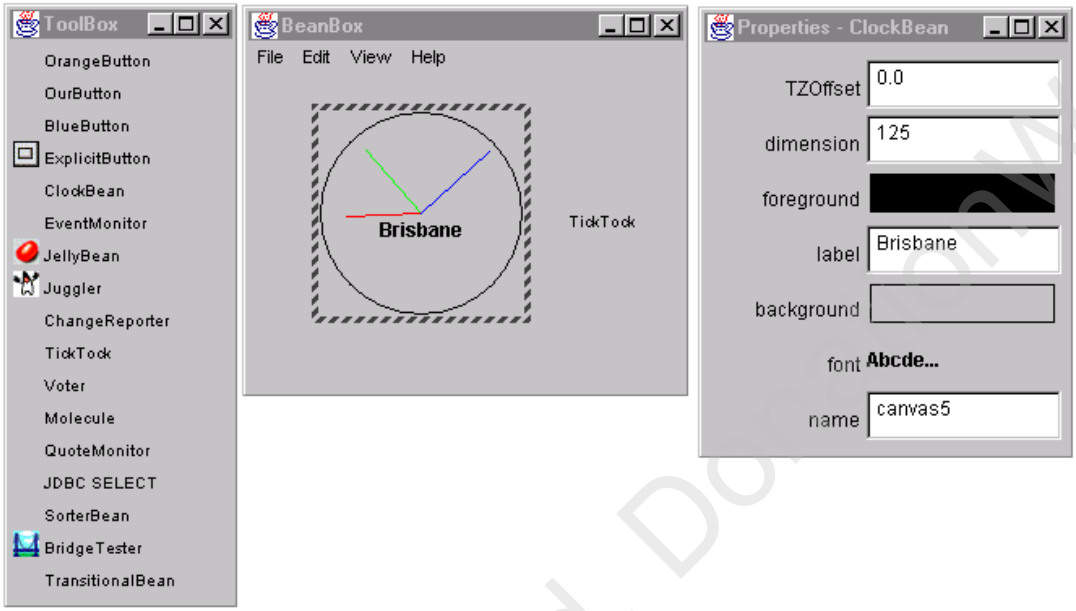
Exercise: ClockBean

In this exercise, you will build a ClockBean and associate it with the standard BDK TickTock bean. You should also gain experience in using the java.util.Calendar class.

Note: you will need the JavaSoft Bean Development Kit installed on your computer to do this exercise.

1. The Exercise

The following picture shows an example of a ClockBean within the BDK's beanbox tool:



As you can see, the ClockBean has a number of properties, some of which (foreground, background, font and name) are standard for all components, while others (TZOffset, dimension and label) are particular to the ClockBean itself.

2. The ClockBean Class

Your bean should "flesh out" the following class fragment:

```
import java.awt.*;
import java.awt.event.*;
import java.util.*;
import java.beans.*;

public class ClockBean extends Canvas implements PropertyChangeListener
  {
  private static final int DEFAULTSIZE = 125,
                           MINSIZE = DEFAULTSIZE;
  private static final double two60 = 60 * 60,
                              twoPI = 2 * Math.PI;

  // TODO: need to add local state for the label, dimension and TZOffset
  //       properties (but consider the properties available via the
  //       superclasses before adding anything, esp. for the dimension
  //       property…)

  public ClockBean ()
    {
    this ("", DEFAULTSIZE, 0.0);
    }
  public ClockBean (String label, int size, double tzOffset) { … }

  public synchronized void setLabel (String label) { … }

  public synchronized String getLabel () { … }

  // throws an IllegalArgumentException if size < MINSIZE
  public synchronized void setDimension (int size)
```

```
    throws IllegalArgumentException { … }

  public synchronized int getDimension () { … }

  public synchronized void setTZOffset (double tzOffset) { … }

  public synchronized double getTZOffset () { … }

  public synchronized Dimension getMinimumSize () { … }

  public synchronized Dimension getPreferredSize () { … }

  public synchronized void propertyChange (PropertyChangeEvent pce) { … }

  private synchronized int seconds () { … }

  public synchronized void paint (Graphics g)
    {
    int size = getSize ().width,
        mid = size / 2,
        seclength = mid - 5,
        minLength = mid - 10,
        hourLength = mid - 15,
        seconds = seconds ();
    double hourAngle = twoPI * (seconds - 3 * two60) / (12 * two60),
           minuteAngle = twoPI * (seconds - 15 * 60) / two60,
           secondAngle = twoPI * (seconds - 15) / 60;

    g.drawOval (0, 0, size - 1, size - 1);

    Color savedColor = g.getColor ();
    g.setColor (Color.red);
    g.drawLine (mid, mid, mid + (int) (hourLength * Math.cos (hourAngle)),
                mid + (int) (hourLength * Math.sin (hourAngle)));

    g.setColor (Color.green);
    g.drawLine (mid, mid, mid + (int) (minLength * Math.cos (minuteAngle)),
                mid + (int) (minLength * Math.sin (minuteAngle)));

    g.setColor (Color.blue);
    g.drawLine (mid, mid, mid + (int) (seclength * Math.cos (secondAngle)),
                mid + (int) (seclength * Math.sin (secondAngle)));
    g.setColor (savedColor);

    FontMetrics fm = getFontMetrics (getFont ());
    int xPos = mid - (fm.stringWidth (label) / 2);
    g.drawString (label, xPos, mid + fm.getHeight () + fm.getLeading ());
    }
}
```

3. Compiling And Installing The Bean In The Beanbox

To compile the bean you have created:

C:> javac ClockBean.java

A bean is packaged within a “Java Archive” file, along with a manifest file that is examined by a Bean’s container. You will need to:

- create a partial manifest file
- create an archive to contain the Bean and the manifest
- move the archive to a location where the beanbox can find it

The next lines show how this process is done *(note: you will need to replace the* C:\BDK *portion of the command line with the location of the BDK on your system)*:

C:> copy CON: Manifest.stub
Name: ClockBean.class
Java-Bean: True
^Z *(type the ‘control-Z’ character, followed by the ‘return’ key)*

C:> jar cfm ClockBean.jar Manifest.stub ClockBean.class
C:> copy ClockBean.jar *C:\BDK*\jars\ClockBean.jar

4. Starting The Beanbox; Assembling And Running The Beans

a) To run the beanbox:

C:> cd *C:\BDK*\beanbox
C:> run

You will now see the beanbox application start up. You should see the name ClockBean contained within the list of beans in the left-hand window.

b) Click on the word 'ClockBean' and then place a new ClockBean by clicking into the main window.

c) Click on the word 'TickTock' and then place a new TickTock bean into the main window. In the right hand 'properties' window, change the value of the displayed 'interval' property from 5 to 1.

d) With the TickTock bean selected, select the Edit/Events/propertyChange/propertyChange menu entry.

e) Point at the ClockBean (you will see a red line following your pointer around) and click; a window will pop up containing a list of available methods in the ClockBean class. Select the propertyChange method from this list. The beanbox will pop up a notice saying that it is generating and compiling an adaptor class. Once this is complete, you will have a working clock that updates itself when it is sent a propertyChange event by the TickTock bean (approx. every one second).

f) Select the View/Disable Design Mode menu and then the View/Hide Invisible Beans menu to see how your ClockBean will appear to a user of your bean.

Congratulations! You have made and used a Java Bean.