

Tutorial

Creating your first BlueDecode GUI Project

The BlueDecode GUI framework allows the users to create a dynamic and rich user interface with custom graphics. The frame work has the following components:

- GUI Window (a window which holds the interface)
- Rendering Panel (the render engine used to draw the widgets on screen)
- TaskManager (a abstract class which holds all the scripting and execution)

Prerequisites

You will need to download the both the framework jar file itself and the sample project files and follow through with the tutorial.

The user will have to install and set up the classpath for the project. To do this, the user will need to point the class path to the frameworks jar file.

eg.

Classpath = <drive>:<directory>\BlueDecode.jar

1. Task Management

For the first project, we will start by creating the task manager for our window. The task manager allows users to customize their program and run time sequences without needing to toy around with too many window options. It achieves this by separating the logic programming from the window drawing part.

The Task Manager does the following things:

- 1) Execute scripted actions when widgets are clicked on by the user
- 2) Do an infinite while loop method.

In this tutorial we will create a testTask class which extends the abstract object TaskManager. The testTask class will implement the doRun and doWidgetAction abstract methods of the TaskManager.

BlueDecode – Sample Project Tutorial

Example Code:

```
import java.util.Enumeration;
import BlueDecode.*;

public class testTask extends TaskManager {

    public void doRun() {
        if(_clock.tick()){
            setClock(0.01);
            for(int i=0;i<_screens.size();i++){
                Screen temps = (Screen) _screens.get(i);
                Enumeration n = temps.getObjects();
                while(n.hasMoreElements()){
                    Widget tempW = (Widget) n.nextElement();
                    if(tempW.isRotatable()){
                        double nn = tempW.getAngle()
                            +tempW.getRotate();
                        tempW.setAngle(nn);
                    }
                }
            }
        }
    }

    public void doWidgetAction(wAction a) {
        if(a.getAction().equalsIgnoreCase("change")){
            changeScreen(a.getTartget());
        }
    }
}
```

On the above example code, during the while loop, the task manager does a run time loop which cycles through all of the screens and all the widgets in its screen to change the rotation of objects which are rotation enabled.

We've also implemented the doWidgetAction method, which takes in a widget's action and depending on the command perform the built in changeScreen method which changes the screen being displayed.

2. GUI XML

The next step is to create the GUI XML file used to put in the widget components into the screens. Please follow the code provided in the next page for reference.

The beauty of the GUI XML system is that it allows developers to hard code the functionality of the program through scripts into the widget, and the logical execution into the task manager. This allows developers to develop the GUI interface and functionality in parallel without needing to recompile and hard code the interface design every time.

The GUI hierarchy is as follows:

- ScreenList(holds all the screen objects)
- Screen (holds widgets)
- Widgets(individual widgets and other components)

Screen

A basic screen object which is used to hold and organize widget objects

```
<screen>
```

```
<screenName> The screen ID
```

```
<visible>    Determines if this screen is visible or not.
```

```
</screen>
```

Note: only the **last** screen with the a visibility set to true is visible at run time.

3. Widget Object

A basic widget object which is used as an interactive visual component

<widget>

<id>	Id of the widget
<X>	X location
<Y>	Y location
<bX>	X boundary length
<bY>	Y boundary length
<angle>	initial angle of the widget
<cmd>	command used
<frame>	initial image used (can use a number of frame objects for animations)

note: the Java run time about to take in JPEG, GIF, and BMP images by default. To get transparency to work for images, the image must be saved as a GIF with a transparent background.

<hint>	mouse over hint to be used
<rotatable>	rotation enabled is set to true if this object can be rotated, false otherwise
<rotation>	degrees being rotated by
<target>	target ID (used when scripting screen changes, and widget interaction)

</widget>

BlueDecode – Sample Project Tutorial

Example:

```
<?xml version="1.0"?><!DOCTYPE Screen[<ELEMENT screenName (#PCDATA)>]>
<ScreenList>
  <Screen>
    <screenName>titlescreen</screenName>
    <visible>true</visible>
    <widget>
      <id>title</id>
      <X>30</X>
      <Y>30</Y>
      <bX>400</bX>
      <bY>200</bY>
      <angle>0.0</angle>
      <cmd>change</cmd>
      <frame>demo_title.jpg</frame>
      <hint>Demo Program Title</hint>
      <rotatable>true</rotatable>
      <rotation>1</rotation>
      <target>scene1</target></widget></Screen>
  <Screen>
    <screenName>scene1</screenName>
    <visible>false</visible>
    <widget>
      <id>light</id>
      <X>-100</X>
      <Y>-100</Y>
      <bX>700</bX>
      <bY>700</bY>
      <rotatable>true</rotatable>
      <rotation>1</rotation>
      <frame>light.jpg</frame></widget>
    <textBox>
      <id>text</id>
      <X>10</X>
      <Y>370</Y>
      <bX>350</bX>
      <bY>90</bY>
      <text>In the beginning there was light.....</text>
      <scrolling>false</scrolling>
      <bgcolor>black</bgcolor>
      <fontcolor>white</fontcolor></textBox>
    </Screen>
  </ScreenList>
```

4. Utilizing the GUI

Then finally, to utilize the XML, we create a constructor and main function entry for the program. To do this need to do the following:

1. instantiate the taskManager object
2. instantiate the GUIwindow object
3. pass both the path to the XML file and the taskManager object to the GUIwindow object

Example code

```
import java.util.Vector;
import BlueDecode.*;

public class GUITest {

    public static void main (String argv[]) throws Exception{
        testTask n = new testTask();
        n.setStatusRun(true);
        n.setClock(0.01);
        GUIwindow cn = new GUIwindow("Test GUI", "", 800,
500, "drive:\\projectfolder\\test.xml", n, 3);
    }
}
```

Compile, and you are set ! Now if user doesn't like what they have create for the graphics, or dislike the layout of the interface, they can always go back to the XML file change the layout and sizes, and even edit the graphics without needing to recompile hard coded interface components again.