

Tutorial: Time Of Day

Part 2

GUI Design in NetBeans

November 18, 2010

Author		Kees Hemerik
Goals		Getting acquainted with NetBeans GUI Builder Illustrate the separation between GUI and computation proper
Version	Date	Comments
0.3	2010 11 18	Various small corrections in variable names; URLs updated.
0.1	2009 11 05	First draft, partly based on http://www.netbeans.org/kb/docs/java/quickstart-gui.html

1 Introduction

In Part 1 [1] of this tutorial we have developed a simple class `TimeOfDay` following the Design by Contract method. In this part we extend the project with the development of a simple graphical user interface, using NetBeans's GUI Builder. The end product will be a window displaying a clock time in hours, minutes and seconds, together with some controls to set the clock to a particular time or to reset the clock. The window will look as follows:



2 Opening or Creating a Project

If you have completed Part 1 of the tutorial, open the project in which the class `TimeOfDay` resides and continue with section 3.

Otherwise, create a new application project with name `prTimeOfDay` as follows:

1. Choose File → New Project. Alternately, you can click the New Project icon in the IDE toolbar.
2. In the Categories pane, select the Java node and in the Projects pane, choose Java Application. Click Next.
3. Enter `prTimeOfDay` in the Project Name field and specify the project location.
4. Leave the Use Dedicated Folder for Storing Libraries checkbox unselected.
5. Ensure that the Set as Main Project checkbox is selected and clear the Create Main Class field.
6. Click Finish.

The IDE creates the `prTimeOfDay` folder on your system in the designated location. This folder contains all of the project's associated files, including its Ant script, folders for storing sources and tests, and a folder for project-specific metadata. To view the project structure, use the IDE's Files window.

3 Creating a GUI container

To proceed with building our interface, we need to create a Java container within which we will place the other required GUI components. In this step we'll create a container using the `JFrame` component and place the container in a new package.

To create a `JFrame` container:

1. In the Projects window, right-click the `prTimeOfDay` node and choose New → `JFrame Form`.
2. Enter `TimeOfDayUI` as the Class Name.
3. Click Finish.

The IDE creates the `TimeOfDayUI` form and the `TimeOfDayUI` class within the `prTimeOfDay` application and opens the `TimeOfDayUI` form in the GUI Builder.

4 Adding Visual Components to the Form

In the GUI Builder on the top right is the Palette with the available controls, grouped in several panes. Open the pane with Swing controls.

4.1 Adding a `JLabel`

1. In the Palette window, select the Label component from the Swing controls category.
2. Move the cursor over the form and click to position the selected label on the form. The label component is added to the form and a corresponding node representing the component is added to the Inspector window (left below).
3. In the Inspector window, right-click the node corresponding to the newly added label. In the popup menu, select Change Variable Name ... to open the Rename dialog. Change the name to `labelTimeDisplay` and click OK.

Before going further, we need to edit the properties of the JLabel we just added. Though you can edit component display text at any point, the easiest way is to do this as you add them. In general, to edit the properties of a component, select the component by clicking on it. Its properties will be listed in the Properties window in the lower right part of the GUI Builder as a list of (name,value) pairs. A property can be edited by clicking on the icon to the right of the value, which opens a custom editor for the property value.

To edit the properties of the JLabel component we just added, click the component and edit its properties as follows:

text The text property determines the text of the label.

1. Select the **text** property;
2. Click its icon to open its editor;
3. Change the text to `00:00:00` ;
4. Click OK to close the editor.

font The font property determines the font in which the text of the JLabel component is displayed.

1. Select the **font** property in the Properties window;
2. Click its icon to open the font editor;
3. Change the font size to 48;
4. Click OK to close the editor.

The size of the JLabel automatically adjusts to the changes of the properties. Drag the JLabel to the desired position (see the picture in the introduction).

4.2 Adding JSpinner Controls

In order to change the hours, minutes, and seconds of our clock we use three JSpinner controls. A spinner control lets the user select a value from a range of values. It consists of an edit field and two small buttons, labeled with an up arrow and a down arrow. The values range of the spinner control is determined by its **model** property.

To add a spinner control to the form, take the following steps:

1. Select a Spinner component from the Palette window and place it on the form;
2. In the Inspector window, rename its variable to `spinnerHours`;
3. In the Properties window, select the **model** property and open its editor;
4. Select the option Spinner Model Editor from the combobox;
5. For the Model Type, select Number;
6. For the Number Type, select Integer;
7. Set the Initial Value to 0;
8. Check Minimum, and set the value to 0;
9. Check Maximum, and set the value to 23;
10. Set the Step Size to 1;
11. Click OK to close the editor;

In a similar way, place a second spinner control, this time with name `spinnerMinutes`, minimum 0 and maximum 59. Place a third spinner control, with name `spinnerSeconds`, minimum 0 and maximum 59.

Drag the spinner controls to their desired positions (see the picture in the introduction).

4.3 Adding JButton Controls

In order to add a Set button to the form, take the following steps:

1. Select a Button component from the Palette window and place it on the form;
2. Rename its variable to `buttonSet`
3. Change the text property to Set

Place a second button, this time with variable name `buttonReset` and text Reset.

Run the application. It will show a window somewhat similar to the one shown in the introduction. It is just an empty facade with nothing behind it, however. We can change the spinner values and click the buttons, but this has no effect on the time displayed. In the following sections we will remedy this by adding some code.

5 Adding Non-visual Components to the Code

In order to maintain the time, we need the following components:

- A `TimeOfDay` object for recording the time data, as developed in part 1 [?] of this tutorial.
- A `Timer` object for generating clock ticks [2].

These components can not be added by the GUI Builder. Therefore we have to switch to the usual Source view and edit it to add code.

1. Change from the Design view to the Source view of the file `prTimeOfDayUI.java`.
2. Towards the top of the file, below the line **package** `my.timeofday`, add the following line:

```
import javax.swing.Timer;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;
```

3. Towards the end of the file, below the grey part with variable declarations maintained by the GUIBuilder, add the following instance variable declarations:

```
private TimeOfDay timeOfDay;
private Timer clockTimer;
```

4. In the constructor `TimeOfDayUI()`, following the method call to `initComponents()`, add the following piece of code:

```
timeOfDay = new TimeOfDay();
clockTimer = new Timer(1000, new ActionListener(){
    public void actionPerformed(ActionEvent e){
        timeOfDay.tick();
        labelTimeDisplay.setText(timeOfDay.toString());
    }
});
clockTimer.start();
```

Run the application. It will display the time, which is updated every second.

6 Adding Event Handling

1. Change from the Source view to the Design view.
2. Double-click the Reset button (or alternatively, right-click and in the pop menu select Events → Action → actionPerformed(..)). This will take you back to the Source view, where the GUIBuilder has created an empty event handler, i.e. a method declaration of the following form:

```
private void buttonResetActionPerformed(java.awt.event.ActionEvent evt) {  
    // TODO add your handling code here:  
}
```

3. Add the following handling code:

```
timeOfDay.reset();  
labelTimeDisplay.setText(timeOfDay.toString());
```

Perform the same steps for the Set button. This time, add the following handling code:

```
timeOfDay.setHMS(  
    (Integer)spinnerHours.getValue(),  
    (Integer)spinnerMinutes.getValue(),  
    (Integer)spinnerSeconds.getValue()  
);  
labelTimeDisplay.setText(timeOfDay.toString());
```

Run the final application. It will display the time, which changes every second. By means of the other controls the time can be set or reset.

7 Concluding Remarks

In this tutorial we have given an impression of how to develop a simple graphical interface with NetBeans's GUI Builder. We have also shown how the interface (class `TimeOfDayUI`) can be separated from the computation proper (class `TimeOfDay`) and we have shown some basic event handling. It should be noted that we have hardly scratched the surface of GUI design and of event handling, however. For some other introductory tutorials on GUI design with NetBeans, see [3, 4]. For a more comprehensive learning on this subject see [5]. For lessons on event handling see [6].

References

- [1] Tutorial: Time OfDay, Part 1, Design by Contract in Java.
- [2] *How to Use Swing Timers*, The Java Tutorials → Creating a GUI with JFC/Swing → Using Other Swing Features,
<http://download.oracle.com/javase/tutorial/uiswing/misc/timer.html>
- [3] *Introduction to GUI Building*, NetBeans IDE 6.5 Tutorial,
<http://www.netbeans.org/kb/docs/java/gui-functionality.html>
- [4] *Designing a Swing GUI in NetBeans IDE*, Tutorial,
<http://www.netbeans.org/kb/docs/java/quickstart-gui.html>
- [5] *Java and javaFX GUI Application Learning Trail*,
<http://www.netbeans.org/kb/trails/matisse.html>
- [6] *Lesson: Writing Event Listeners*, The Java Tutorials →
<http://download.oracle.com/javase/tutorial/uiswing/events/index.html>