

## **48024 Object Oriented Design**

### **Self-Study Module: GUI Building in Java**

This module covers elementary graphical user interface design. It comprises a laboratory exercise exploring the basics of GUI programming in Java.

## 48024 Object Oriented Design

### Self-Study Module: GUI Building in Java

#### Exercise 1 - Graphical User Interface

Objectives:

- To gain skill in the use of Java GUI classes.
- To understand the mechanism of event-driven applications in Java.

Recommended reading:

- Wu, Chapter 13: GUI Objects and Event-Driven Programming.
- Schildt, Chapters 22 and 26.

Online resources:

- SampleFrame class (Downloadable from UTSONline OOD Course Documents/ Self-Study Modules).

#### Task

A GUI component is a visual object with which the user interacts via the mouse or the keyboard. An event-driven application can respond to or handle events the GUI elements generate from an action of the user over them.

Java offers two main collections or packages of GUI classes that you can use to create GUIs as simple or as complicated as you wish. These two packages are: the older original one called AWT (Abstract Window Toolkit) and an improved one called Swing. You can write the solution to this exercise is AWT or Swing since both packages contain the standard classes that you will need but we strongly recommend that you use Swing since that is what is commonly used nowadays for both commercial and non-commercial applications.

In this exercise, you will create the main window (a frame) of a GUI Java class similar to that shown in Figure 1. It will contain the following GUI components and responses to user actions:

- A menu bar with the following items:
  - File → Open - When selected, a dialog box displays the message “Open menu item was selected”.
  - Edit → Copy - When selected, a dialog box displays the message “Copy menu item was selected”.
  - Quit → Quit. When selected, the window is closed and the program terminates.
- A button of label “Next” - Every time the user clicks over the button, a label with the text “Next button was clicked” is displayed.
- A button of label “Back” - Every time user clicks over the button, a label with the text “Back button was clicked” is displayed.
- Every time the user double clicks on any part of the frame, the background colour of the frame changes to a different colour. If the user double clicks again, the colour of the background will change again to its original colour. The user can do this as many times as he/she wishes.

**Self-Study Module: GUI Building in Java**

First, compile and test the SampleFrame class. Call the main method by right clicking on the class in the BlueJ environment. Study what the class does and its code; use it as a starting point to build the solution to this exercise.

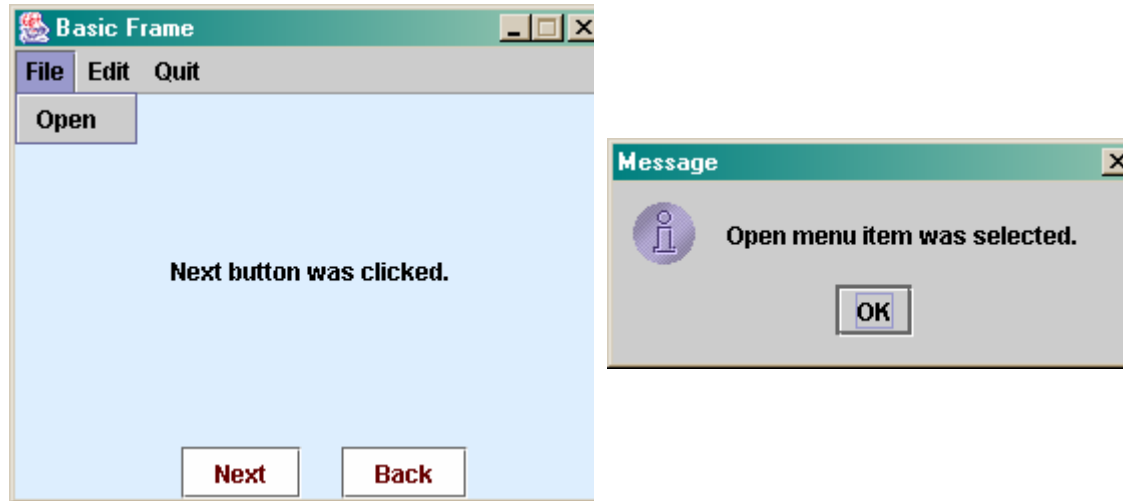


Figure 1. Look of sample solution

In this exercise you will practice four of the tasks usually involved in any GUI development:

1. Definition of windows and containers objects (the use of layouts is also included here).
2. Definition (creation and specification of characteristics such as size, colour, etc) of components (visual or not) and their placement in a container (whether a panel or a frame), i.e. adding the component to the container.
3. Handling and responding to events of windows and components.
4. Creation and display of dialog boxes.

**Hints and development suggestions**

There are quite a few visual components you have to create and which events you might need to handle in order to achieve the desired behaviour based on the user interaction with each component. It is recommended that you have a complete reference of the Java classes you'd like to use so you know what the attributes and public methods (including constructors) you can use when you need to create objects of a class and change their characteristics.

One option is to access the Sun Java API official web page <http://java.sun.com/j2se/1.4.2/docs/api/index.html>. Here you can look for any class in any Java package and get its specification details.

*A quick task*

Go to the given website and look for the class `JButton` in the `java.swing` package and answer the following questions:

- What do I need to do to create a new `JButton`?
- How can I change the colour of the background of the button?
- Which Java interface do I have to implement to handle the event created when the `JButton` is clicked?

Now, look at the code of the `SampleFrame` class and answer the following questions:

- Where is the size of the main window set?
- What objects are created in the class?
- Which interfaces are implemented and why?
- What happens when the button is clicked? Where does the program control go? Use the BlueJ debugger to find the answer by placing a break point where you suspect the program goes when you click the button.

Understanding the code of the `SampleFrame` and answering these questions should give enough knowledge to start working on the solution to this laboratory.

Some components in a GUI class might not create events result of user interaction so they do not need to have a listener associated with them. In these cases they might maintain the same state/appearance through out the life of the program (e.g. a label title of the panel).

Take into account that important information about an event is stored in the input parameter of the methods of the interface handling the events. Looking at this could give you the answers to questions such as how do you know that there was a single or a double click of the mouse?

When writing event-driven application take into account:

- the best components and layouts that will guarantee good GUI design principles (usability, consistency, clarity, etc).
- how those components will interact with the user and if there are events that you need to handle and which the program responses are.
- in the specific case of Java, you should identify the interfaces, classes and methods that will participate in the handling of event.

**48024 Object Oriented Design**  
**Self-Study Module: GUI Building in Java**

**Review Questions**

1. What modification would have to be done if instead of using Button objects we use two Checkbox objects?
2. The following is a screen print of the main window of an application called Special Effects. Can you list all the Java GUI classes that might have been used to develop this user interface?

