

NGÔN NGỮ LẬP TRÌNH JAVA

Nội dung

Cơ bản về ngôn ngữ lập trình Java

Lập trình hướng
đối tượng

Biến, từ khoá,
kiểu dữ liệu

Biểu thức, các
cấu trúc điều
khiển

Dữ liệu kiểu
mảng

Các khía cạnh nâng cao của lập trình hướng đối tượng

Thiết kế lớp

Thiết kế lớp
nâng cao

Xử lý ngoại lệ

Generics

Java Collection
Framework

Xây dựng ứng dụng đồ họa Java

LT đồ họa với
AWT

LT đồ họa với
SWING

GUI PROGRAMMING WITH SWING

Nội dung

1. Giới thiệu về SWING
2. Window Builder

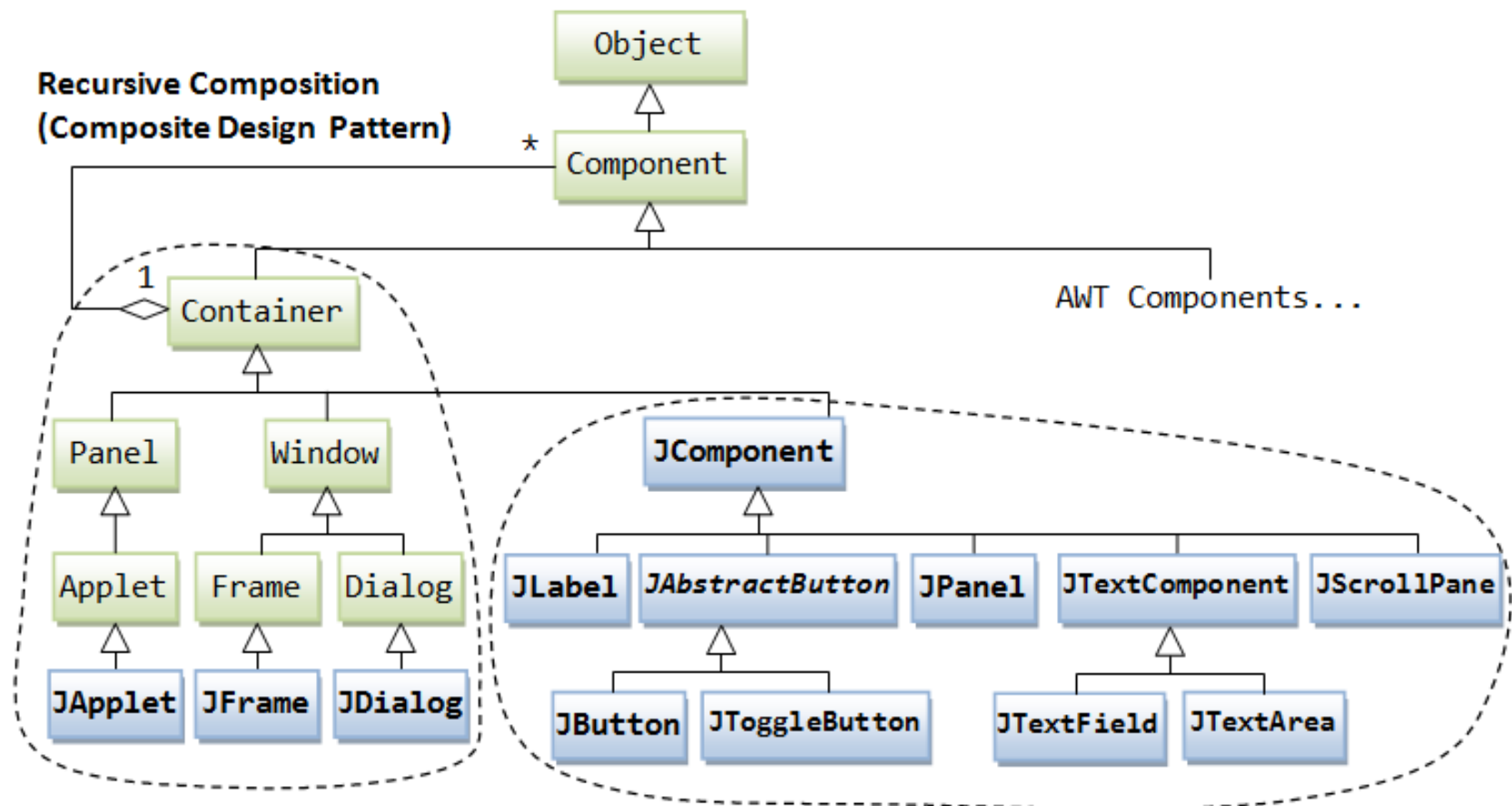
3. Lập trình GUI với SWING

3.1 Giới thiệu về SWING

❑ SWING:

- ❑ là một phần của Java Foundation Classes (JFC)
- ❑ là một tập các JavaBeans components có thể sử dụng dễ dàng bằng cách kéo và thả vào các giao diện.
- ❑ phức tạp, lớn và chứa 18 packages
- ❑ Sử dụng các lớp xử lý sự kiện của AWT (gói `java.awt.event`)
- ❑ Sử dụng `LayoutManager` của AWT
 - ❑ Bổ sung: `GridBagLayout`, `FlowLayout`, `BoxLayout` (gói `javax.swing`)
- ❑ Bổ sung: `JLayeredPane` và `JInternalFrame` để tạo ra ứng dụng đa tài liệu (Multiple Document Interface)
 - ❑ A MDI is a graphical user interface in which multiple windows reside under a single parent window

SWING's components



Swing components bắt đầu với tiền tố J, ví dụ: JLabel, Jbutton, ...

Container

❑ Top-level container

- ❑ JFrame
- ❑ JDialog
- ❑ JApplet

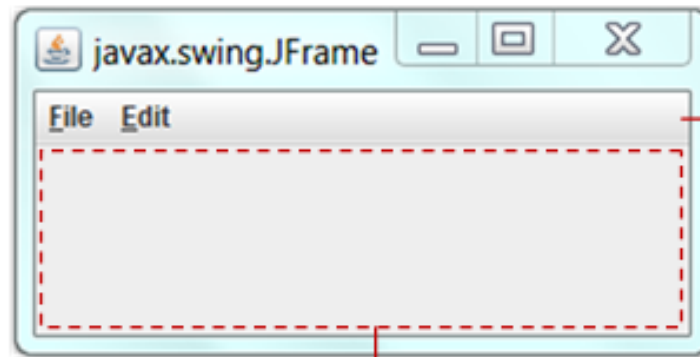
❑ Secondary container

- ❑ JPanel

❑ Content-Pane

- ❑ JComponents ko thể được thêm trực tiếp vào toplevel container
- ❑ Jcomponents được thêm vào content-pane của toplevel container
- ❑ Content-Pane là một JPanel (secondary container)
 - ❑ Để nhóm các components lại với nhau
 - ❑ Và thiết lập layout

`javax.swing.JFrame`



Menu Bar
(Optional)

Content Pane

```
Container cp = aJFrame.getContentPane();
aJFrame.setContentPane(aPanel);
```

Get & set content-pane

```
public class TestGetContentPane extends JFrame {
    // Constructor
    public TestGetContentPane() {
        // Get the content-pane of this JFrame, which is the
        // All operations, such as setLayout() and
        Container cp = this.getContentPane();
        cp.setLayout(new FlowLayout());
        cp.add(new JLabel("Hello, world!"));
        cp.add(new JButton("Button"));
        .....
    }
    .....
}
```

```
public class TestSetContentPane extends JFrame {
    // Constructor
    public TestSetContentPane() {
        // The "main" JPanel holds all the GUI component
        JPanel mainPanel = new JPanel(new FlowLayout());
        mainPanel.add(new JLabel("Hello, world!"));
        mainPanel.add(new JButton("Button"));

        // Set the content-pane of this JFrame to the mainPanel
        this.setContentPane(mainPanel);
        .....
    }
    .....
}
```

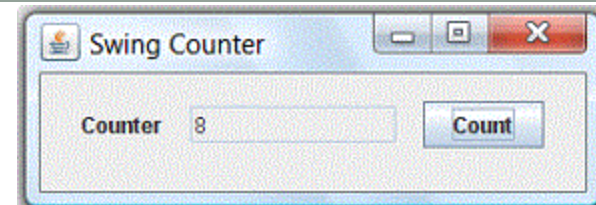

Swing program template 1

```
1  import java.awt.*;
2  import java.awt.event.*;
3  import javax.swing.*;
4
5  // A Swing GUI application inherits from top-level container javax.swing.JFrame
6  public class ..... extends JFrame {
7
8      // private variables
9      // .....
10
11     /** Constructor to setup the GUI components */
12     public .....() {
13         Container cp = this.getContentPane();
14
15         // Content-pane sets layout
16         // cp.setLayout(new ....Layout());
17
18         // Allocate the GUI components
19         // .....
20
21         // Content-pane adds components
22         // cp.add(....);
```

Swing program template 2

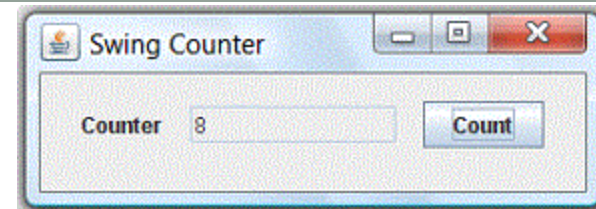
```
24         // Source object adds listener
25         // .....
26
27         setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
28         // Exit the program when the close-window button clicked
29         setTitle("....."); // "this" JFrame sets title
30         setSize(300, 150);   // "this" JFrame sets initial size (or pack())
31         setVisible(true);    // show it
32     }
33
34     /** The entry main() method */
35     public static void main(String[] args) {
36         // Run GUI codes in Event-Dispatching thread for thread-safety
37         SwingUtilities.invokeLater(new Runnable() {
38             @Override
39             public void run() {
40                 new .....(); // Let the constructor do the job
41             }
42         });
43     }
44 }
```

Exam 1: SWINGCounter 1



```
1  import java.awt.*;           // Using AWT containers and components
2  import java.awt.event.*;     // Using AWT events and listener interfaces
3  import javax.swing.*;        // Using Swing components and containers
4
5  // A Swing GUI application inherits from top-level container javax.swing.JFrame
6  public class SwingCounter extends JFrame {
7      private JTextField tfCount; // Use Swing's JTextField instead of AWT's TextField
8      private int count = 0;
9
10     /** Constructor to setup the GUI */
11     public SwingCounter () {
12         // Retrieve the content-pane of the top-level container JFrame
13         // All operations done on the content-pane
14         Container cp = getContentPane();
15         cp.setLayout(new FlowLayout());
16
17         cp.add(new JLabel("Counter"));
18         tfCount = new JTextField("0", 10);
19         tfCount.setEditable(false);
20         cp.add(tfCount);
21
22         JButton btnCount = new JButton("Count");
23         cp.add(btnCount);
```

Exam 1: SWINGCounter 2



```

25 // Allocate an anonymous instance of an anonymous inner class that
26 // implements ActionListener as(ActionEvent) listener
27 btnCount.addActionListener(new ActionListener() {
28     @Override
29     public void actionPerformed(ActionEvent e) {
30         ++count;
31         tfCount.setText(count + "");
32     }
33 });
34
35 setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE); // Exit program if close-window button clicked
36 setTitle("Swing Counter"); // "this" JFrame sets title
37 setSize(300, 100); // "this" JFrame sets initial size
38 setVisible(true); // "this" JFrame shows
39 }
40
41 /** The entry main() method */
42 public static void main(String[] args) {
43     // Run the GUI construction in the Event-Dispatching thread for thread-safety
44     SwingUtilities.invokeLater(new Runnable() {
45         @Override
46         public void run() {
47             new SwingCounter(); // Let the constructor do the job
48         }
49     });
50 }
51 }

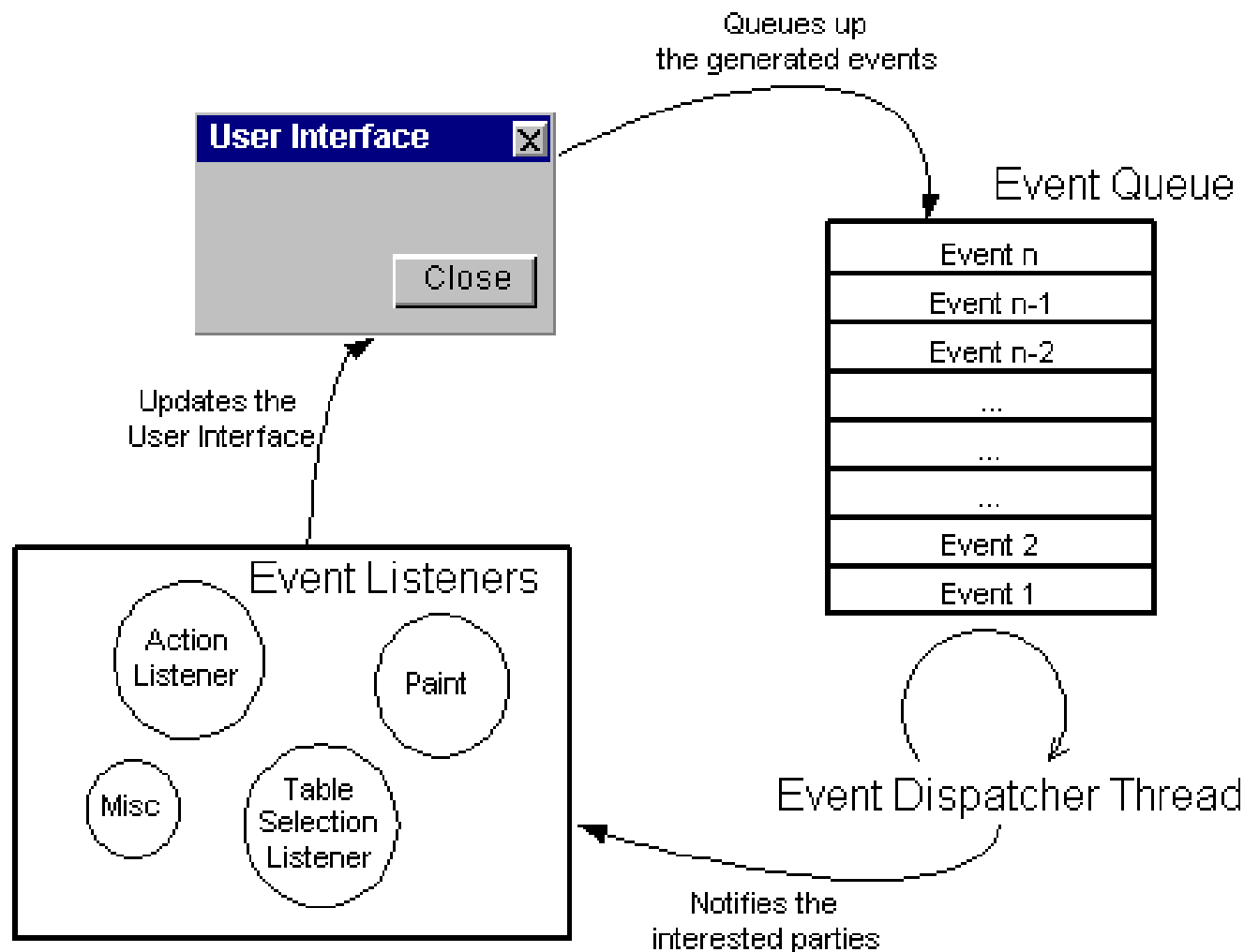
```

EventQueue.invokeLater()

Để an toàn, khuyến nghị sử dụng thread "Event-Dispatching" thay vì "Main-Program"

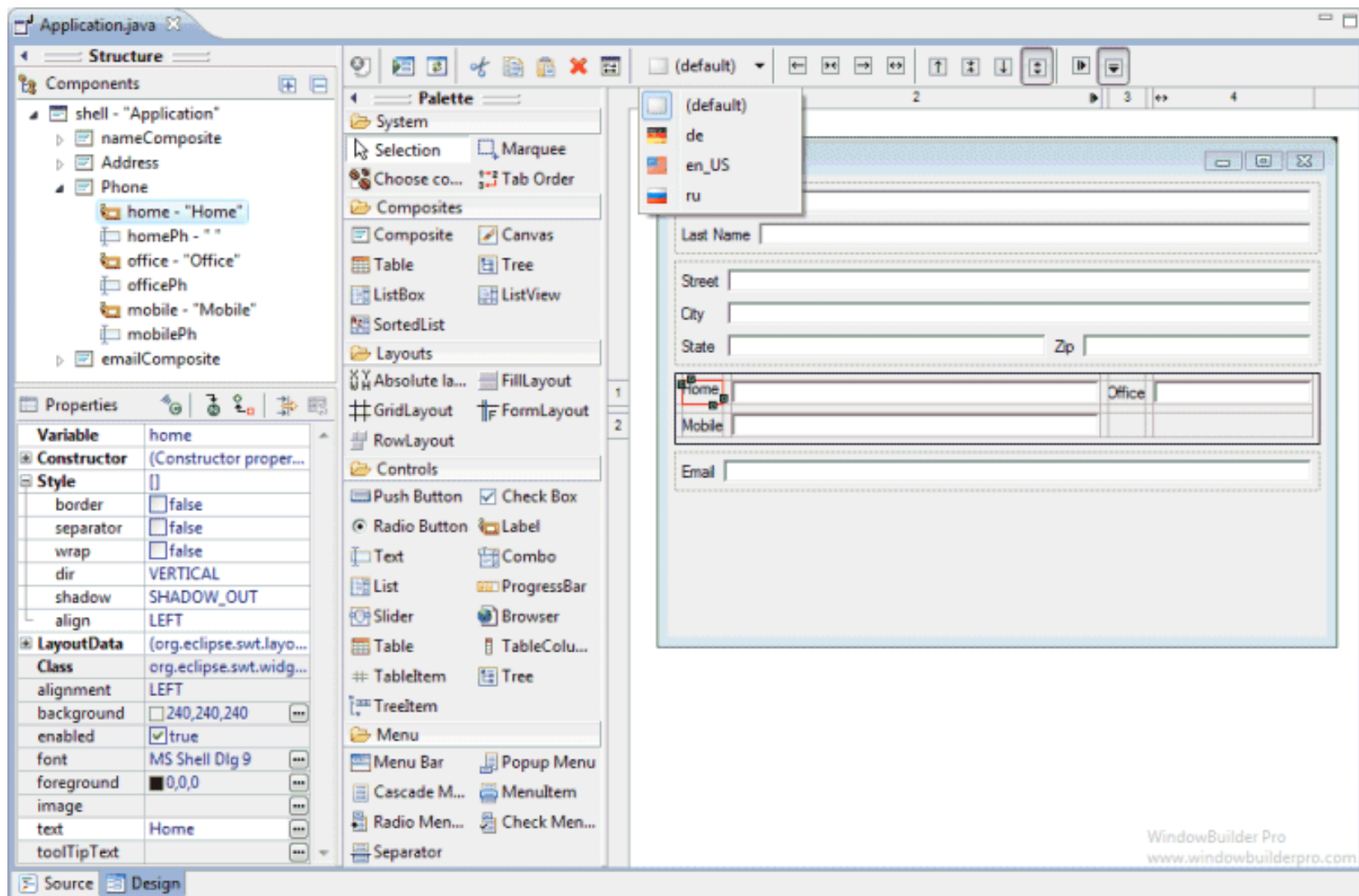
Swing event-dispatch model

(<http://www.javaworld.com/article/2073477/swing-gui-programming/customize-swingworker-to-improve-swing-guis.html>)



3. Lập trình GUI với SWING

3.2. WindowBuilder



3. Lập trình GUI với SWING

3.2 Window Builder

❑ Cài đặt :

- ❑ kiểm tra version Eclipse
- ❑ Help/InstallNewSoftware
- ❑ Thêm update site:

<http://www.eclipse.org/windowbuilder/download.php>

```

.eclipseproduct Comodo_JFrame.java
1 name=Eclipse Platform
2 id=org.eclipse.platform
3 version=4.3.2

```




Update Sites

Eclipse Version	Release Version		Integration Version	
	Update Site	Zipped Update Site	Update Site	Zipped Update Site
4.4 (Luna)			link	link (MD5 Hash)
4.3 (Kepler)	link	link (MD5 Hash)	link	link (MD5 Hash)
4.2 (Juno)	link	link (MD5 Hash)	link	link (MD5 Hash)
3.8 (Juno)	link	link (MD5 Hash)	link	link (MD5 Hash)
3.7 (Indigo)	link	link (MD5 Hash)	link	link (MD5 Hash)
3.6 (Helios)	link	link (MD5 Hash)		
3.5 (Galileo)	link	link (MD5 Hash)		
3.4 (Ganymede)	link	link (MD5 Hash)		

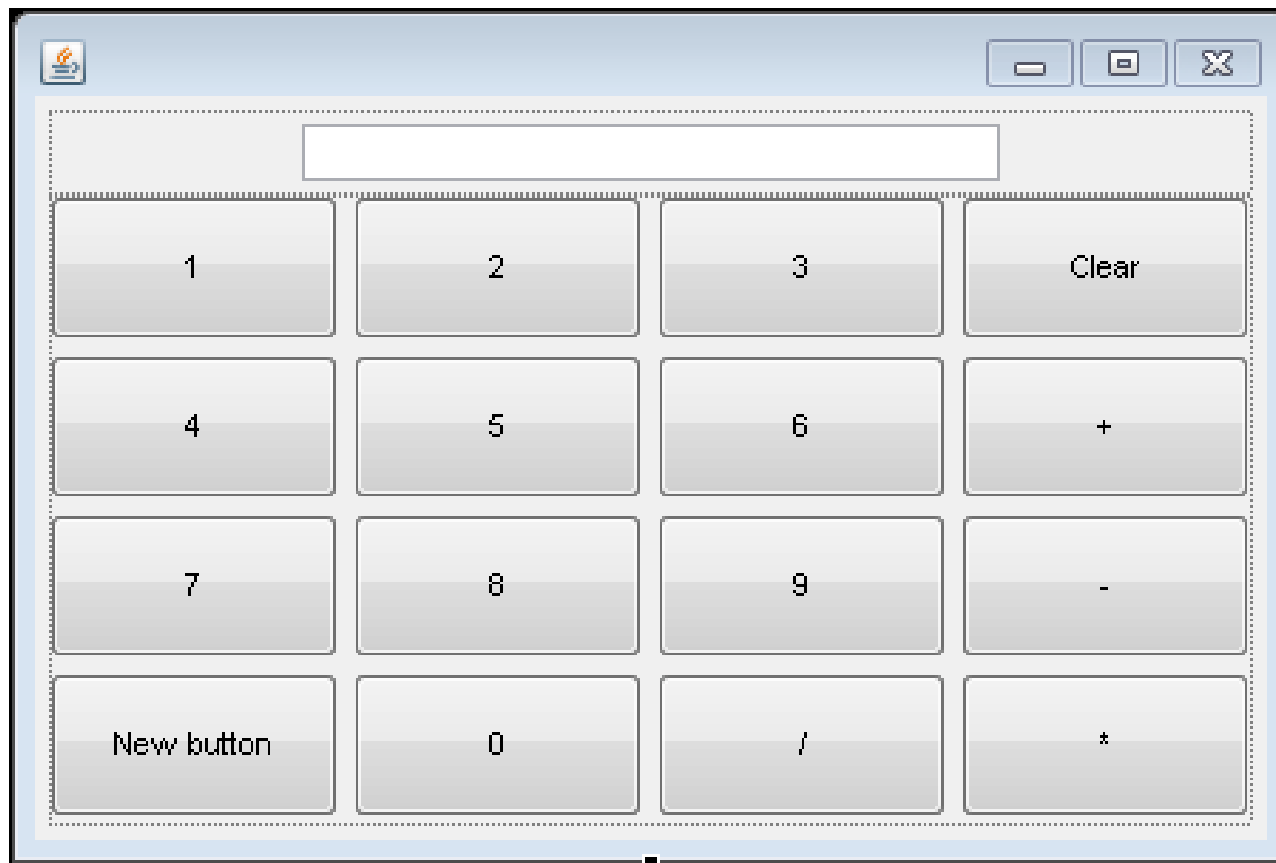
Work with:

Find more software by working with the ["Available Software Sites"](#) preferences.

type filter text

Name
▶  Swing Designer
▶  SWT Designer
▶  WindowBuilder Engine (Required)

Bài tập 1: Xây dựng Calculator với WindowBuilder



Bài tập 2 : MenuDemo

