UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

NGÔN NGỮ LẬP TRÌNH JAVA

BUỔI 12
LẬP TRÌNH GIAO DIỆN VỚI AWT, SWING (TT)

GVGD: ThS. Lê Thanh Trọng

NỘI DUNG

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

1. **Một số component thông dụng**
   - JTextField
   - JTextArea
   - JComboBox
   - Jlist
   - JTabbedPane
   - JSplitPane
   - JToolBar, Icon, ImageIcon
2. **Look And Feel**
3. **Đồ họa trong Java**

NỘI DUNG

1. **Một số component thông dụng**
2. Look And Feel
3. Đồ họa trong Java

JTextField

- Là ô nhập dữ liệu dạng văn bản trên 1 dòng
- Thuộc tính
  - text
  - horizontalAlignment
  - editable
  - Columns
- Các constructor của JTextField:

  JTextField()

  JTextField(int columns): Tạo một text field trống có số cột xác định

  JTextField(String text): Tạo một text field với văn bản có sẵn

  JTextField(String text, int columns): Tạo một text field với văn bản có sẵn và số cột xác định

JTextField

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Một số phương thức khác
  - *public void setText(String text)*
  - *public String getText()*
  - *public void setEditable(boolean b)*
  - *void setFont(Font f)*
  - *void setForeground(Color c)*
  - *void setHorizontalAlignment(int alignment)*
  - *void addActionListener(ActionListener l)*

JTextArea

- Khung cho phép người sử dụng nhập vào nhiều dòng văn bản
- Các constructor của JTextArea:

  JTextArea()

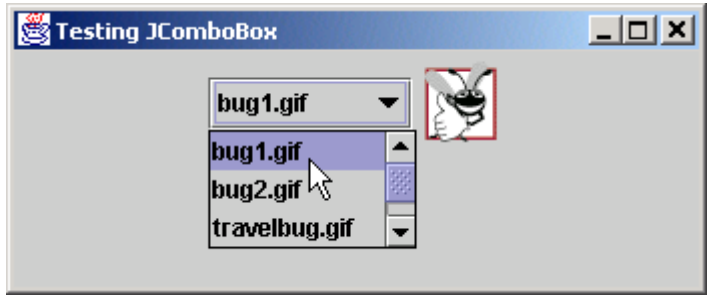
  JTextArea(String s)

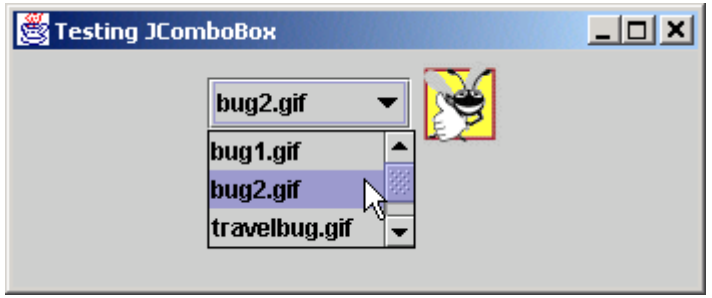
  JTextArea(int rows, int columns)

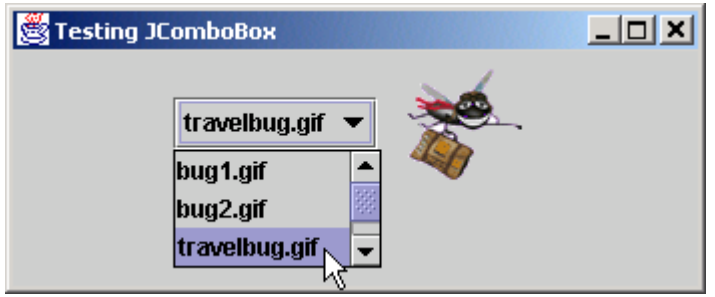
  JTextArea(String s, int rows, int columns)
- Các phương thức khác:
  - *public void append(String text)*
  - *public int getRows()*
  - *public int getColumns()*

JComboBox

- Dùng để liệt kê danh sách các mục mà người dùng có thể chọn
- Còn được gọi là drop-down list
- Phát sinh sự kiện ItemEvent khi người sử dụng chọn 1 mục trong danh sách
- *JComboBox*(Object[] items)





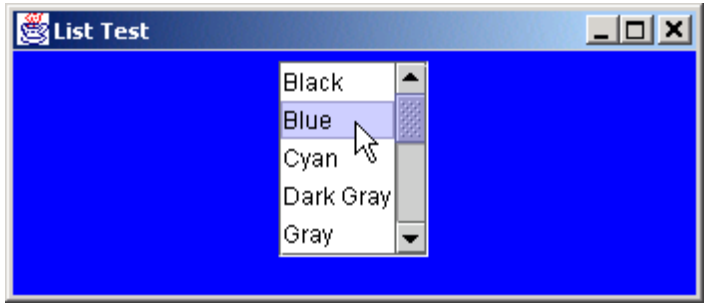


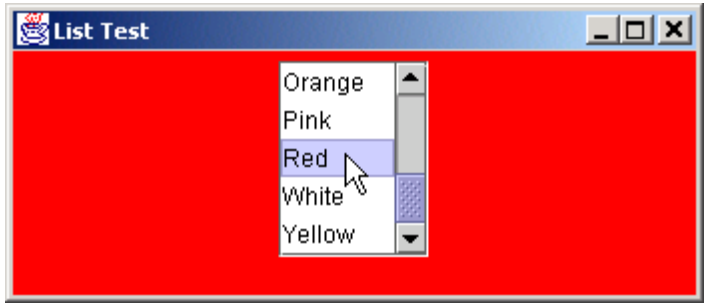
Ví dụ

```
private JComboBox imagesJComboBox; // combobox to hold
names of icons
   private JLabel label; // label to display selected icon
   private static final String[] names =
     { "bug1.gif", "bug2.gif",  "travelbug.gif", "buganim.gif" };
   private Icon[] icons = {
     new ImageIcon( getClass().getResource( names[ 0 ] ) ),
     new ImageIcon( getClass().getResource( names[ 1 ] ) ),
     new ImageIcon( getClass().getResource( names[ 2 ] ) ),
     new ImageIcon( getClass().getResource( names[ 3 ] ) )
};
     public ComboBoxFrame()
   {
     super( "Testing JComboBox" );
     setLayout( new FlowLayout() ); // set frame layout
     imagesJComboBox = new JComboBox( names ); // set up
JComboBox
      imagesJComboBox.setMaximumRowCount( 3 ); //
 display three rows
      imagesJComboBox.addItemListener(
        new ItemListener() // anonymous inner class
        {
          // handle JComboBox event
          public void itemStateChanged( ItemEvent event )
          {
            // determine whether item selected
            if ( event.getStateChange() ==
ItemEvent.SELECTED )
              label.setIcon( icons[
                imagesJComboBox.getSelectedIndex() ] );
          } // end method itemStateChanged
        } // end anonymous inner class
      ); // end call to addItemListener

      add( imagesJComboBox ); // add combobox to JFrame
      label = new JLabel( icons[ 0 ] ); // display first icon
      add( label ); // add label to JFrame
    } // end ComboBoxFrame constructor
}
```

JList

- Jlist
  - Danh sách các mục chọn
  - Có thể chọn 1 hoặc nhiều mục
  - Phát sinh ListSelectionEvent khi người dùng chọn
- Các phương thức
  - *JList(Object[] listData)*
  - *int getSelectedIndex()*
  - *Object[] getSelectedValues()*
  - *void setListData(Object[] listData)*
  - *void setSelectedIndex(int idx)*
- ListSelectionListener
  - *void valueChanged(ListSelectionEvent e)*

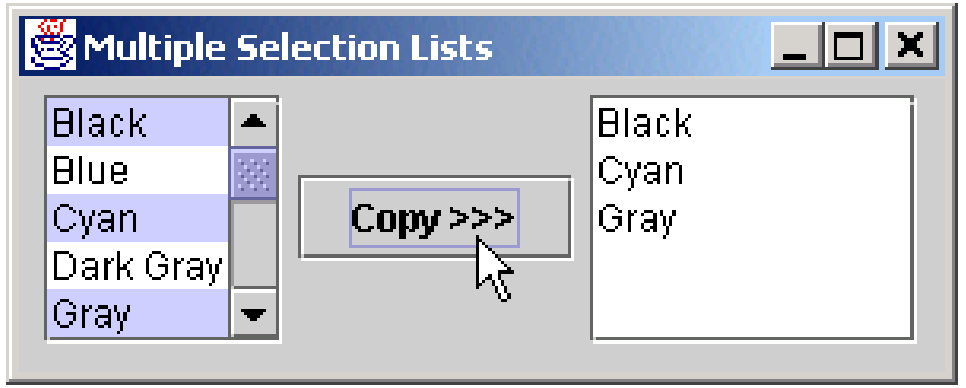




Multiple-Selection Lists

- Chọn nhiều mục trên Jlist

```
private JList colorList
private String colorNames[] = { "Black", "Blue", "Cyan", "Dark Gray", "Gray"};
Container container = getContentPane();
container.setLayout( new FlowLayout() );
colorList = new JList(colorNames );
colorList.setVisibleRowCount( 5 );
colorList.setFixedCellHeight( 15 );
colorList.setSelectionMode(
ListSelectionModel.MULTIPLE_INTERVAL_SELECTION );
container.add(new JScrollPane( colorList ) );
```



JTabbedPane

- Được sử dụng để chuyển đổi giữa một nhóm các thành phần bằng cách nhấp vào tab có tiêu đề hoặc biểu tượng nhất định

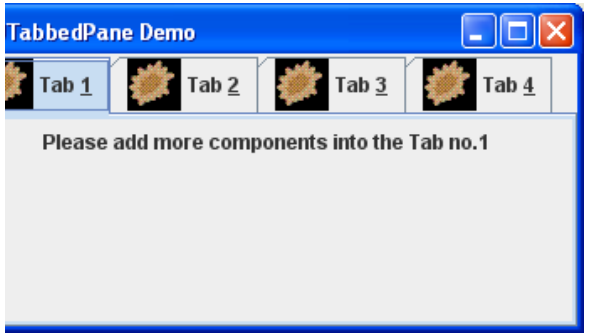
javax.swing

Class JTabbedPane

java.lang.Object
  └ java.awt.Component
    └ java.awt.Container
      └ javax.swing.JComponent
        └ **javax.swing.JTabbedPane**

All Implemented Interfaces:

ImageObserver, MenuContainer, Serializable, Accessible, SwingConstants



JTabbedPane

- Tạo mới đối tượng JTabbedPane

  JTabbedPane tabbedPane = new JTabbedPane();

- Gắn thêm 1 Tab mới vào đối tượng JTabbedPane

  tabbedPane.addTab("Tab name", icon, component, "Tooltip");

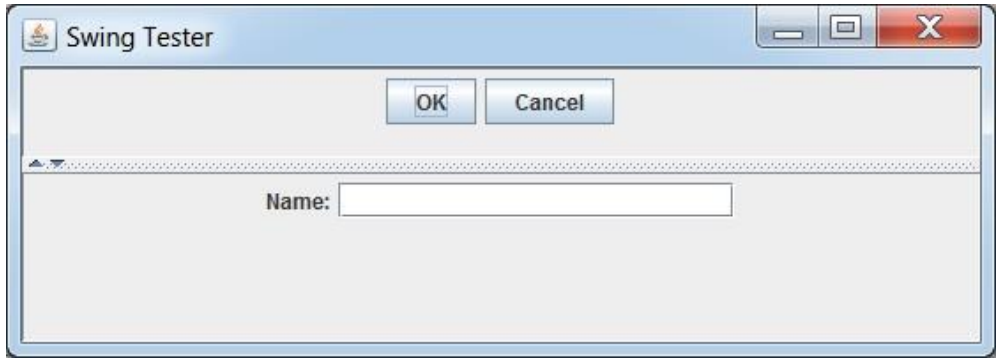
```
JTabbedPane tabbedPane = new JTabbedPane();
ImageIcon icon = new ImageIcon("middle.gif");
JPanel panel1 = new JPanel(new FlowLayout());
panel1.add(new JLabel("Please add more components into the Tab no.1"));
tabbedPane.addTab("Tab 1", icon, panel1, "This is the tab no.1");
tabbedPane.setMnemonicAt(0, KeyEvent.VK_1);

JPanel panel2 = new JPanel(new FlowLayout());
panel2.add(new JLabel("Please add more components into the Tab no.2"));
tabbedPane.addTab("Tab 2", icon, panel2, "This is the tab no.2");
tabbedPane.setMnemonicAt(1, KeyEvent.VK_2);
```

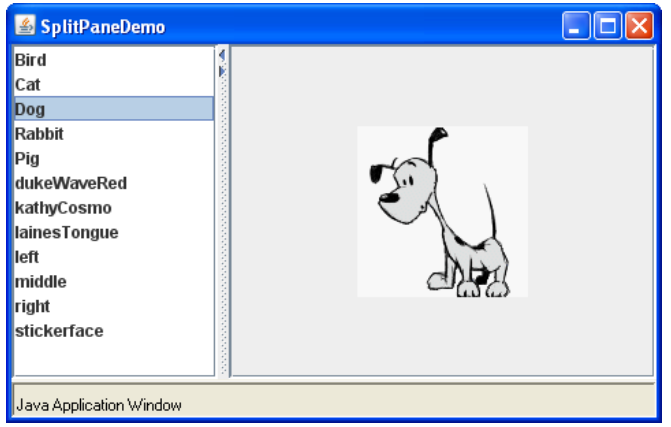
JTabbedPane


```
JPanel panel3 = new JPanel(new FlowLayout());
panel3.add(new JLabel("Please add more components into the Tab no.3"));
tabbedPane.addTab("Tab 3", icon, panel3, "This is the tab no.3");
tabbedPane.setMnemonicAt(2, KeyEvent.VK_3);

JPanel panel4 = new JPanel(new FlowLayout());
panel4.add(new JLabel("Please add more components into the Tab no.4"));
panel4.setPreferredSize(new Dimension(410, 50));
tabbedPane.addTab("Tab 4", icon, panel4, "This is the tab no.4");
tabbedPane.setMnemonicAt(3, KeyEvent.VK_4);

JFrame fr = new JFrame("JTabbedPane Demo");
fr.setBounds(50, 50, 400, 300);
fr.add(tabbedPane, BorderLayout.CENTER);

fr.setVisible(true);
```

JSplitPane

- JSplitPane được sử dụng để phân chia hai thành phần và có thể thay đổi kích thước các thành phần
- Bằng cách sử dụng JSplitPane, người dùng có thể thay đổi kích thước thành phần theo cách thủ công cho đến kích thước tối thiểu của nó
- JSplitPane có thể có hai loại, một là splitpane dọc và ngang





JSplitPane


```
JFrame fr = new JFrame("JSplitPane Demo");
fr.setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);
JSplitPane splitPane;
JLabel picture = new JLabel();;
String[] imageNames = { "Bird", "Cat", "Dog", "Rabbit", "Pig", "dukeWaveRed",
                        "kathyCosmo", "lainesTongue", "left", "middle", "right", "stickerface"};

//Create the list of images and put it in a scroll pane.
JList list = new JList(imageNames);
list.setSelectionMode(ListSelectionModel.SINGLE_SELECTION);
list.setSelectedIndex(0);
MyListSelectionListener listener = new MyListSelectionListener(picture, imageNames);
list.addListSelectionListener(listener);
JScrollPane listScrollPane = new JScrollPane(list);
```

JSplitPane


```
// Create a split pane with the two scroll panes in it.
splitPane = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT, listScrollPane, pictureScrollPane);
splitPane.setOneTouchExpandable(true);
splitPane.setDividerLocation(150);

// Provide minimum sizes for the two components in the split pane.
Dimension minimumSize = new Dimension(100, 50);
listScrollPane.setMinimumSize(minimumSize);
pictureScrollPane.setMinimumSize(minimumSize);

//Provide a preferred size for the split pane.
splitPane.setPreferredSize(new Dimension(400, 200));

fr.getContentPane().add(splitPane);
fr.pack();
fr.setVisible(true);
```

JSplitPane

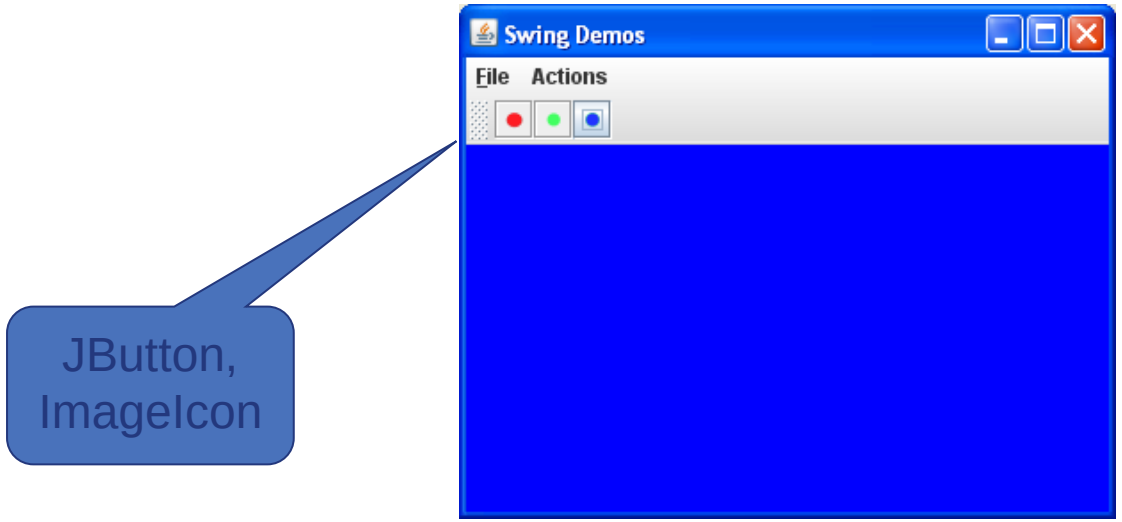
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

```
public class MyListSelectionListener implements ListSelectionListener {
    String[] imageNames;
    JLabel picture;

    MyListSelectionListener(JLabel pic, String[] images) {
        picture = pic;
        imageNames = images;
    }

    public void valueChanged(ListSelectionEvent e) {
        JList list = (JList)e.getSource();
        ImageIcon icon = new ImageIcon(imageNames[list.getSelectedIndex()] + ".gif");
        picture.setIcon(icon);

    }
}
```

JToolBar, Icon, ImageIcon

- **JToolBar** là một nhóm các thành phần thường được sử dụng như các Button hoặc ComboBox
  - Người dùng có thể kéo JToolBar đến các vị trí khác nhau trong màn hình ứng dụng
- **Icon** là hình ảnh kích thước cố định nhỏ, thường được sử dụng để trang trí các thành phần
- **ImageIcon** là một lớp hiện thực Icon Iterface, để vẽ các Icon từ các Image

JToolBar, Icon, ImageIcon




```
// Create toolbar
JToolBar toolbar = new JToolBar();
MainToolBarListener actionListener = new MainToolBarListener(this);
Icon red = new ImageIcon("images/red.png");
redIcon = new JButton(red);
redIcon.addActionListener(actionListener);
toolbar.add(redIcon);
```

JToolBar, Icon, ImageIcon

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

```
// Create toolbar
JToolBar toolbar = new JToolBar();
MainToolBarListener actionListener = new MainToolBarListener(this);
Icon red = new ImageIcon("images/red.png");
redIcon = new JButton(red);
redIcon.setToolTipText("Set red color for the background");
redIcon.addActionListener(actionListener);
toolbar.add(redIcon);
```

Đặt tooltip cho Icon trên thanh toolbar

NỘI DUNG

1. Một số component thông dụng
2. **Look And Feel**
3. Đồ họa trong Java

Pluggable Look And Feel (L&F)

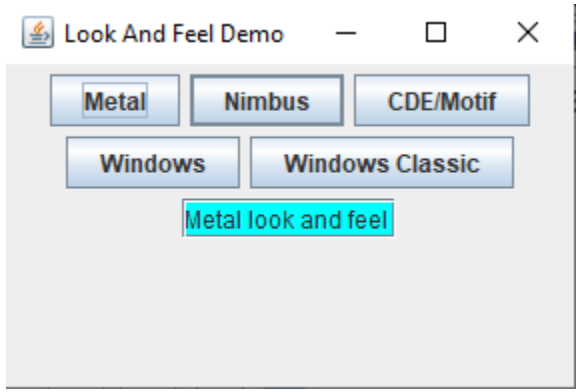
- L&F thiết lập giao diện tổng thể của GUI với nhiều phong cách giao diện khác nhau
- "Look" đề cập cách hiển thị của các tiện ích GUI (chính thức hơn là JComponents) và "Feel" đề cập đến cách các tiện ích hoạt động
- Lớp liên quan và một số phương thức
  - UIManager: Lớp quản lý L&F
  - UIManager.*getCrossPlatformLookAndFeelClassName*(): get L&F
  - UIManager.*setLookAndFeel*(): Set L&F của component
  - UIManager.*getInstalledLookAndFeels*(): Trả về một mảng L&F info đại diện cho các triển khai L&F hiện có

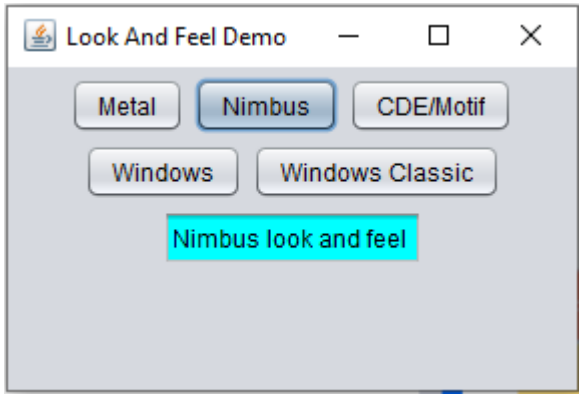
Ví dụ

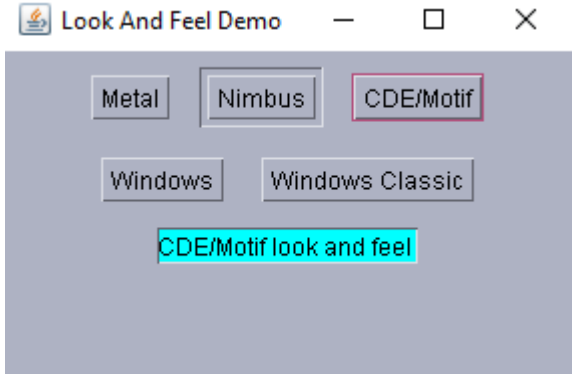
UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

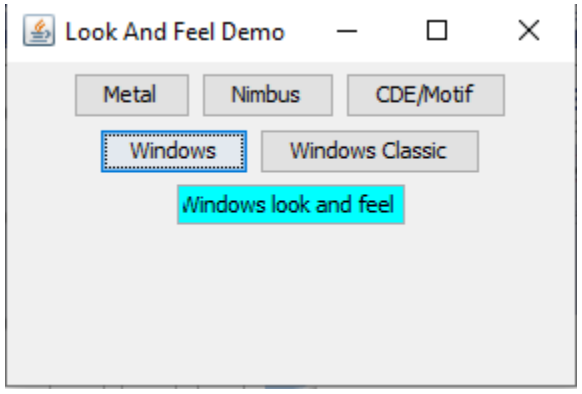
```
LookAndFeelInfo[] installedLookAndFeels = UIManager.getInstalledLookAndFeels()
for(LookAndFeelInfo lookAndFeel : installedLookAndFeels) {
    final String name = lookAndFeel.getName();
    final String className = lookAndFeel.getClassName();
    JButton button = new JButton(name);
    panel.add(button);
    button.addActionListener(new ActionListener()
    {
       public void actionPerformed(ActionEvent event)
       {
          try
          {
             UIManager.setLookAndFeel(className);
             SwingUtilities.updateComponentTreeUI(LookAndFeel.this);
             textField.setText(name + " look and feel");
          }
          catch (Exception e)
          {
             e.printStackTrace();
          }
       }
    });
}
```

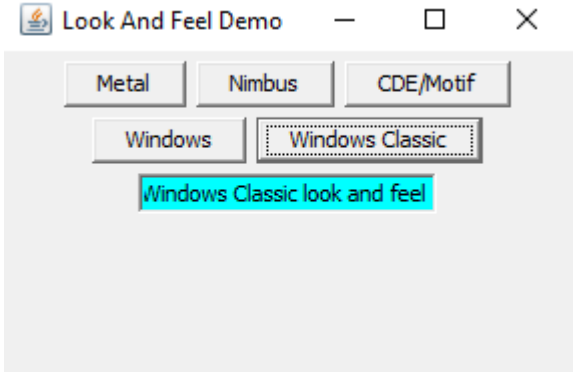
Ví dụ


Look And Feel Demo
Metal
Nimbus
CDE/Motif
Windows
Windows Classic
Metal look and feel


Look And Feel Demo
Metal
Nimbus
CDE/Motif
Windows
Windows Classic
Nimbus look and feel


Look And Feel Demo
Metal
Nimbus
CDE/Motif
Windows
Windows Classic
CDE/Motif look and feel


Look And Feel Demo
Metal
Nimbus
CDE/Motif
Windows
Windows Classic
Windows look and feel


Look And Feel Demo
Metal
Nimbus
CDE/Motif
Windows
Windows Classic
Windows Classic look and feel

NỘI DUNG

1. Một số component thông dụng
2. Look And Feel
3. **Đồ họa trong Java**

Graphics Context và Object

UIT
TRƯỜNG ĐẠI HỌC
CÔNG NGHỆ THÔNG TIN

- Graphics context
  - Hỗ trợ thao tác vẽ trên màn hình
  - Đối tượng Graphics quản lý graphics context
    - Điều khiển cách vẽ
    - Cung cấp các phương thức để vẽ, chọn font, màu….
  - Graphics là 1 lớp trừu tượng!
- Class Component
  - Là lớp cơ sở của các thành phần trong java.awt và javax.swing
  - Phương thức *paint*(Graphics g)

Lớp Color

- Hỗ trợ các thao tác trên màu sắc
- *Color*(int red, int green, int blue)
- Lớp Graphics:
  - void *setColor*(Color c): chọn màu dùng để vẽ
  - Color *getColor*(): lấy về màu đang chọn

Lớp Font

- *Font*(String name, int style, int size)
  - Name: tên font có trong hệ thống
  - Style: FONT.PLAIN, FONT.ITALIC, FONT.BOLD
  - Size: kích thước đơn vị point (1/72 inch)
- Lớp Graphics
  - Font *getFont*()
  - void *setFont*(Font f)

Lớp Graphics

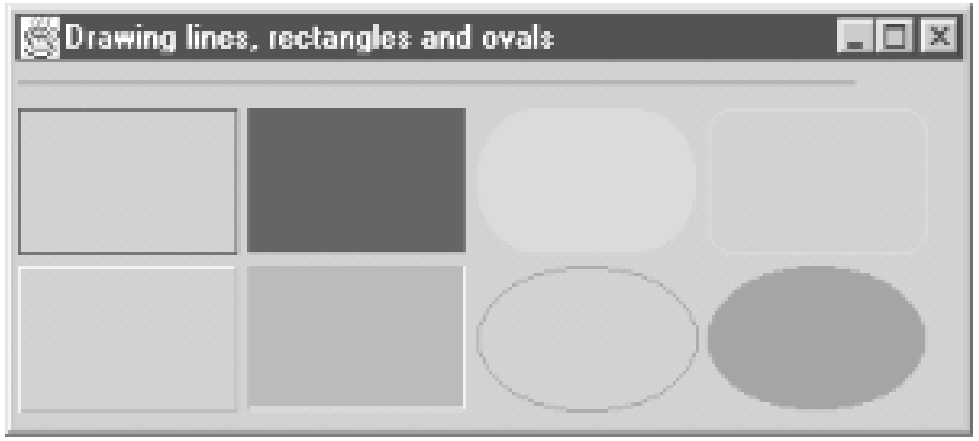
- *drawString*(s, x, y): Vẽ choỗi s tại vị trí (x,y)
- *drawLine*(x1, y1, x2, y2): vẽ đoạn thẳng từ vị trí (x1,y1) đến vị trí (x2,y2)
- *drawRect*(x1, y1, width, height): Vẽ hình chữ nhật với (x1, y1) là đỉnh trái-trên, cùng chiều dài và chiều rộng
- *fillRect*(x1, y1, width, height): Tô hình chữ nhật với (x1, y1) là đỉnh trái-trên, cùng chiều dài và chiều rộng với màu hiện tại
- *clearRect*(x1, y1, width, height): Tương tự như trên, tô hình chữ nhật với background color

Lớp Graphics

- *draw3DRect*(x1, y1, width, height, isRaised)
- *fill3DRect*(x1, y1, width, height, isRaised)
- *drawRoundRect*(x, y, width, height, arcWidth, arcHeight)
- *fillRoundRect*(x, y, width, height, arcWidth, arcHeight)
- *drawOval*(x, y, width, height)
- *fillOval*(x, y, width, height)

Ví Dụ

```
public class LinesRectsOvals extends JFrame {
  private String s = "Using drawString!";
  public LinesRectsOvals()  {
    super( "Drawing lines, rectangles and ovals" );
    setSize( 400, 165 );
    setVisible(true);
  }
  public void paint( Graphics g )   {
    g.setColor( Color.red );
    g.drawLine( 5, 30, 350, 30 );
    g.setColor( Color.blue );
    g.drawRect( 5, 40, 90, 55 );
    g.fillRect( 100, 40, 90, 55 );
    g.setColor( Color.cyan );
    g.fillRoundRect( 195, 40, 90, 55, 50, 50 );
    g.drawRoundRect( 290, 40, 90, 55, 20, 20 );
    g.setColor( Color.yellow );
    g.draw3DRect( 5, 100, 90, 55, true );
    g.fill3DRect( 100, 100, 90, 55, false );

    g.setColor( Color.magenta );
    g.drawOval( 195, 100, 90, 55 );
    g.fillOval( 290, 100, 90, 55 );
  }

  public static void main( String args[] ) {
    LinesRectsOvals app = new LinesRectsOvals();
  }
}
```



Tóm tắt bài học

- JTextField là ô nhập dữ liệu dạng văn bản trên 1 dòng
- JTextArea cho phép người sử dụng nhập vào nhiều dòng văn bản
- JComboBox dùng để liệt kê danh sách các mục mà người dùng có thể chọn, còn được gọi là drop-down list
- Jlist giúp quản lý danh sách các mục chọn, có thể chọn 1 hoặc nhiều mục
- JTabbedPane được sử dụng để chuyển đổi giữa một nhóm các thành phần bằng cách nhấp vào tab có tiêu đề hoặc biểu tượng nhất định
- JSplitPane được sử dụng để phân chia hai thành phần và có thể thay đổi kích thước các thành phần
- JToolBar là một nhóm các thành phần thường được sử dụng như các Button hoặc ComboBox

Tóm tắt bài học

- Icon là hình ảnh kích thước cố định nhỏ, thường được sử dụng để trang trí các thành phần
- ImageIcon là một lớp hiện thực Icon Iterface, để vẽ các Icon từ các Image
- Swing hỗ trợ look-and-feel giúp thay đổi phong cách hiển thị giao diện nhằm phù hợp với nhiều ngữ cảnh/nền tảng khác nhau
- Graphics context hỗ trợ thao tác vẽ trên màn hình. Đối tượng Graphics quản lý graphics context
- Class Component là lớp cơ sở của các thành phần trong Swing, có phương thức paint(Graphics g)
- Lớp Font giúp quản lý các font trong chương trình
- Graphics hỗ trợ vẽ các đối tượng hình (điểm, đoạn thẳng, hình chữ nhật,..) và các thao tác đồ họa