import javax.swing.\*;

import java.awt.\*;

/\*\*

\* A sample frame to illustrate the placing of nested panels.

\* This class illustrates the building of user interface only,

\* logic control for actually playing the game is NOT included.

\*/

class Ch14NestedPanels1 extends JFrame {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 500;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 350;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14NestedPanels1 frame = new Ch14NestedPanels1();

frame.setVisible(true);

}

//----------------------------------

// Constructors

//----------------------------------

/\*\*

\* Default constructor

\*/

public Ch14NestedPanels1() {

Container contentPane;

Ch14TicTacToePanel gamePanel;

JPanel controlPanel;

JPanel scorePanel;

//set the frame properties

setSize (FRAME\_WIDTH, FRAME\_HEIGHT);

setTitle ("Program Ch14NestedPanels1");

setLocation (FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN);

contentPane = getContentPane( );

contentPane.setLayout(new BorderLayout(10, 0));

gamePanel = new Ch14TicTacToePanel();

gamePanel.setBorder(BorderFactory.createLoweredBevelBorder());

controlPanel = new JPanel();

controlPanel.setLayout(new BorderLayout( ));

contentPane.add(gamePanel, BorderLayout.CENTER);

contentPane.add(controlPanel, BorderLayout.EAST);

scorePanel = new JPanel();

scorePanel.setBorder( BorderFactory.createTitledBorder("Scores:"));

scorePanel.setLayout(new GridLayout(2, 2));

scorePanel.add(new JLabel("Player 1:"));

scorePanel.add(new JLabel(" 0"));

scorePanel.add(new JLabel("Player 2:"));

scorePanel.add(new JLabel(" 0"));

controlPanel.add(scorePanel,BorderLayout.NORTH);

controlPanel.add(new JButton("New Game"), BorderLayout.SOUTH);

//register 'Exit upon closing' as a default close operation

setDefaultCloseOperation( EXIT\_ON\_CLOSE );

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

/\*\*

\* Ch14JCheckBoxSample1 class

\*

\* <p>

\* A sample frame to illustrate the use of checkbox buttons.

\*/

class Ch14JCheckBoxSample1 extends JFrame implements ActionListener {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 300;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 200;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

/\*\*

\* An array of JCheckBox objects

\*/

private JCheckBox[] checkBox;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14JCheckBoxSample1 frame = new Ch14JCheckBoxSample1();

frame.setVisible(true);

}

//----------------------------------

// Constructors

//----------------------------------

/\*\*

\* Default constructor

\*/

public Ch14JCheckBoxSample1() {

Container contentPane;

JPanel checkPanel, okPanel;

JButton okButton;

String[] btnText = {"Java", "C++", "Smalltalk", "Ada"};

//set the frame properties

setSize (FRAME\_WIDTH, FRAME\_HEIGHT);

setTitle ("Program Ch14JCheckBoxSample1");

setLocation (FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN);

contentPane = getContentPane( );

contentPane.setBackground(Color.WHITE);

contentPane.setLayout(new BorderLayout());

//create and place four checkboxes

checkPanel = new JPanel(new GridLayout(0,1));

checkPanel.setBorder(BorderFactory.createTitledBorder("Can Program In"));

checkBox = new JCheckBox[btnText.length];

for (int i = 0; i < checkBox.length; i++) {

checkBox[i] = new JCheckBox(btnText[i]);

checkPanel.add(checkBox[i]);

}

//create and place the OK button

okPanel = new JPanel(new FlowLayout());

okButton = new JButton("OK");

okButton.addActionListener(this);

okPanel.add(okButton);

contentPane.add(checkPanel, BorderLayout.CENTER);

contentPane.add(okPanel, BorderLayout.SOUTH);

//register 'Exit upon closing' as a default close operation

setDefaultCloseOperation( EXIT\_ON\_CLOSE );

}

public void actionPerformed(ActionEvent event) {

StringBuffer skill = new StringBuffer("You can program in\n");

for (int i = 0; i < checkBox.length; i++) {

if (checkBox[i].isSelected()) {

skill.append(checkBox[i].getText() + "\n");

}

}

JOptionPane.showMessageDialog(this, skill.toString());

}

}

**Now go over the next program!**

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

/\*\*

\* Ch14JCheckBoxSample2 class

\*

\* <p>

\* A sample frame to illustrate the use of checkbox buttons. This sample

\* will process the ItemEvent events generated by the JCheckBox buttons.

\*/

class Ch14JCheckBoxSample2 extends JFrame

implements ActionListener,

ItemListener {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 300;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 200;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

/\*\*

\* An array of JCheckBox objects

\*/

private JCheckBox[] checkBox;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14JCheckBoxSample2 frame = new Ch14JCheckBoxSample2();

frame.setVisible(true);

}

//----------------------------------

// Constructors

//----------------------------------

/\*\*

\* Default constructor

\*/

public Ch14JCheckBoxSample2() {

Container contentPane;

JPanel checkPanel, okPanel;

JButton okButton;

String[] btnText = {"Java", "C++", "Smalltalk", "Ada"};

//set the frame properties

setSize (FRAME\_WIDTH, FRAME\_HEIGHT);

setTitle ("Program Ch14JCheckBoxSample2");

setLocation (FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN);

contentPane = getContentPane( );

contentPane.setBackground(Color.WHITE);

contentPane.setLayout(new BorderLayout());

//create and place four checkboxes

checkPanel = new JPanel(new GridLayout(0,1));

checkPanel.setBorder(BorderFactory.createTitledBorder("Can Program In"));

checkBox = new JCheckBox[btnText.length];

for (int i = 0; i < checkBox.length; i++) {

checkBox[i] = new JCheckBox(btnText[i]);

checkPanel.add(checkBox[i]);

checkBox[i].addItemListener(this);

}

//create and place the OK button

okPanel = new JPanel(new FlowLayout());

okButton = new JButton("OK");

okButton.addActionListener(this);

okPanel.add(okButton);

contentPane.add(checkPanel, BorderLayout.CENTER);

contentPane.add(okPanel, BorderLayout.SOUTH);

//register 'Exit upon closing' as a default close operation

setDefaultCloseOperation( EXIT\_ON\_CLOSE );

}

public void actionPerformed(ActionEvent event) {

StringBuffer skill = new StringBuffer("You can program in\n");

for (int i = 0; i < checkBox.length; i++) {

if (checkBox[i].isSelected()) {

skill.append(checkBox[i].getText() + "\n");

}

}

JOptionPane.showMessageDialog(this, skill.toString());

}

public void itemStateChanged(ItemEvent event) {

JCheckBox source = (JCheckBox) event.getSource();

String state;

if (event.getStateChange() == ItemEvent.SELECTED) {

state = "is selected";

} else {

state = "is deselected";

}

JOptionPane.showMessageDialog(this, "JCheckBox '" +

source.getText() +

"' " + state);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

/\*\*

\*

\* <p>

\* A sample frame to illustrate the use of JComboBox.

\*/

class Ch14JComboBoxSample extends JFrame

implements ActionListener,

ItemListener {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 300;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 200;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

/\*\*

\* A combo box of programming languages

\*/

private JComboBox comboBox;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14JComboBoxSample frame = new Ch14JComboBoxSample();

frame.setVisible(true);

}

//----------------------------------

// Constructors

//----------------------------------

/\*\*

\* Default constructor

\*/

public Ch14JComboBoxSample() {

Container contentPane;

JPanel comboPanel, okPanel;

JButton okButton;

String[] comboBoxItem = {"Java", "C++", "Smalltalk", "Ada"};

//set the frame properties

setSize (FRAME\_WIDTH, FRAME\_HEIGHT);

setTitle ("Program Ch14JComboBoxSample");

setLocation (FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN);

contentPane = getContentPane( );

contentPane.setBackground(Color.WHITE);

contentPane.setLayout(new BorderLayout());

//create and place a combo box

comboPanel = new JPanel(new FlowLayout());

comboPanel.setBorder(

BorderFactory.createTitledBorder("Pick your favorite"));

comboBox = new JComboBox(comboBoxItem);

comboBox.addItemListener(this);

comboPanel.add(comboBox);

//create and place the OK button

okPanel = new JPanel(new FlowLayout());

okButton = new JButton("OK");

okButton.addActionListener(this);

okPanel.add(okButton);

contentPane.add(comboPanel, BorderLayout.CENTER);

contentPane.add(okPanel, BorderLayout.SOUTH);

//register 'Exit upon closing' as a default close operation

setDefaultCloseOperation( EXIT\_ON\_CLOSE );

}

public void actionPerformed(ActionEvent event) {

String favorite;

int loc;

favorite = (String) comboBox.getSelectedItem();

loc = comboBox.getSelectedIndex();

JOptionPane.showMessageDialog(this, "Currently selected item '" +

favorite + "' is at index position " + loc);

}

public void itemStateChanged(ItemEvent event) {

String state;

if (event.getStateChange() == ItemEvent.SELECTED) {

state = "is selected";

} else {

state = "is deselected";

}

JOptionPane.showMessageDialog(this, "JComboBox Item '" +

event.getItem() +

"' " + state);

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

/\*\*

\* Ch14JListSample class

\*

\* <p>

\* A sample frame to illustrate the use of JList.

\*/

class Ch14JListSample extends JFrame

implements ActionListener {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 300;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 250;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

/\*\*

\* A JList of three-letter animal names

\*/

private JList list;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14JListSample frame = new Ch14JListSample();

frame.setVisible(true);

}

//----------------------------------

// Constructors

//----------------------------------

/\*\*

\* Default constructor

\*/

public Ch14JListSample() {

Container contentPane;

JPanel listPanel, okPanel;

JButton okButton;

String[] names = {"Ape", "Bat", "Bee", "Cat",

"Dog", "Eel", "Fox", "Gnu",

"Hen", "Man", "Sow", "Yak"};

//set the frame properties

setSize (FRAME\_WIDTH, FRAME\_HEIGHT);

setTitle ("Program Ch14JListSample2");

setLocation (FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN);

contentPane = getContentPane( );

contentPane.setBackground(Color.WHITE);

contentPane.setLayout(new BorderLayout());

//create and place a JList

listPanel = new JPanel(new GridLayout(0,1));

listPanel.setBorder(BorderFactory.createTitledBorder(

"Three-letter Animal Names"));

list = new JList(names);

listPanel.add(new JScrollPane(list));

list.setSelectionMode(ListSelectionModel.MULTIPLE\_INTERVAL\_SELECTION);

//this is default, so the explicit call is not necessary

//---------------------------------------

//Other selection modes

//

//list.setSelectionMode(ListSelectionModel.SINGLE\_INTERVAL\_SELECTION);

//list.setSelectionMode(ListSelectionModel.SINGLE\_SELECTION);

//---------------------------------------

//create and place the OK button

okPanel = new JPanel(new FlowLayout());

okButton = new JButton("OK");

okButton.addActionListener(this);

okPanel.add(okButton);

contentPane.add(listPanel, BorderLayout.CENTER);

contentPane.add(okPanel, BorderLayout.SOUTH);

//register 'Exit upon closing' as a default close operation

setDefaultCloseOperation( EXIT\_ON\_CLOSE );

}

public void actionPerformed(ActionEvent event) {

Object[] name;

int[] loc;

name = list.getSelectedValues();

loc = list.getSelectedIndices();

System.out.println("Currently selected animal names are ");

for (int i = 0; i < name.length; i++) {

System.out.println((String)name[i] + " at position " + loc[i]);

}

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

**For SplitPane – Code from an internet site (**[**https://www.geeksforgeeks.org/**](https://www.geeksforgeeks.org/)**). I tested it. It Does Work!**

// Java Program to create a horizontal JSplitPane

// to separate two text areas

import javax.swing.event.\*;

import java.awt.\*;

import javax.swing.\*;

class solve extends JFrame {

// frame

static JFrame f;

// text areas

static JTextArea t1, t2;

// main class

public static void main(String[] args)

{

// create a new frame

f = new JFrame("frame");

// create a object

solve s = new solve();

// create a panel

JPanel p1 = new JPanel();

JPanel p = new JPanel();

// create text areas

t1 = new JTextArea(10, 10);

t2 = new JTextArea(10, 10);

// set texts

t1.setText("this is first text area");

t2.setText("this is second text area");

// add text area to panel

p1.add(t1);

p.add(t2);

// create a splitpane

JSplitPane sl = new JSplitPane(SwingConstants.HORIZONTAL, p1, p);

// add panel

f.add(sl);

// set the size of frame

f.setSize(300, 300);

f.show();

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import javax.swing.\*;

import java.awt.event.\*;

import java.io.\*;

/\*\*

\* Ch14TrackMouseFrame

\*

\* This program tracks the mouse click events. When a mouse button (any

\* mouse button supported by the platform) is clicked, the

\* location where the mouse button is clicked is displayed.

\*/

class Ch14TrackMouseFrame extends JFrame implements MouseListener {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 450;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 300;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

/\*\*

\* Constant for mouse button double click

\*/

private static final int DOUBLE\_CLICK = 2;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14TrackMouseFrame frame = new Ch14TrackMouseFrame();

frame.setVisible(true);

}

//----------------------------------

// Constructors

//----------------------------------

/\*\*

\* Default constructor

\*/

public Ch14TrackMouseFrame() {

//set frame properties

setTitle ("TrackMouseFrame");

setSize (FRAME\_WIDTH, FRAME\_HEIGHT);

setResizable (false);

setLocation (FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN);

//to change the background color, do

//setBackground(Color.white);

//((JPanel)getContentPane()).setOpaque(false);

// or

// ((JPanel)getContentPane()).setBackground(Color.white);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

//register self as a mouse event listener

addMouseListener( this );

}

//-------------------------------------------------

// Public Methods:

//

// void mouseClicked ( MouseEvent )

//

// void start ( )

//

//------------------------------------------------

public void mouseClicked(MouseEvent event) //**part of MouseListener**

{

if (event.getClickCount() == DOUBLE\_CLICK) {

System.exit(0);

} else {

int x, y;

x = event.getX(); //get the x and y coordinates of

y = event.getY(); //the mouse click point

System.out.println("[" + x + "," + y + "]");

}

}

public void mouseEntered ( MouseEvent event ) { }

public void mouseExited ( MouseEvent event ) { }

public void mousePressed ( MouseEvent event ) {

System.out.println("Down");

}

public void mouseReleased ( MouseEvent event ) {

System.out.println("Up");

}

}

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

import javax.swing.\*;

import java.awt.\*;

import java.awt.event.\*;

class Ch14SketchPad extends JFrame

implements MouseListener, MouseMotionListener {

//----------------------------------

// Data Members

//----------------------------------

/\*\*

\* Default frame width

\*/

private static final int FRAME\_WIDTH = 450;

/\*\*

\* Default frame height

\*/

private static final int FRAME\_HEIGHT = 300;

/\*\*

\* X coordinate of the frame default origin point

\*/

private static final int FRAME\_X\_ORIGIN = 150;

/\*\*

\* Y coordinate of the frame default origin point

\*/

private static final int FRAME\_Y\_ORIGIN = 250;

/\*\*

\* Last x position of the mouse dragging

\*/

private int last\_x;

/\*\*

\* Last y position of the mouse dragging

\*/

private int last\_y;

//----------------------------------

// Main method

//----------------------------------

public static void main(String[] args) {

Ch14SketchPad frame = new Ch14SketchPad();

frame.setVisible(true);

}

//--------------------------------

// Constructor

//--------------------------------

public Ch14SketchPad( ) {

//set frame properties

setTitle ("Chapter 14 SketchPad");

setSize ( FRAME\_WIDTH, FRAME\_HEIGHT );

setResizable( false );

setLocation ( FRAME\_X\_ORIGIN, FRAME\_Y\_ORIGIN );

//to change the background color, do

//

// ((JPanel)getContentPane()).setBackground(Color.white);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

last\_x = last\_y = 0;

addMouseListener( this ); //adds itself as mouse and

addMouseMotionListener( this ); //mouse motion listener

}

//--------------------------------

// Mouse Event Handling

//--------------------------------

public void mousePressed( MouseEvent event ) {

int x = event.getX();

int y = event.getY();

if ( event.isMetaDown() ) {

//the right mouse button is pressed, so erase the contents

Graphics g = getGraphics();

Rectangle r = getBounds();

g.clearRect(0, 0, r.width, r.height);

g.dispose();

} else {

//the left mouse button is pressed,

//remember the starting point of a new mouse drag

last\_x = x;

last\_y = y;

}

}

public void mouseClicked ( MouseEvent event ) { }

public void mouseEntered ( MouseEvent event ) { }

public void mouseExited ( MouseEvent event ) { }

public void mouseReleased( MouseEvent event ) { }

//--------------------------------

// Mouse Motion Event Handling

//--------------------------------

public void mouseDragged( MouseEvent event ) {

int x = event.getX();

int y = event.getY();

if ( !event.isMetaDown() ) {

//don’t process the right button drag

Graphics g = getGraphics();

g.drawLine(last\_x, last\_y, x, y);

g.dispose();

last\_x = x;

last\_y = y;

}

}

public void mouseMoved ( MouseEvent event ) { }

}