SAN JACINTO CATHOLIC SCHOOL, INC.

ComProg 5-6-12ICT

San Jacinto, Pangasinan

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**COMPUTER PROGRAMMING 5-6 (ICT 12)**

GRADE 12 LEARNER’S MODULE 3 & 4 – 3rd Quarter, Weeks 5 - 8

**CHAPTER 1: JAVA SWING CONTROLERS**

**Lesson 1: Introduction to the Java Swing Toolkit**

**OBJECTIVES**: In this lesson, you will learn to:

* Determine different library of Java swing.
* Identifying the different packages of Java swing.
* Describe what library is.
* Creating Java JSubMenu program.
* Understanding the use of JSubMenu program.
* Determine the different uses of JCheckMenu.
* Creating a Pop-up menu.

**INTRODUCTION:**

This is an introductory Swing tutorial. The purpose of this tutorial is to get you started with the Java Swing toolkit. The tutorial has been created and tested on Linux.

**INSTRUCTION / DISCUSSION:**

**Lesson 5: Menus and Toolbars in Java Swing**

* **Creating a Menu Bar**

Each menu can also have a submenu. This way we can group similar commnads into groups. For example we can place commands that hide/show various toolbars like personal bar, address bar, status bar or navigation bar into a submenu called toolbars. Within a menu, we can seperate commands with a separator. It is a simple line. It is common practice to separate commands like new, open, save from commands like print, print preview with a single separator. Menus commands can be launched via keyboard shortcuts. For this, we define menu item accelerators.

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import javax.swing.ImageIcon;

import javax.swing.JFrame;

import javax.swing.JMenu;

import javax.swing.JMenuBar;

import javax.swing.JMenuItem;

import javax.swing.KeyStroke;

public class Submenu extends JFrame {

public Submenu() {

setTitle("Submenu");

JMenuBar menubar = new JMenuBar();

ImageIcon iconNew = new ImageIcon("new.png");

ImageIcon iconOpen = new ImageIcon("open.png");

ImageIcon iconSave = new ImageIcon("save.png");

ImageIcon iconClose = new ImageIcon("exit.png");

JMenu file = new JMenu("File");

file.setMnemonic(KeyEvent.VK\_F);

JMenu imp = new JMenu("Import");

imp.setMnemonic(KeyEvent.VK\_M);

JMenuItem newsf = new JMenuItem("Import newsfeed list...");

JMenuItem bookm = new JMenuItem("Import bookmarks...");

JMenuItem mail = new JMenuItem("Import mail...");

imp.add(newsf);

imp.add(bookm);

imp.add(mail);

JMenuItem fileNew = new JMenuItem("New", iconNew);

fileNew.setMnemonic(KeyEvent.VK\_N);

JMenuItem fileOpen = new JMenuItem("Open", iconOpen);

fileNew.setMnemonic(KeyEvent.VK\_O);

JMenuItem fileSave = new JMenuItem("Save", iconSave);

fileSave.setMnemonic(KeyEvent.VK\_S);

JMenuItem fileClose = new JMenuItem("Close", iconClose);

fileClose.setMnemonic(KeyEvent.VK\_C);

fileClose.setToolTipText("Exit application");

fileClose.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_W,

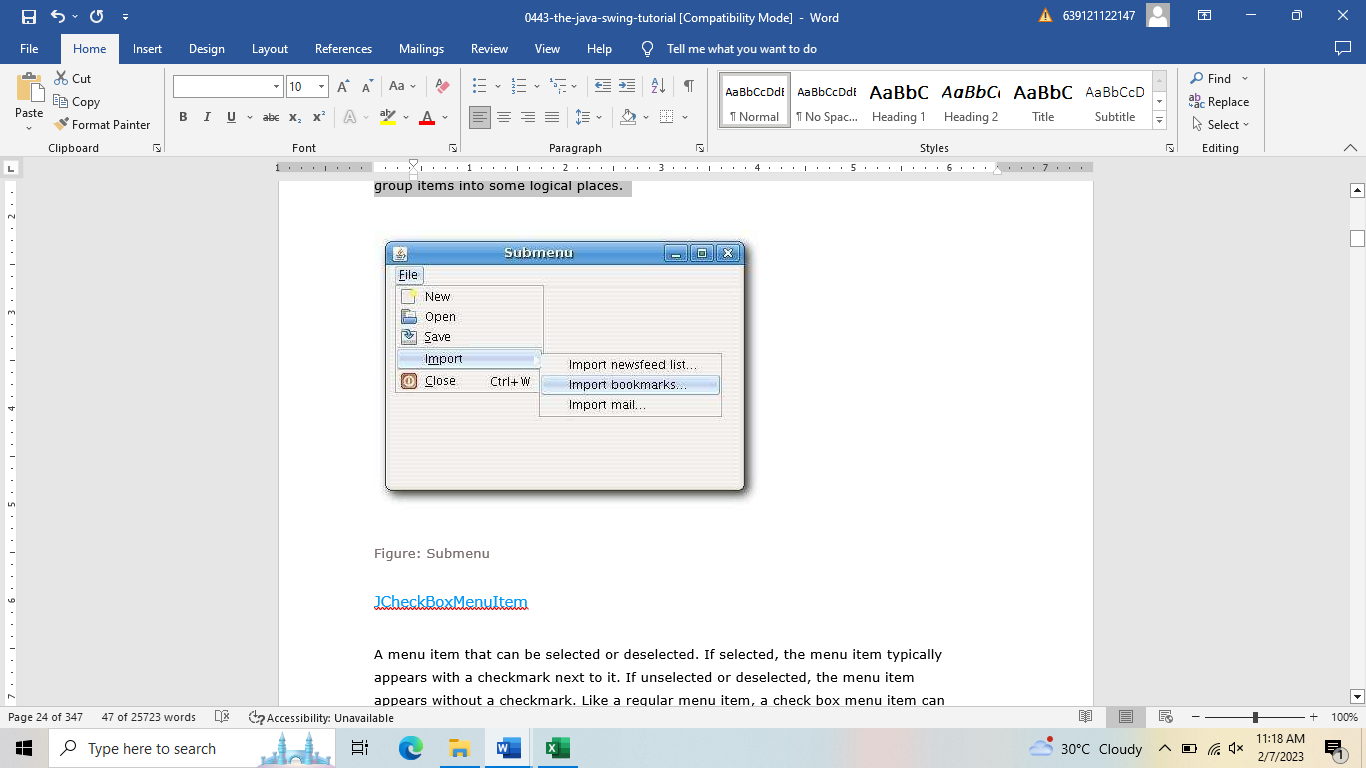
ActionEvent.CTRL\_MASK));

fileClose.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent event) {

System.exit(0);

}

});

file.add(fileNew);

file.add(fileOpen);

file.add(fileSave);

file.addSeparator();

file.add(imp);

file.addSeparator();

file.add(fileClose);

menubar.add(file);

setJMenuBar(menubar);

setSize(360, 250);

setLocationRelativeTo(null);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

}

public static void main(String[] args) {

new Submenu();

}

JMenu imp = new JMenu("Import");

..

file.add(imp);

A submenu is just like any other normal menu. It is created the same way. We simply add a menu to existing menu.

fileClose.setAccelerator(KeyStroke.getKeyStroke(KeyEvent.VK\_W,

ActionEvent.CTRL\_MASK));

An accelerator is a key shortcut that launches a menu item. In our case, by pressing **Ctrl + W** we close the application.

file.addSeparator();

A separator is a vertical line that visually separates the menu items. This way we can group items into some logical places.

JCheckBoxMenuItem

A menu item that can be selected or deselected. If selected, the menu item typically appears with a checkmark next to it. If unselected or deselected, the menu item appears without a checkmark. Like a regular menu item, a check box menu item can have either text or a graphic icon associated with it, or both.

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import java.awt.event.KeyEvent;

import javax.swing.BorderFactory;

import javax.swing.JCheckBoxMenuItem;

import javax.swing.JFrame;

import javax.swing.JLabel;

import javax.swing.JMenu;

import javax.swing.JMenuBar;

import javax.swing.UIManager;

import javax.swing.border.EtchedBorder;

public class CheckMenuItem extends JFrame {

private JLabel statusbar;

public CheckMenuItem() {

setTitle("CheckBoxMenuItem");

JMenuBar menubar = new JMenuBar();

Menu file = new JMenu("File");

file.setMnemonic(KeyEvent.VK\_F);

JMenu view = new JMenu("View");

view.setMnemonic(KeyEvent.VK\_V);

JCheckBoxMenuItem sbar = new JCheckBoxMenuItem("Show StatuBar");

sbar.setState(true);

sbar.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent event) {

if (statusbar.isVisible()) {

statusbar.setVisible(false);

} else { statusbar.setVisible(true);

}

}

});

view.add(sbar);

menubar.add(file);

menubar.add(view);

setJMenuBar(menubar);

statusbar = new JLabel(" Statusbar");

statusbar.setBorder(BorderFactory.createEtchedBorder(

EtchedBorder.RAISED));

add(statusbar, BorderLayout.SOUTH);

setSize(360, 250);

setLocationRelativeTo(null);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

}

public static void main(String[] args) {

new CheckMenuItem();

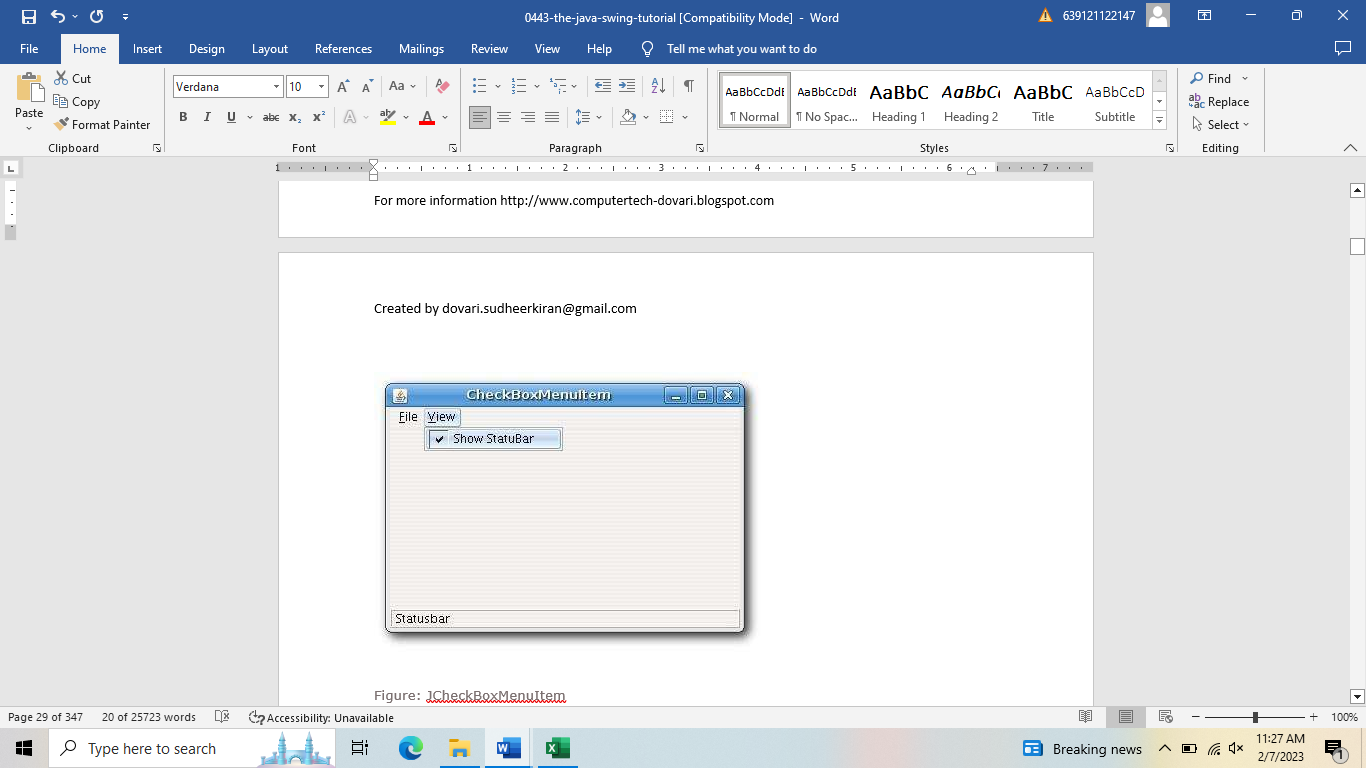
}

}

The example shows a **JCheckBoxMenuItem.**. By selecting the menu item, we toggle the visibility of the statusbar.

JCheckBoxMenuItem sbar = new JCheckBoxMenuItem("Show StatuBar");

sbar.setState(true);

We create the **JCheckBoxMenuItem** and check it by default. The statusbar is initially visible.

if (statusbar.isVisible()) {

statusbar.setVisible(false);

} else {

statusbar.setVisible(true);

}

Here we toggle the visibility of the statusbar.

statusbar = new JLabel(" Statusbar");

statusbar.setBorder(BorderFactory.createEtchedBorder(EtchedBorder.RAISED));

The statusbar is a simple **JLabel** component. We put a raised **EtchedBorder** around the label, so that it is discernible.

**A popup menu**

Another type of a menu is a popup menu. It is sometimes called a context menu. It is usually shown, when we right click on a component. The idea is to provide only the commands, that are relevant to the current context. Say we have an image. By right clicking on the image, we get a window with commands to save, rescale, move etc the image.

import java.awt.Toolkit;

import javax.swing.\*;

import java.awt.event.\*;

public class PopupMenu {

private JPopupMenu menu;

private Toolkit toolkit;

public PopupMenu(){

JFrame frame = new JFrame("JPopupMenu");

frame.setDefaultCloseOperation(JFrame.EXIT\_ON\_CLOSE);

toolkit = frame.getToolkit();

menu = new JPopupMenu();

JMenuItem menuItemBeep = new JMenuItem("Beep");

menuItemBeep.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

toolkit.beep();

}

});

menu.add(menuItemBeep);

JMenuItem menuItemClose = new JMenuItem("Close");

menuItemClose.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent e) {

System.exit(0);

}

});

menu.add(menuItemClose);

frame.addMouseListener(new MouseAdapter() {

public void mouseReleased(MouseEvent e) {

if (e.getButton() == e.BUTTON3) {

menu.show(e.getComponent(), e.getX(), e.getY());

}

}

});

frame.setSize(250, 200);

frame.setLocationRelativeTo(null);

frame.setVisible(true);

}

public static void main(String[] args) {

new PopupMenu();

}

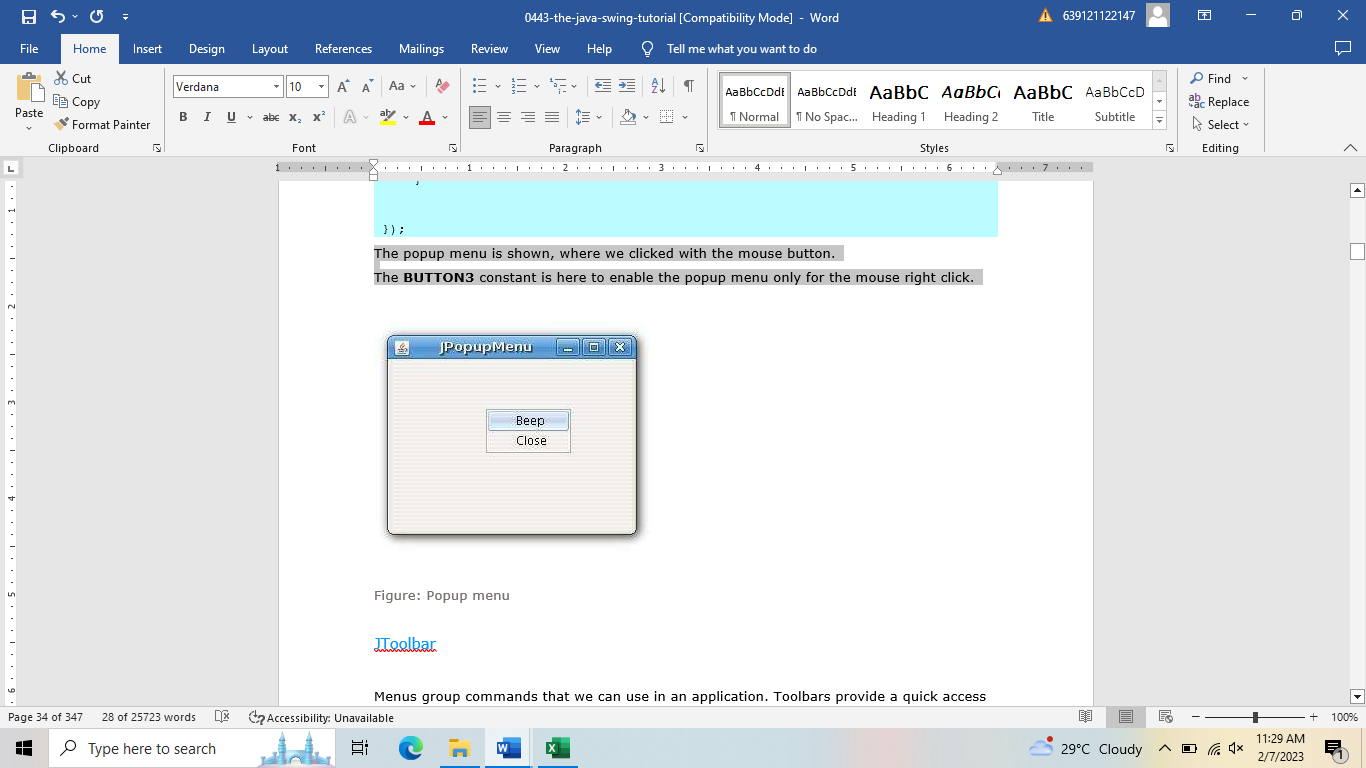
}

Our example shows a demonstrational popup menu with two commands. The first option of the popup menu will beep a sound, the second one will close the window.

In our example, we create a submenu, menu separators and create an accelerator key.

menu = new JPopupMenu();

To create a popup menu, we have a class called **JPopupMenu**.

JMenuItem menuItemBeep = new JMenuItem("Beep");

The menu item is the same, as with the standard **JMenu**

frame.addMouseListener(new MouseAdapter() {

public void mouseReleased(MouseEvent e) {

if (e.getButton() == e.BUTTON3) {

menu.show(e.getComponent(), e.getX(), e.getY());

}

}

});

The popup menu is shown, where we clicked with the mouse button.

The **BUTTON3** constant is here to enable the popup menu only for the mouse right click.

**Lesson 6: Proceeding with the JToolBar**

Menus group commands that we can use in an application. Toolbars provide a quick access to the most frequently used commands. In Java Swing, the **JToolBar** class creates a toolbar in an application.

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JMenu;

import javax.swing.JMenuBar;

import javax.swing.JToolBar;

public class SimpleToolbar extends JFrame {

public SimpleToolbar() {

setTitle("SimpleToolbar");

JMenuBar menubar = new JMenuBar();

JMenu file = new JMenu("File");

menubar.add(file);

setJMenuBar(menubar);

JToolBar toolbar = new JToolBar();

ImageIcon icon = new

ImageIcon(getClass().getResource("exit.png"));

JButton exit = new JButton(icon);

toolbar.add(exit);

exit.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent event) {

System.exit(0);

}

});

add(toolbar, BorderLayout.NORTH);

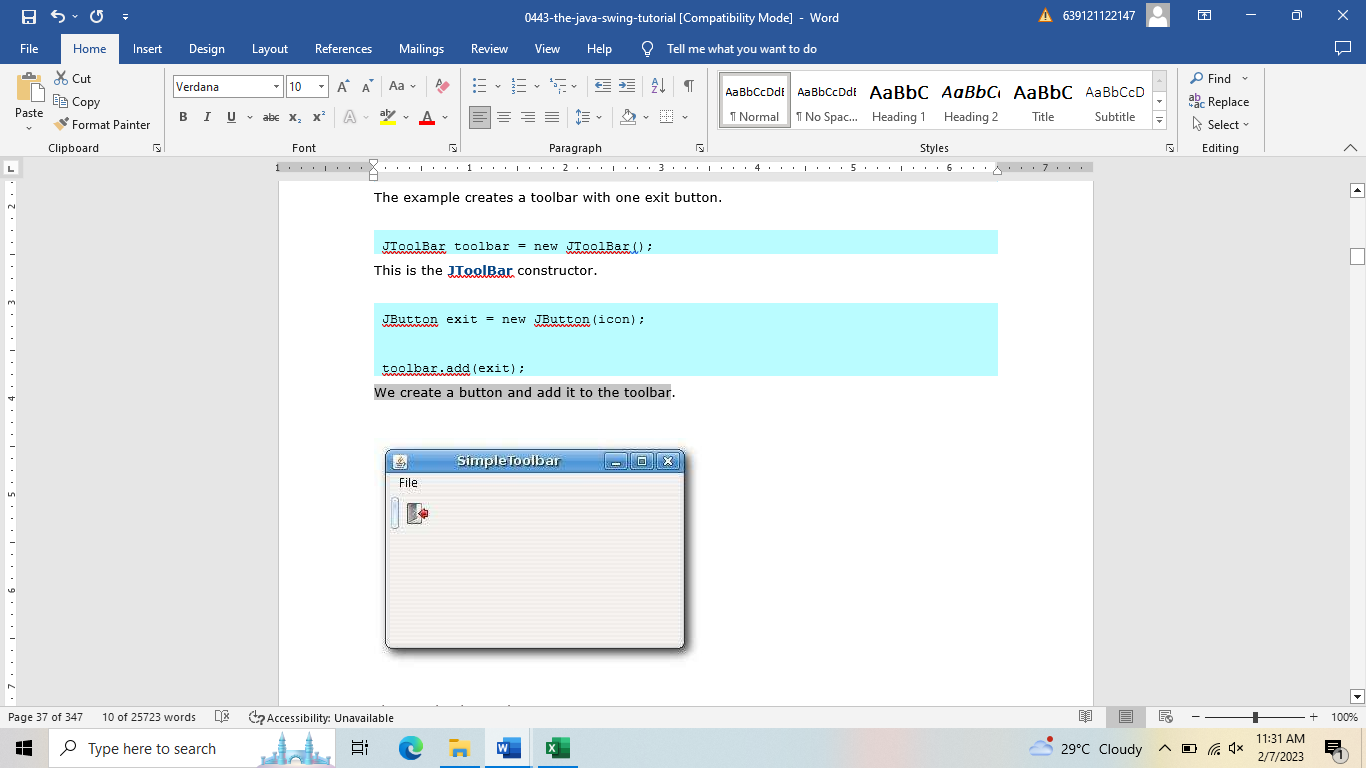
setSize(300, 200);

setLocationRelativeTo(null);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

}

public static void main(String[] args) {

new SimpleToolbar();

}

}

The example creates a toolbar with one exit button.

JToolBar toolbar = new JToolBar();

This is the **JToolBar** constructor.

JButton exit = new JButton(icon);

toolbar.add(exit);

We create a button and add it to the toolbar

***Toolbars***

Say, we wanted to create two toolbars. The next example shows, how we could do it.

import java.awt.BorderLayout;

import java.awt.event.ActionEvent;

import java.awt.event.ActionListener;

import javax.swing.BoxLayout;

import javax.swing.ImageIcon;

import javax.swing.JButton;

import javax.swing.JFrame;

import javax.swing.JPanel;

import javax.swing.JToolBar;

public class Toolbars extends JFrame {

public Toolbars() {

setTitle("Toolbars");

JToolBar toolbar1 = new JToolBar();

JToolBar toolbar2 = new JToolBar();

JPanel panel = new JPanel();

panel.setLayout(new BoxLayout(panel, BoxLayout.Y\_AXIS));

ImageIcon newi = new ImageIcon(

getClass().getResource("new.png"));

ImageIcon open = new ImageIcon(

getClass().getResource("open.png"));

ImageIcon save = new ImageIcon(

getClass().getResource("save.png"));

ImageIcon exit = new ImageIcon(

getClass().getResource("exit.png"));

JButton newb = new JButton(newi);

JButton openb = new JButton(open);

JButton saveb = new JButton(save);

toolbar1.add(newb);

toolbar1.add(openb);

toolbar1.add(saveb);

toolbar1.setAlignmentX(0);

JButton exitb = new JButton(exit);

toolbar2.add(exitb);

toolbar2.setAlignmentX(0);

exitb.addActionListener(new ActionListener() {

public void actionPerformed(ActionEvent event) {

System.exit(0);

}

});

panel.add(toolbar1);

panel.add(toolbar2);

add(panel, BorderLayout.NORTH);

setSize(300, 200);

setLocationRelativeTo(null);

setDefaultCloseOperation(EXIT\_ON\_CLOSE);

setVisible(true);

}

public static void main(String[] args) {

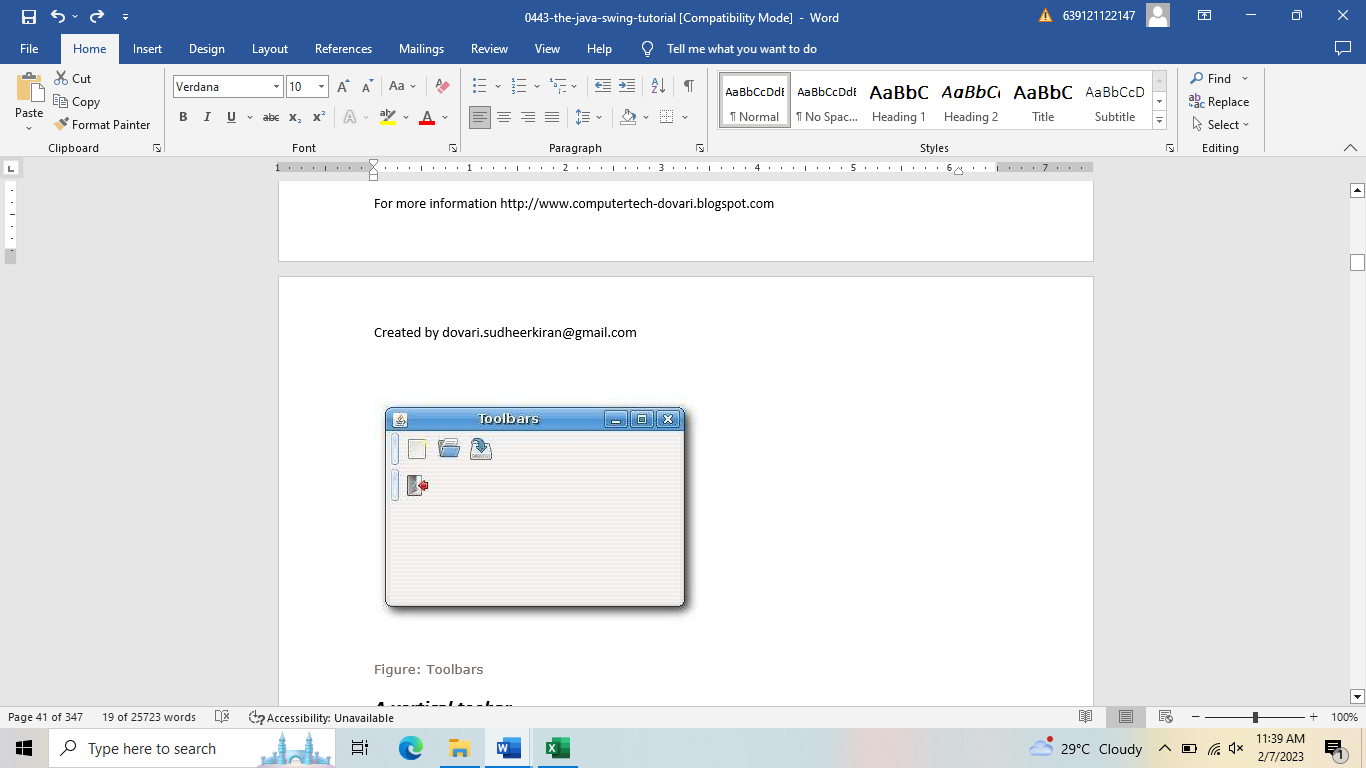
new Toolbars();

}

}

We show only one way, how we could create toolbars. Of course, there are several possibilities. We put a **JPanel** to the north of the **BorderLayout** manager. The panel has a vertical **BoxLayout**. We add the two toolbars into this panel.

JToolBar toolbar1 = new JToolBar();

JToolBar toolbar2 = new JToolBar();

Creation of two toolbars.

JPanel panel = new JPanel();

panel.setLayout(new BoxLayout(panel, BoxLayout.Y\_AXIS));

The panel has a vertical **BoxLayout**.

toolbar1.setAlignmentX(0);

The toolbar is left aligned.

panel.add(toolbar1);

panel.add(toolbar2);

add(panel, BorderLayout.NORTH);

We add the toolbars to the panel. Finally, the panel is located into the north part of the frame.

**References:** TLE-TVL Series: Programming Java

Phoenix Publishing House

Prepared by: Mr. Christian U. de Guzman Checked by: Mrs. Lailanie L. De Guzman, M.E.d

Subject Teacher Gr.12 Coordinator

SAN JACINTO CATHOLIC SCHOOL, INC.

**Computer Programming 5 (ICT 12) – Evaluation for Module # 3**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Gr. 12 ICT:St. Gabriel Date:\_\_\_\_\_\_\_ Score: \_\_\_\_

**WRITTEN WORK:**

**Activity 1:** Write True if the statement is correct and False if it is not. Write your answer on the space provide.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_1. Menus commands can be launched via keyboard function keys.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_2. It is common practice to combine commands like new, open, save from commands like print, print preview with a single separator.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_3. Each menu doesn’t consist of submenu.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_4. If unselected or deselected, the menu item appears with a checkmark.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_5. We use “.java” for our file type.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_6. When compiling a file we probably use “javac”.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_7. A separator is a horizontal line that visually separates the menu items.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_8. Errors can be fixed thru gmail.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_9. Class name represent the file name of your folder.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_10. A submenu is a special type of menu.

**Activity 2:** Identification. Identify the following sentences. Write your answer on the space provided.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1. An \_\_\_\_\_\_\_ is a key shortcut that launches a menu item.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2. The \_\_\_\_\_\_\_\_\_\_ is a simple **JLabel** component.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3. A menu item that can be selected or deselected.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4. If selected, the menu item typically appears with a \_\_\_\_ next to it.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5. It is sometimes called a context menu.

**PERFORMANCE TASKS**

**A.** PERFORM. Draw an example of interface for Enrolment Registration and explain on how to use it. (30 points)

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Activity 1**: Hands-On Activity: Create the following programming below.

Instruction: 1. This activity is for individual only.

2. Cellphones are prohibited to use during hands-on.

3. Only notes are allowed to use during the hands-on.

4. Each activity should be program while teacher is doing your hands-on.

5. After taking the activities write your program inside of the box and draw the output beside.

A. Create a program on how to use the Image Icon.

B. Create a program with Sub-Menu.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Excellent(10) | Good(8) | Satisfactory(5) | Needs Improvement(1) |
| Following Activity Directions | All directions were exceeded | You followed most directions | You followed some directions | None of the directions were followed |
| Use of Creativity | You used your own ideas and imagination | You used your own ideas most of the time | You used some imagination | You did not use your own ideas or imagination |
| Effort put into Activity | You took your time and worked hard on the activity | You worked hard for most of the time | You put a small effort into the activity | You rushed through and did not work hard |

SAN JACINTO CATHOLIC SCHOOL, INC.

**Computer Programming 5-6 (ICT 12) – Evaluation for Module # 4**

Name: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Gr. 12 ICT:St. Gabriel Date:\_\_\_\_\_\_\_ Score: \_\_\_\_

**WRITTEN WORK:**

**Activity 1:** Write a program on how to create a simple GUI in Java. Write your program inside the box .(15 points)

**Activity 2:** Identification. Identify the following sentences. Write your answer on the space provided.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_1. GUI stands for \_\_\_\_\_\_\_\_\_.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2. File type of java.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_3. Shortcut key to save a file in java.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_4. Shortcut key to copy a file.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_5. Shortcut key to paste a file.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_6. Shortcut key to select all files.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_7. Shortcut key to undo previous action.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_8. Shortcut key to create a new file.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_9. It is a color black screen where you compile your program in Java.

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_10. File type of Java program.

**PERFORMANCE TASKS**

**A.** PERFORM. Show the output the following code. (30 points)

a. Present your System in the class and discuss on how it will works. Show the interface of your program iside of the box.

**Activity 2**: Hands-On Activity: Create the following programming below.

Instruction: 1. This activity is for individual only.

2. Cellphones are prohibited to use during hands-on.

3. Only notes are allowed to use during the hands-on.

4. Each activity should be program while teacher is doing your hands-on.

A. Create a program by using SWING and insert two buttons.

B. Create a program by using SWING title on it.