**LAB : 3**

**OBJECTIVE :**

Create an application which

1. draws basic graphical primitives on the screen.

2. draws a bar graph to display. Data values can be given at int[]array.

3. examines a phone number, that a user entered in a given format. \*Area code should be one of the following: 040, 041, 050, and 0400,044

**Requrements :**

(a) Windows PC (Windows 7/8/10) / Mac

(b) JDK 1.5

(c) Java Wireless Toolkit 2.5.2

**Implementation :**

1. Draws basic graphical primitives on the screen

**GraphicsPremitives.java :**

import javax.microedition.lcdui.\*;

import javax.microedition.midlet.\*;

import javax.microedition.io.\*;

import java.io.\* ;

import java.lang.\*;

public class GraphicsPremitives extends MIDlet implements CommandListener {

Display display;

Form form;

List menu;

Ticker ticker;

static final Command backCommand = new Command("Back", Command.BACK, 0);

static final Command exitCommand = new Command("Exit", Command.STOP, 1);

String currentlyAt;

public GraphicsPremitives() {

super();

}

public void startApp() throws MIDletStateChangeException {

display = Display.getDisplay(this);

menu = new List("LAB 3", Choice.IMPLICIT);

menu.append("1. Line", null);

menu.append("2. Rectangle", null);

menu.append("3. Rounded Rectangle", null);

menu.append("4. Circle", null);

menu.append("5. Ellipse", null);

menu.append("6. Arc", null);

menu.append("7. Triangle", null);

menu.addCommand(exitCommand);

menu.setCommandListener(this);

ticker = new Ticker("18124004 : Lab 3 - Graphics Premitives");

menu.setTicker(ticker);

mainMenu();

}

void mainMenu() {

display.setCurrent(menu);

currentlyAt = "main";

}

public void pauseApp() {

display = null;

form = null;

ticker = null;

menu = null;

currentlyAt = null;

}

public void destroyApp(boolean unconditional) {

notifyDestroyed();

}

public void commandAction(Command cm, Displayable ds) {

String label = cm.getLabel();

if (label.equals("Exit")) {

destroyApp(true);

} else if (label.equals("Back")) {

mainMenu();

} else {

List down = (List)display.getCurrent();

switch (down.getSelectedIndex()) {

case 0: drawLine(); break;

case 1: drawRect(); break;

case 2: drawRoundRect(); break;

case 3: drawCirc(); break;

case 4: drawOval(); break;

case 5: drawArc(); break;

case 6: drawTri(); break;

}

}

}

public void drawLine() {

graphicsCanvas c = new graphicsCanvas(0);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "Line";

}

public void drawRect() {

graphicsCanvas c = new graphicsCanvas(1);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "Rectangle";

}

public void drawRoundRect() {

graphicsCanvas c = new graphicsCanvas(2);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "RoundedRect";

}

public void drawCirc() {

graphicsCanvas c = new graphicsCanvas(3);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "Circle";

}

public void drawOval() {

graphicsCanvas c = new graphicsCanvas(4);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "Oval";

}

public void drawArc() {

graphicsCanvas c = new graphicsCanvas(5);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "Arc";

}

public void drawTri() {

graphicsCanvas c = new graphicsCanvas(6);

c.addCommand(backCommand);

c.setCommandListener(this);

display.setCurrent(c);

currentlyAt = "Triangle";

}

}

class graphicsCanvas extends Canvas {

int choice;

public graphicsCanvas (int i) {

super();

choice = i;

}

public void paint(Graphics g) {

g.setColor(0xffffff);

g.fillRect(0, 0, getWidth(), getHeight());

g.setColor(0x0000ff);

if (choice == 0) { //line

g.drawLine(50, 20, 100, 200);

} else if (choice == 1) { //rectangle

g.drawRect(20, 20, 100, 120);

} else if (choice == 2) { //rounded rectangle

g.drawRoundRect(20, 20, 100, 120, 20, 20);

} else if (choice == 3) { //circle

g.drawArc(50, 50, 50, 50, 0, 360);

} else if (choice == 4) { //ellipse

g.drawArc(50, 50, 100, 50, 0, 360);

} else if (choice == 5) { //arc

g.drawArc(50, 50, 100, 100, 30, 200);

} else if (choice == 6) { //triangle

g.fillTriangle(20, 20, 160, 40, 120, 20);

} else {

g.setFont(Font.getFont(Font.FACE\_SYSTEM, Font.STYLE\_BOLD, Font.SIZE\_MEDIUM));

g.drawString("ERROR: UNIDENTIFIED SHAPE CHOICE", 0, 30, g.LEFT | g.TOP);

}

}

}

**Output :**

|  |  |  |
| --- | --- | --- |
|  |  |  |
|  |  |  |
|  |  |  |

1. draws a bar graph to display. Data values can be given at int[]array.

**PhoneNumberValidation.java :**

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

public class PhoneNumberValidation extends MIDlet implements CommandListener {

public Form form;

public TextField textfield;

public Command exitCommand;

public Command okCommand;

public StringItem st;

public String instruction;

public Display display;

public PhoneNumberValidation() {

display = Display.getDisplay(this);

form = new Form("Insert the Phone number");

exitCommand = new Command("Exit", Command.EXIT, 1);

okCommand = new Command("Ok", Command.OK, 1);

st = new StringItem("Phone Number is ", "");

instruction = "Format : <areaCode>XXXXXX\nArea code must be 040|050|041|0400|044\n";

textfield = new TextField("Phone Number:", "", 30, TextField.ANY);

form.append(textfield);

form.append(instruction);

form.addCommand(okCommand);

form.addCommand(exitCommand);

form.setCommandListener(this);

}

public void startApp() { display.setCurrent(form); }

public void pauseApp() { }

public void destroyApp(boolean unconditional) { }

public void commandAction(Command cmd, Displayable displayable) {

if (cmd == exitCommand) {

notifyDestroyed();

} else if (cmd == okCommand) {

String s = textfield.getString();

boolean correct = false;

int len = s.length();

if (len == 9 || len == 10) {

String number = s.substring(len - 6);

String areaCode = s.substring(0, len - 6);

boolean numberIsNumeric = true;

try {

int num = Integer.parseInt(number);

} catch (NumberFormatException e) {

numberIsNumeric = false;

}

if (areaCode.equals("040") || areaCode.equals("041") || areaCode.equals("050") || areaCode.equals("0400") || areaCode.equals("044")) {

if (number.length() == 6 && numberIsNumeric ) {

correct = true;

}

}

}

if (correct) {

st.setText("OK");

} else {

st.setText("wrong\n");

}

form.append(st);

}

}

}

**Output :**

|  |  |
| --- | --- |
|  |  |
|  |  |

1. examines a phone number, that a user entered in a given format. \*Area code should be one of the following: 040, 041, 050, and 0400,044

**BarGraph.java :**

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

public class BarGraph extends MIDlet implements CommandListener {

public Form form;

public Command exitCommand;

public Command OkCommand;

public Command backCommand;

public Displayable d;

public Display display;

public TextField []textfield;

public static int []color = {0x00CED1, 0xff0033, 0x0a75ad, 0xffb6c1, 0xee8899};

public static String []labels = {"DBMS :", "OS :", "CN :", "OOPS :", "JAVA :"};

public BarGraph() {

display = Display.getDisplay(this);

form = new Form("BarGraph : Enter marks (out of 100):");

textfield = new TextField[5];

for (int i = 0; i < 5 ; ++i ) {

textfield[i] = new TextField(labels[i], "", 30, TextField.ANY);

form.append(textfield[i]);

}

OkCommand = new Command("Ok", Command.OK, 1);

exitCommand = new Command("Exit", Command.EXIT, 1);

backCommand = new Command("Back", Command.BACK, 1);

form.addCommand(OkCommand);

form.addCommand(exitCommand);

form.setCommandListener(this);

}

public void startApp() { display.setCurrent(form); }

public void pauseApp() { }

public void destroyApp(boolean unconditional) { }

public void commandAction(Command command, Displayable displayable) {

if (displayable == form) {

if (command == OkCommand) {

int[] data = new int[5];

for (int i = 0; i < 5 ; ++i ) {

data[i] = Integer.parseInt(textfield[i].getString());

}

d = new BarCanvas(data);

d.addCommand(backCommand);

d.setCommandListener(this);

display.setCurrent(d);

} else if (command == exitCommand) {

notifyDestroyed();

}

} else if (displayable == d) {

if (command == backCommand) {

display.setCurrent(form);

}

}

}

class BarCanvas extends Canvas {

int[] data;

int height = 20;

int width = 0;

int ox = 50, oy = 50;

int px = 5, py = 55;

int inc = 25;

int maxWidth = 150;

int maxMarks = 100;

public BarCanvas() {}

public BarCanvas(int[] data) {

this.data = data;

}

public void paint(Graphics g) {

g.setColor(255, 255, 255);

g.fillRect(0, 0, this.getWidth(), this.getHeight());

int i = 0;

while (i < data.length) {

//find horizontal lenght

width = (int)data[i] \* maxWidth / maxMarks;

//draw bar

g.setColor(color[i]);

g.fillRect(ox, oy, width, height);

//print label

g.setColor(0, 0, 0);

g.drawString(labels[i], px, py, g.TOP | g.LEFT);

//next element

oy += inc;

py += inc;

i++;

}

//draw lines

g.setColor(0, 0, 0);

g.drawLine(ox, 30, ox, oy);

g.drawLine(ox, oy, ox + 160, oy);

//plot markings

int cur = 0;

while (cur <= 100) {

int newX = (int)(cur \* maxWidth / maxMarks) + ox;

g.drawLine(newX, oy, newX, oy + 5);

g.drawString("" + cur, newX, oy + 10, g.TOP | g.LEFT);

cur += 25;

}

g.drawString("Marks", ox + 60, oy + 25, g.TOP | g.LEFT); }

}

}

**Output :**

|  |  |
| --- | --- |
|  |  |