**LAB : 1**

**OBJECTIVE :**

Working with J2ME Features

1. Create a program which creates a menu
2. Event Handling
3. Input Checking

**Requrements :**

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

(b) JDK 1.5

(c) Java Wireless Toolkit 2.5.2

**Description :**

J2ME (Java 2 Micro Edition) is an advanced technology in Java, developed with the help of Java Community Process Program. J2ME is a reduced version of the Java API and Java Virtual Machine that is designed to operate within the limited resources available in the embedded computers and microcomputers.

J2ME is targeted to developers of intelligent wireless devices and small computing devices that need to incorporate cross-platform functionality in their products. A key benefit of using J2ME is compatibility with all Java-enabled devices. Motorola, Nokia, Panasonic all have Java enabled devices. A J2ME application is a balance between local and server-side processing.

**Implementation :**

1. Create a program which creates a menu

**CreateMenu.java :**

import javax.microedition.lcdui.\*;

import javax.microedition.midlet.\*;

public class CreateMenu extends MIDlet implements CommandListener {

Display display = null;

List menu = null;

TextBox input = null;

// command

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

static final Command mainMenuCommand = new Command("Main", Command.SCREEN, 1);

static final Command exitCommand = new Command("Exit", Command.STOP, 2);

String currentMenu = null;

public CreateMenu() {}

public void startApp() throws MIDletStateChangeException {

display = Display.getDisplay(this);

menu = new List("Menu Items", Choice.IMPLICIT);

for (int i = 1; i < 7; i++) {

menu.append("Option " + i, null);

}

menu.addCommand(exitCommand);

menu.setCommandListener(this);

mainMenu();

}

public void pauseApp() {

display = null;

menu = null;

input = null;

}

public void destroyApp(boolean unconditional) {

notifyDestroyed();

}

// main menu

void mainMenu() {

display.setCurrent(menu);

currentMenu = "Main";

}

public void prepare() {

input = new TextBox("Enter some text: ", "", 5, TextField.ANY);

input.addCommand(backCommand);

input.setCommandListener(this);

input.setString("");

display.setCurrent(input);

}

public void testItem1() {

prepare();

currentMenu = "item1";

}

public void testItem2() {

prepare();

currentMenu = "item2";

}

public void testItem3() {

prepare();

currentMenu = "item3";

}

public void testItem4() {

prepare();

currentMenu = "item4";

}

//Handle events.

public void commandAction(Command c, Displayable d) {

String label = c.getLabel();

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

destroyApp(true);

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

if (currentMenu.equals("item1") || currentMenu.equals("item2") ||

currentMenu.equals("item3") || currentMenu.equals("item4")) {

// go back to menu

mainMenu();

}

} else {

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

switch (down.getSelectedIndex()) {

case 0: testItem1(); break;

case 1: testItem2(); break;

case 2: testItem3(); break;

case 3: testItem4(); break;

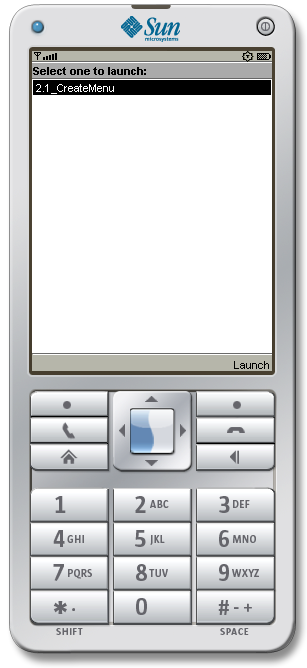
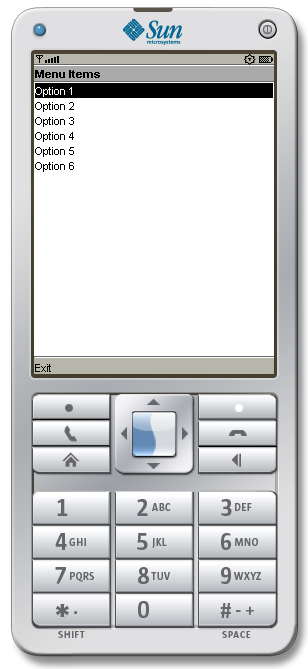
}

}

}

}

**Output :**

1. Event Handling

**EventHandling.java :**

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

public class EventHandling extends MIDlet implements CommandListener, ItemStateListener {

public ChoiceGroup choicegroup;

public ChoiceGroup choicegroup1;

public Form form;

public Form form1;

public Display display;

public Command View;

public Command Exit;

public Command Back;

public StringItem options;

public Item item;

public EventHandling() {

display = Display.getDisplay(this);

form = new Form("");

form1 = new Form("Selcted Options are");

choicegroup = new ChoiceGroup("Preferences", Choice.MULTIPLE);

for (int i = 1; i < 7 ; i++) {

String t = "Option " + i;

choicegroup.append(t, null);

}

choicegroup.setSelectedIndex(0, true);

form.append(choicegroup);

choicegroup1 = new ChoiceGroup("", Choice.EXCLUSIVE);

choicegroup1.append("select all", null);

choicegroup1.append("unselect all", null);

choicegroup1.setSelectedIndex(1, true);

form.append(choicegroup1);

View = new Command("View", Command.OK, 1);

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

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

form.addCommand(View);

form.addCommand(Exit);

form1.addCommand(Back);

form.setCommandListener(this);

form1.setCommandListener(this);

form.setItemStateListener(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 == View) {

boolean opt[] = new boolean[choicegroup.size()];

options = new StringItem("", "");

String values = "";

choicegroup.getSelectedFlags(opt);

options.setText("");

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

if (opt[i]) {

values += choicegroup.getString(i) + "\n";

}

}

options.setText(values);

form1.append(options);

display.setCurrent(form1);

} else if (command == Exit) {

destroyApp(true);

notifyDestroyed();

}

} else if (displayable == form1) {

if (command == Back) {

display.setCurrent(form);

options.setText("");

}

}

}

public void itemStateChanged(Item item) {

if (item == choicegroup1) {

int size = choicegroup.size();

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

if (choicegroup1.getSelectedIndex() == 0) {

choicegroup.setSelectedIndex(i, true);

} else {

choicegroup.setSelectedIndex(i, false);

}

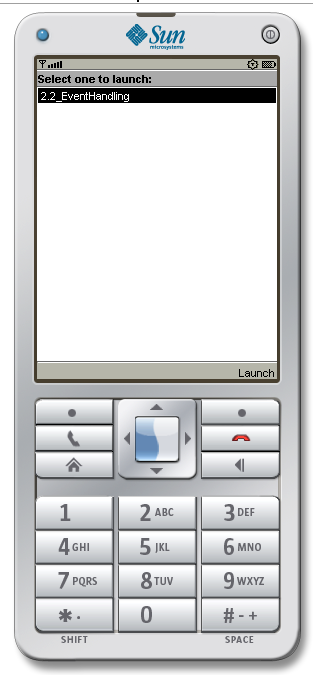
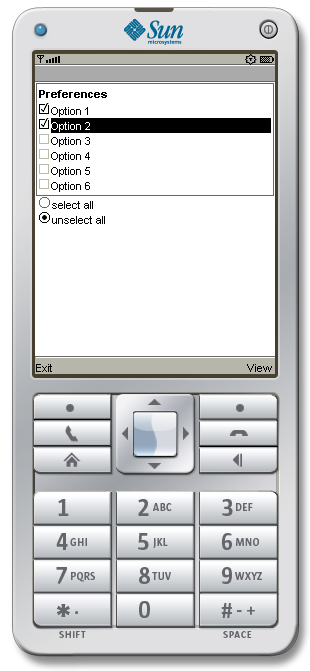
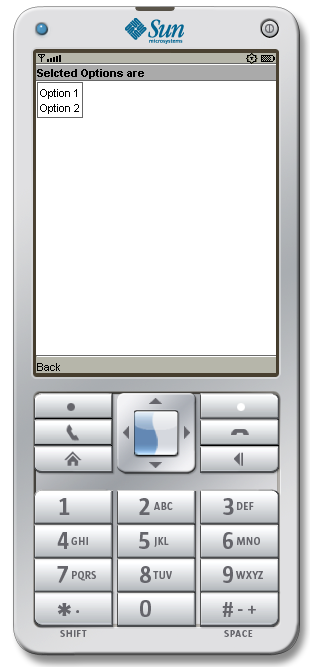
}

}

}

}

**Output :**

1. Input Checking

**InputValidation.java :**

import javax.microedition.midlet.\*;

import javax.microedition.lcdui.\*;

public class InputValidation extends MIDlet implements CommandListener {

private Display display;

private Form registrationFrm;

private String uname = "18124004";

private String pass = "12345";

private Alert messageAlert;

private TextField userName;

private TextField passwordTxt;

private Command okCmd;

private Command exitCmd;

public InputValidation() {

// userName = new TextField("Email:", "", 100, TextField.EMAILADDR);

userName = new TextField("Roll No: ", "", 16, TextField.ANY);

passwordTxt = new TextField("Password:", "", 16, TextField.PASSWORD);

registrationFrm = new Form("User Registration",

new Item[] {userName, passwordTxt});

messageAlert = new Alert("Registration Complete");

messageAlert.setTimeout(3000);

messageAlert.setType(AlertType.CONFIRMATION);

okCmd = new Command("OK", Command.OK, 0);

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

registrationFrm.addCommand(okCmd);

registrationFrm.addCommand(exitCmd);

registrationFrm.setCommandListener(this);

}

protected void startApp() {

display = Display.getDisplay(this);

display.setCurrent(registrationFrm);

}

public void commandAction(Command c, Displayable d) {

if (d == registrationFrm) {

if (c == okCmd) {

String messageContent;

if (userName.getString().length() == 0 || passwordTxt.getString().length() == 0) {

messageContent = "Please Enter All fields";

} else {

boolean c1 = uname.equals(userName.getString());

boolean c2 = pass.equals(passwordTxt.getString());

if (c1 && c2)

messageContent = "Login Successfull";

else

messageContent = "Wrong Credentials";

}

messageAlert.setString(messageContent);

display.setCurrent(messageAlert, registrationFrm);

} else if (c == exitCmd) {

notifyDestroyed();

}

}

}

protected void pauseApp() {

}

protected void destroyApp(boolean unconditional) {

}

}

**Output :**

