

LABORATORIUM SYSTEMÓW MOBILNYCH

STWORZENIE PRZYKŁADOWEJ APLIKACJI MOBILNEJ W J2ME

I. Temat ćwiczenia

Stworzenie przykładowej aplikacji mobilnej w J2ME

II. Wymagania

Wykonanie poprzedniego ćwiczenia

III. Ćwiczenie

1. Stworzenie aplikacji

Celem ćwiczenia jest przygotowanie mobilnej aplikacji w języku Java (J2ME) dla telefonu komórkowego realizującej wizualizację równania kwadratowego o podanych przez użytkownika parametrach.

Utworzenie aplikacji rozpocząć możemy od przygotowania głównej formy programu.

Na początku definiujemy obszar wyświetlacza oraz formularz na którym umieścimy komponenty komunikacji z użytkownikiem (Listing 1).

```
private Display display;  
private Form form;
```

Listing 1 – Definicja obszaru wyświetlacza oraz formularza

Definiujemy komponenty formularza, służące do wprowadzenia parametrów równania kwadratowego – **TextField** (Listing 2).

```
private TextField tf1 = new TextField("A:", "", 6, TextField.ANY);  
private TextField tf2 = new TextField("B:", "", 6, TextField.ANY);  
private TextField tf3 = new TextField("C:", "", 6, TextField.ANY);
```

Listing 2 – Definicja komponentów komunikacji z użytkownikiem

W konstruktorze tworzymy obiekt formularza oraz dodajemy do niego zdefiniowane uprzednio komponenty (Listing 3).

```

display = Display.getDisplay(this);
form = new Form("Rownanie kwadratowe");

form.append(new String("Wprowadz parametry rownania kwadratowego:\n f(x) = Ax^2 + Bx + C"));
form.append(tf1);
form.append(tf2);
form.append(tf3);

```

Listing 3 – Utworzenie obiektów oraz dodanie komponentów do formularza

W następnym kroku definiujemy obiekty odpowiedzialne za nawigację w aplikacji, która odbywa się poprzez posługiwanie się przez użytkownika klawiszami sterującymi telefonu (Listing 4).

```

private Command exitCommand = new Command("Wyjście", Command.SCREEN, 2);
private Command calculateCommand = new Command("Oblicz", Command.SCREEN, 1);

```

Listing 4 – Utworzenie obiektów obsługi klawiszy telefonu

Umieszczamy utworzone obiekty na formularzu (Listing 5).

```

form.addCommand(exitCommand);
form.addCommand(calculateCommand);

```

Listing 5 – Dodanie obiektów obsługi klawiszy telefonu do formularza

W kolejnym etapie definiujemy metodę obsługi naciśnięcia klawiszy (Listing 6).

```

public void commandAction(Command c, Displayable s) {
    if (c == exitCommand) {
        destroyApp(false);
        notifyDestroyed();
    }
    if (c == calculateCommand) {
        float A,B,C;
        try {
            A = Float.parseFloat(tf1.getString().equals("")?"0":tf1.getString());
            B = Float.parseFloat(tf2.getString().equals("")?"0":tf2.getString());
            C = Float.parseFloat(tf3.getString().equals("")?"0":tf3.getString());
        } catch (Exception e) {
            A=B=C=0;
        }
        display.setCurrent(new Wykres(form, display, (int)A, (int)B, (int)C));
    }
}

```

Listing 6 – Przykładowa metoda obsługi naciśnięcia klawiszy

Kompletny, przykładowy kod obsługi głównej formy aplikacji (Listing 7).

```
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.TextField;
import javax.microedition.midlet.MIDlet;

public class MojPierwszyMidlet extends MIDlet implements CommandListener {

    private Command exitCommand = new Command("Wyjście", Command.SCREEN, 2);
    private Command calculateCommand = new Command("Oblicz", Command.SCREEN, 1);

    private Display display;
    private Form form;

    private TextField tf1 = new TextField("A:", "", 6, TextField.ANY);
    private TextField tf2 = new TextField("B:", "", 6, TextField.ANY);
    private TextField tf3 = new TextField("C:", "", 6, TextField.ANY);

    public MojPierwszyMidlet() {
        display = Display.getDisplay(this);
        form = new Form("Równanie kwadratowe");

        form.append(new String("Wprowadz parametry rownania kwadratowego:\n
            f(x) = Ax^2 + Bx + C"));
        form.append(tf1);
        form.append(tf2);
        form.append(tf3);

        form.addCommand(exitCommand);
        form.addCommand(calculateCommand);
        form.setCommandListener(this);
    }

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

    public void pauseApp() {
    }

    public void destroyApp(boolean unconditional) {
    }

    public void commandAction(Command c, Displayable s) {
        if (c == exitCommand) {
            destroyApp(false);
            notifyDestroyed();
        }
        if (c == calculateCommand) {
            float A, B, C;
            try {
                A = Float.parseFloat(tf1.getString().equals("") ? "0" :
                    tf1.getString());
                B = Float.parseFloat(tf2.getString().equals("") ? "0" :
                    tf2.getString());
                C = Float.parseFloat(tf3.getString().equals("") ? "0" :
                    tf3.getString());
            } catch (Exception e) {
                A = B = C = 0;
            }
            display.setCurrent(new Wykres(form, display, (int)A, (int)B, (int)C));
        }
    }
}
```

Listing 7 – Przykładowy kod obsługi głównej formy aplikacji

W kolejnym kroku możemy przystąpić do utworzenia formy wyświetlającej wykres funkcji kwadratowej.

Przykładowy kod wizualizacji wykresu funkcji (Listing 8).

```
import javax.microedition.lcdui.Canvas;
import javax.microedition.lcdui.Command;
import javax.microedition.lcdui.CommandListener;
import javax.microedition.lcdui.Display;
import javax.microedition.lcdui.Displayable;
import javax.microedition.lcdui.Form;
import javax.microedition.lcdui.Graphics;

public class Wykres extends Canvas implements CommandListener {

    Form form;
    Display display;

    private Command backCommand = new Command("Wstecz", Command.SCREEN, 2);

    // szerokość i długość ekranu
    int screenWidth, screenHeight;

    int A, B, C;

    public Wykres(Form f, Display d, int a, int b, int c) {
        display = d;
        form = f;

        A = a;
        B = b;
        C = c;

        screenWidth = getWidth();
        screenHeight = getHeight();

        addCommand(backCommand);
        setCommandListener(this);
    }

    protected void paint(Graphics g) {
        // zmiana aktualnego koloru
        g.setColor(0xffffffff);
        // namalowanie wypełnionego aktualnym kolorem prostokąta
        g.fillRect(0, 0, screenWidth, screenHeight);

        // zmiana aktualnego koloru
        g.setColor(0x0);
        g.drawLine(0, screenHeight / 2, screenWidth, screenHeight / 2);
        g.drawLine(screenWidth / 2, 0, screenWidth / 2, screenHeight);

        g.setColor(0xff0000);
        int prevY = calcY(0 - screenWidth / 2);
        int newY;
        for (int k = 1; k < screenWidth; k++) {
            newY = calcY(0 - screenWidth / 2 + k);
            g.drawLine(k - 1, prevY + screenHeight/2, k, newY + screenHeight/2);
            prevY = newY;
        }
    }

    private int calcY(int x) {
        return -(A * x * x + B * x + C);
    }

    public void commandAction(Command c, Displayable s) {
        if (c == backCommand) {
            display.setCurrent(form);
        }
    }
}
```

Listing 8 – Przykładowy kod wizualizacji wykresu funkcji kwadratowej

3. Zadanie

Napisać aplikację rysującą wybrane przez użytkownika figury geometryczne. Aplikacja ma umożliwiać użytkownikowi:

- wybór figury geometrycznej do narysowania (kwadrat, trójkąt, koło, romb, trapez),
- wybór liczby figur do narysowania,
- zdefiniowanie koloru tła, obramowania i wypełnienia figur.

Aplikację należy wzbogacić o mechanizm powiększania oraz zmniejszania wyświetlanej figury posługując się klawiszami funkcyjnymi telefonu.