Facultatea Calculatoare, Informatica si Microelectronica

Universitatea Tehnica a Moldovei

Medii Interactive de Dezvoltare a Produselor Soft

Lucrarea de laborator#2

GUI Development

Autor:
Eugen MIROVSCHI

lector asistent:
Irina Cojanu
lector superior:
Svetlana Cojocaru

Laboratory work #2

1 Scopul lucrarii de laborator

Studierea unui IDE, framework ce permite crearea unei aplicații cu interfață grafică și elaborarea unui calculator utilizând aceste tehnologii.

2 Obiective

a) Realizeaza un simplu GUI Calculator

b) Operatiile simple: +,-,*,/,putere,radical,InversareSemn(+/-),operatii cu numere zecimale.

c) Divizare proiectului in doua module - Interfata grafica(Modul GUI) si Modulul de baza(Core Module).

3 Laboratory work implementation

3.1 Tasks and Points

a) Realizeaza un simplu GUI calculator care suporta urmatoare functii: +, -, /, *, putere, radical, InversareSemn(+/-), operatii cu numere zecimale.

b) Divizare proiectului in doua module - Interfata grafica(Modul GUI) si Modulul de baza(Core Module).

3.2 Analiza lucrarii de laborator

a) Primul pas a fost inițializarea unui nou repozitoriu pe GitHub și clonarea acestuia pe calculatorul personal: `https://github.com/emirovschi/MIDPS\-2`.

b) Proiectul a fost creat utilizând IntelliJ[1] și folosind Maven[2] ca build manager. Acesta oferă posibilitatea de a adăuga automat toate dependențele inclusiv cele tranzitive.

c) Acest proiect este compus din 2 module:

core Conține toată logica de calcul și conversie a numerelor din șir de caractere în numere cu virgula mobilă;

gui Este dependent de GUI framework și conține descrierea interfeței grafice precum și setul de acțiuni care vor fi executate pentru diferite tipuri de evenimente gen apăsare de buton virtual sau fizic.

Maven ofera posibilitatea de a defini aceste module care la rândul său vor deveni proiecte de acest tip și vor avea fișierul propriu cu configurare.

```
<groupId>com.emirovschi</groupId>
<artifactId>midps2</artifactId>
<packaging>pom</packaging>
<version>1.0</version>
<modules>
    <module>gui</module>
    <module>core</module>
</modules>
```

Figure 3.1 – Crearea modulelor in Maven

d) Interfața grafică a fost creată utilizând framework-ul gratuit Apache Pivot[3]. Componentele vizuale sunt definite utilizând un fișier XML iar legătura bidirecțională se face automat folosind anotarea **@BXML**. Acest framework permite extinderea componentelor existente pentru a adăuga careva funcționalități noi sau comportament specific.

```
<TablePane.Row height="1*">
    <Label bxml:id="currentOperation" TablePane.columnSpan="5"
           styles="{horizontalAlignment:'right',  verticalAlignment:'center', wrapText:true, font:{size:15}}}"/>
</TablePane.Row>
<TablePane.Row height="1*">
    <Separator TablePane.columnSpan="5"/>
</TablePane.Row>
<TablePane.Row height="2*">
    <Label bxml:id="currentValue" TablePane.columnSpan="5" text="0"
           styles="{horizontalAlignment:'right',  verticalAlignment:'center', wrapText:true, font:{size:30, bold:true}}"/>
</TablePane.Row>
```

Figure 3.2 – Definirea interfeței grafice folosind fișierul de configurare XML

```
@BXML
private Label currentOperation;

@BXML
private Label currentValue;
```

Figure 3.3 – Crearea legăturii între componente și variabile utilizând anotația **@BXML**

e) Asocierea evenimentelor pentru butoane se face utilizând tipul componentelor. Respectiv fiecare buton va avea propriul său comportament.

```
Matcher.of(button)
        .when(InputButton.class).then(addListener(b -> graphicCalculator.push(b.getDigit())))
        .when(DecimalButton.class).then(addListener(b -> graphicCalculator.startDecimal()))
        .when(ClearButton.class).then(addListener(b -> graphicCalculator.clear()))
        .when(OperatorButton.class).then(addListener(b -> graphicCalculator.push(b.getOperator())))
        .when(EqualsButton.class).then(addListener(b -> graphicCalculator.calculate()))
        .when(ChangeSignButton.class).then(addListener(b -> graphicCalculator.changeSign()))
        .match();
```

Figure 3.4 – Adăugarea evenimentelor

3.3 Imagini

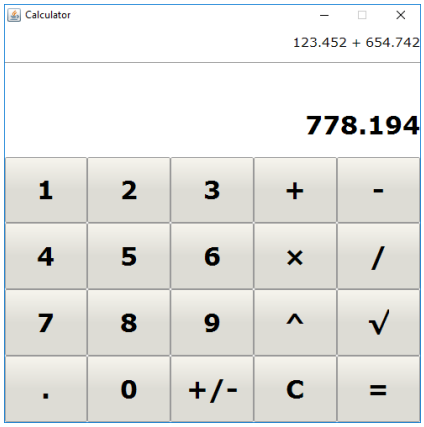

Figure 3.5 – Exemplu de calcul

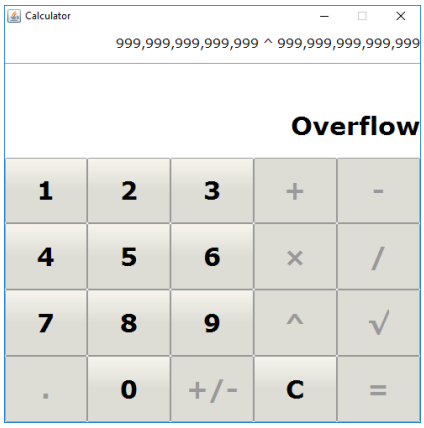

Figure 3.6 – Exemplu de exces

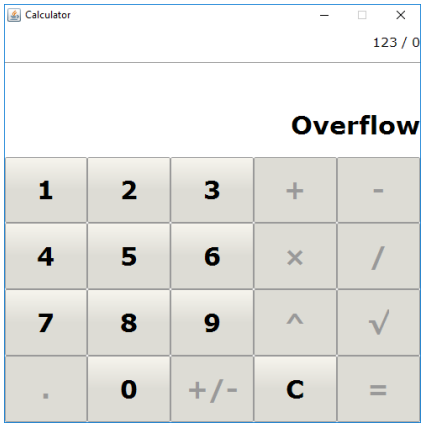

Figure 3.7 – Divizarea la zero

Concluzie

Pentru efectuarea acestei lucrări am utilizat un set de instrumente care m-au ajutat la implementarea unei aplicații grafice. Am folosit git ca sistem de control al versiunilor. Acesta oferă posibilitatea de a înregistra fiecare modificare ca o versiune separată precum și crearea mai multor branch-uri. Ca mediu de dezvoltare am folosit IntelliJ care permite redactarea optimă a codului Java și are integrarea cu git și Maven. Pentru a controla structura și modul de construire a proiectului, precum și includerea eficientă și rapidă a dependențelor externe am folosit managerul Maven. Una din dependențele principale utilizate este biblioteca gratuita create de Apache foundation numită Pivot care oferă posibilitatea de a construi interfețe grafice. Ea permite definirea înfățișării folosind un fișier XML apoi manipularea acestor componente și atașarea evenimentelor folosind anotarea @BXML. Toate aceste instrumente mi-au permis sa dezvolt eficient un produs software calitativ.

References

1 JetBrains IntelliJ, *offical page*, `https://www.jetbrains.com/idea/`

2 Apache Maven, *offical page*, `https://maven.apache.org/`

3 Apache Pivot, *documentation*, `http://pivot.apache.org/tutorials/`