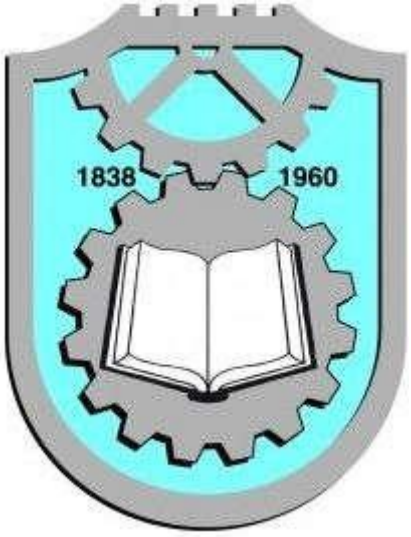
Универзитет у Крагујевцу

Факултет инжењерских наука



ПРОЈЕКТНИ ЗАДАТАК

Предмет:

Софтверски инжењеринг

Тема:

Препознавање контура на слици

Професор:

Ненад Филиповић

Тијана Шуштершич

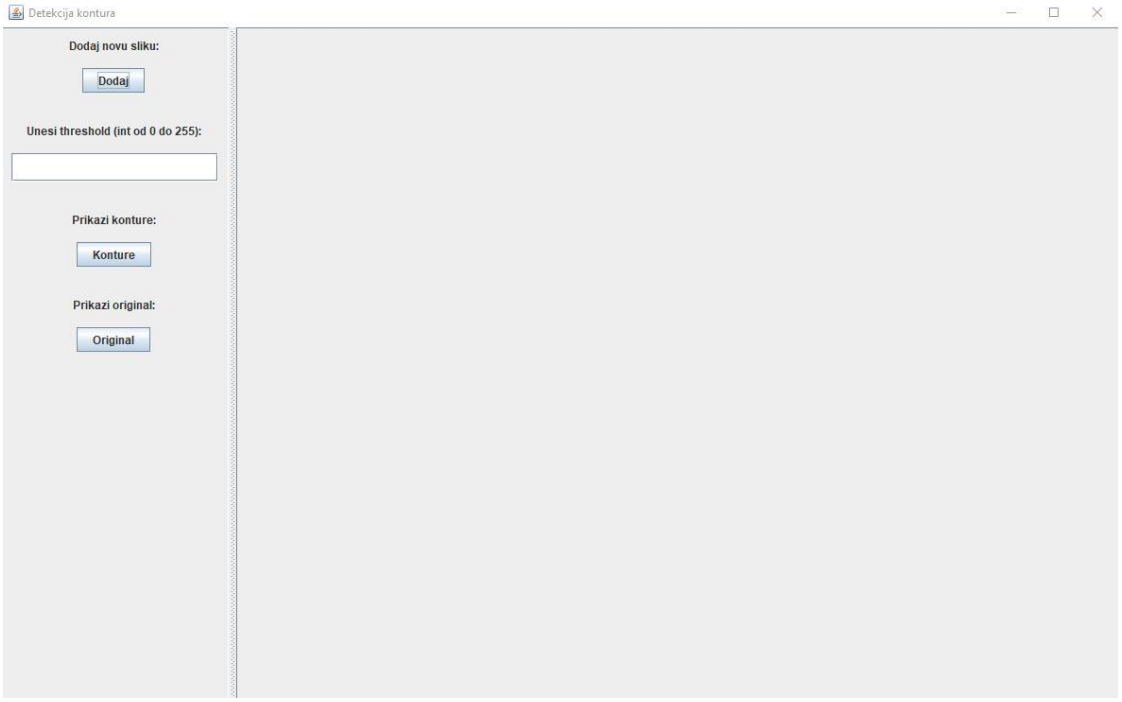
Студент:

Никола Џајевић 615/2017

Садржај

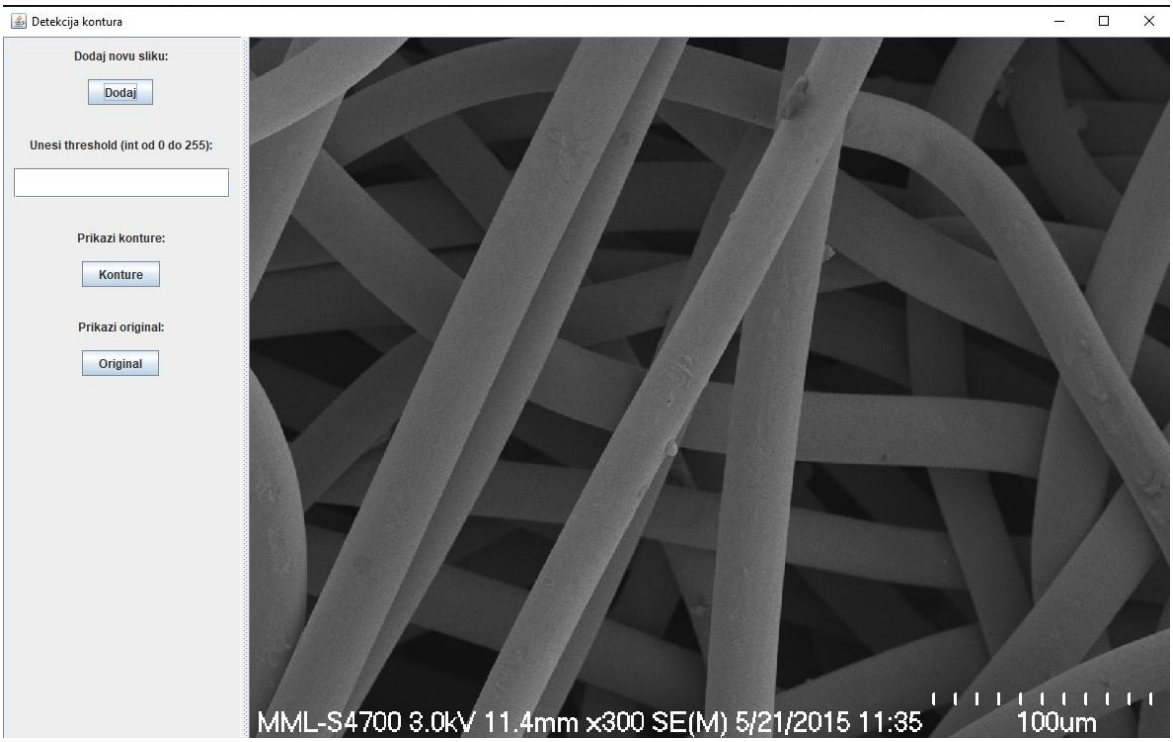
Опис функционалности пројекта

Пројекат представља апликацију која учитава слику (у BIT map формату) и на основу унетог treshold-а препознаје контуре на слици. Апликација омогућава да корисник учита фотографију, затим унесе treshold (вредност од 0 до 255) и као решење добије слику на којој су издвојене контуре.
Приликом покретања апликације отвара се главни прозор (Слика1), притиском на дугме ''Dodaj'' кориснику се отвара нов прозор у којем корисник проналази и додаје жељену слику.



Слика1

Након учитавања слике, програм проверава да ли је слика црно-бела, уколико слика задовољава критеријум појављује се у апликацији (Слика2), уколико не задовољава програм избацује грешку и обавештава корисника да унесе валидну слику.



Слика2

За унету слику, корисник у пољу ''Unesi treshold'' уноси жељени treshold и уколико унета вредност задовољава критеријум да је типа integer (0-255), притиском на дугме ''Konture'' добија слику на којој су издвојене контуре (Слика3). Да би упоредио почетну слику и слику на којој су издвојене контуре, корисник притиском на дугме ''Original'', враћа унету фотографију (Слика1).

Слика3

Package main

Апликација поседује три пакета. То су пакети*: main, view* и *controller.*

Пакет *main* има класу *Main* и задужена је за покретање самог програма (главног прозора) .

Слика4 (*Изглед пакета main package)*

Package view

Пакет view представља све класе и радње које формирају графички интерфејс (*GUI*) који се приказује кориснику при покретању програма. Састоји се из три класе са својим методама.
То су класе:

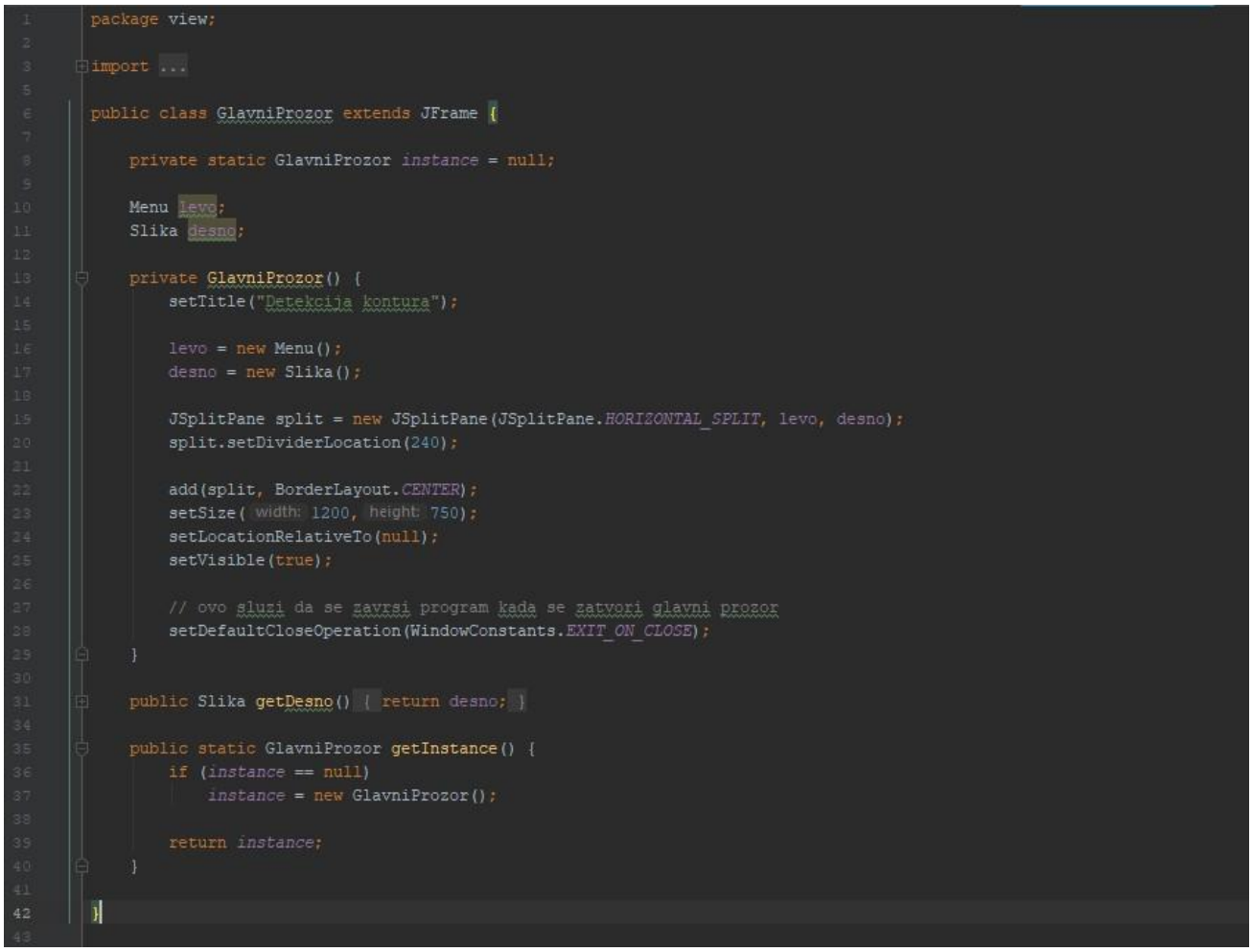
*1.**GlavniProzor***

*2.**Menu***

*3.**Slika***

У пакету *view* су позване Java-ине компоненте javax.swing, java.awt, javax.imageio.ImageIO, Java.awt.image.BufferedImage, java.io.File које код праве много флексибилнијим, практичнијим и јаснијим за тумачење.

Класа *GlavniProzor (Слика* 5) одређује изглед прозора покренутог програма, у њој су дефинисане методе које деле прозор на две стране, на левој је кориснички део где корисник има могућност да унесе слику и treshold, док на десној страни има увид у слику (унету или са издвојеним контурама). У класи је дефинисана приватна метода *GlavniProzor* јер се односи само на главни прозор програма и типа је *singleton.*
Такође је у класи дефинисана метода која при затварању прозора завршава радњу програма, како он не би радио у позадини



```java
package view;

import ...

public class GlavniProzor extends JFrame {

    private static GlavniProzor instance = null;

    Menu levo;
    Slika desno;

    private GlavniProzor() {
        setTitle("Detekcija kontura");

        levo = new Menu();
        desno = new Slika();

        JSplitPane split = new JSplitPane(JSplitPane.HORIZONTAL_SPLIT, levo, desno);
        split.setDividerLocation(240);

        add(split, BorderLayout.CENTER);
        setSize( width: 1200, height: 750);
        setLocationRelativeTo(null);
        setVisible(true);

        // ovo sluzi da se zavrsi program kada se zatvori glavni prozor
        setDefaultCloseOperation(WindowConstants.EXIT_ON_CLOSE);
    }

    public Slika getDesno() { return desno; }

    public static GlavniProzor getInstance() {
        if (instance == null)
            instance = new GlavniProzor();

        return instance;
    }

}
```

Слика5(изглед класе GlavniProzor)

Класа *Menu* (Слика6) дефинише изглед корисничког дела, где корисник уноси вредности. Састоји се из три дугмета и једног дела за унос текста, у овом случају tresholda. Дугме ’’Dodaj’’ кориснику омогућава да изабере слику којом жели да манипулише, у пољу за унос tresholda корисник уноси целобројну вредност од 0 до 255, затим притиском на дугме ''Konture’’ добија слику са издвојеним контурама и дугме ''Original’’ које враћа на екран унету слику уколико корисник жели да види разлику између унете слике и слике на којој су контуре издвојене.

```java
package view;

import ...

public class Menu extends JPanel {

    public Menu() {

        BoxLayout bl = new BoxLayout( target: this, BoxLayout.Y_AXIS);
        setLayout(bl);
        setBorder(BorderFactory.createEmptyBorder( top: 10,  left: 10,  bottom: 10,  right: 10));

        JLabel l1 = new JLabel( text: "Dodaj novu sliku:");
        l1.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        add(l1);
        add(new JLabel( text: " "));

        JButton dugmeDodaj = new JButton( text: "Dodaj");
        dugmeDodaj.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        dugmeDodaj.addActionListener(new DodajSliku());
        add(dugmeDodaj);

        add(new JLabel( text: " "));
        add(new JLabel( text: " "));

        JLabel l2 = new JLabel( text: "Unesi threshold (int od 0 do 255):");
        l2.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        add(l2);
        add(new JLabel( text: " "));
        JTextField textThreshold = new JTextField();
        textThreshold.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        textThreshold.setMaximumSize(new Dimension( width: 240, height: 30));
        add(textThreshold);

        add(new JLabel( text: " "));
        add(new JLabel( text: " "));

        JLabel l3 = new JLabel( text: "Prikazi konture:");
        l3.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        add(l3);
        add(new JLabel( text: " "));

        JButton dugmeKonture = new JButton( text: "Konture");
        dugmeKonture.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        dugmeKonture.addActionListener(new PrikaziKonture(textThreshold));
        add(dugmeKonture);
        add(new JLabel( text: " "));
        add(new JLabel( text: " "));

        JLabel l4 = new JLabel( text: "Prikazi original:");
        l4.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        add(l4);
        add(new JLabel( text: " "));

        JButton dugmeOriginal = new JButton( text: "Original");
        dugmeOriginal.setAlignmentX(JLabel.CENTER_ALIGNMENT);
        dugmeOriginal.addActionListener(new PrikaziOriginal());
        add(dugmeOriginal);
    }

}
```

Слика6(изглед класе Menu)

Класа *Slika (Слика7)* дефинише унос слике и њено приказивање на екрану. При уносу слике, програм проверава да ли је изабрана слика црно-бела, уколико је услов испуњен изабрана слика јесте црно-бела и приказује се на екрану, а уколико се услов не испуни, програм избацује грешку и корисник мора изабрати другу слику.

Провера да ли је слика црно-бела се врши за сваки пиксел, тако што се вредност првог осмобитног низа, који представља црвену боју (R), SHIFT-ује за 16 битова у десно и затим се на тај SHIFT-ован

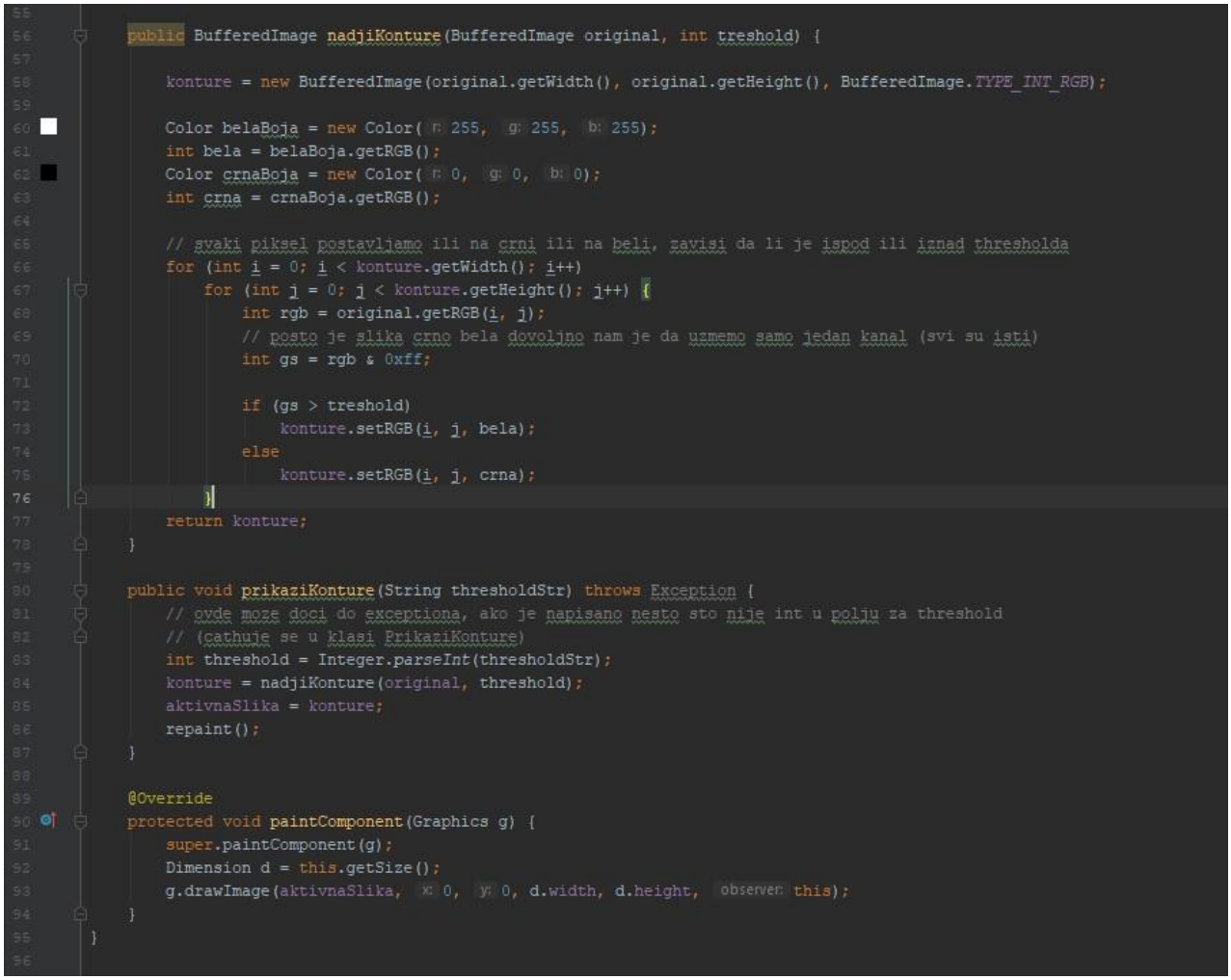
низ AND-ује хексадецимална вредност 0xff која је у бинарном систему 11111111 и та вредност се чува у промењивој **r**, друга вредност је вредност зелене боје која се SHIFT-ује за осам битова у десно и AND-ује јој се хексадецимална вредност која се чува у промењивој **g** и на крају вредност плаве боје се не SHIFT-ује већ јој се само AND-ује хексадецимална вредност и таква се чува у промењивој **b**. Провером да ли су вредности једнаке долазимо до решења да ли је слика црно-бела или није.

Када слика задовољи критеријум програм је прикаже на екрану, након уноса threshold-а од стране корисника и притиском на дугме ''Konture'', програм користи методу *nadjiKonture* у којој су дефинисане боје црна и бела и која за унету слику проверава вредност пиксела и формира нову слику на основу принципа да ако је вредност пиксела изнад treshold-а тај пиксел боји у бело, а ако је вредност пиксела мања од вредности treshold-а пиксел боји у црно. Принцип рада је да метода узима један осмобитни канал (јер је слика црно-бела, сви канали су исти) и проверава његову вредност. На основу проверене вредности одређује бојење.

Дефинисана је и метода *repaint* која служи да освежи приказ, тако што приказује нову слику уколико је дошло до неке промене од стране корисника.

Уколико унета вредност treshold-а није целобројна, под претпоставком да је корисник нешто грешком написао, програм избацује грешку уноса која корисника обавештава да је направио грешку приликом уноса вредности.У прилогу испод текста је приказана Слика7.

```
package view;

import ...

public class Slika extends JPanel {

    BufferedImage aktivnaSlika, original, konture;

    // proverava da li je uneta slika crno bela
    public boolean proveriSliku(BufferedImage slika) {

        int w = slika.getWidth();
        int h = slika.getHeight();

        int p, r, g, b;

        for (int i = 0; i < w; i++)
            for (int j = 0; j < h; j++) {
                // vrednost jednog pixela
                p = slika.getRGB(i, j);
                // prvih 8 bitova je crvena
                r = (p >> 16) & 0xff;
                // sledecih 8 bitova je zelena
                g = (p >> 8) & 0xff;
                // poslednjih 8 bitova je plava
                b = (p) & 0xff;

                // provera da li je R=G=B, ako nije onda nije crno bela slika
                if (r != g || g != b )
                    return false;
            }

        return true;
    }

    public void dodajSliku(File novaSlika) throws Exception {

        if (!proveriSliku(ImageIO.read(novaSlika)))
            throw new Exception("Slika nije validna");

        original = ImageIO.read(novaSlika);
        aktivnaSlika = original;
        // repaint sluzi da se osvezi prikaz, tj da se prikaze nova slika ako se nesto menjalo
        repaint();
    }
```



```
    public BufferedImage nadjiKonture(BufferedImage original, int treshold) {

        konture = new BufferedImage(original.getWidth(), original.getHeight(), BufferedImage.TYPE_INT_RGB);

        Color belaBoja = new Color( r: 255,  g: 255,  b: 255);
        int bela = belaBoja.getRGB();
        Color crnaBoja = new Color( r: 0,  g: 0,  b: 0);
        int crna = crnaBoja.getRGB();

        // svaki piksel postavljamo ili na crni ili na beli, zavisi da li je ispod ili iznad thresholda
        for (int i = 0; i < konture.getWidth(); i++)
            for (int j = 0; j < konture.getHeight(); j++) {
                int rgb = original.getRGB(i, j);
                // posto je slika crno bela dovoljno nam je da uzmemo samo jedan kanal (svi su isti)
                int gs = rgb & 0xff;

                if (gs > treshold)
                    konture.setRGB(i, j, bela);
                else
                    konture.setRGB(i, j, crna);
            }
        return konture;
    }

    public void prikaziKonture(String thresholdStr) throws Exception {
        // ovde moze doci do exceptiona, ako je napisano nesto sto nije int u polju za threshold
        // (cathuje se u klasi PrikaziKonture)
        int threshold = Integer.parseInt(thresholdStr);
        konture = nadjiKonture(original, threshold);
        aktivnaSlika = konture;
        repaint();
    }

    @Override
    protected void paintComponent(Graphics g) {
        super.paintComponent(g);
        Dimension d = this.getSize();
        g.drawImage(aktivnaSlika,  x: 0,  y: 0, d.width, d.height,  observer: this);
    }
}
```

Слика8(изглед класе Slika)

Package controller

Пакет controller представља све класе и радње које повезују радње програма и кориснички унетих података, представља везу између корисника и програма. Састоји се из три класе са својим методама. То су класе:

*1.**DodajSliku***

*2.**PrikaziKonture***

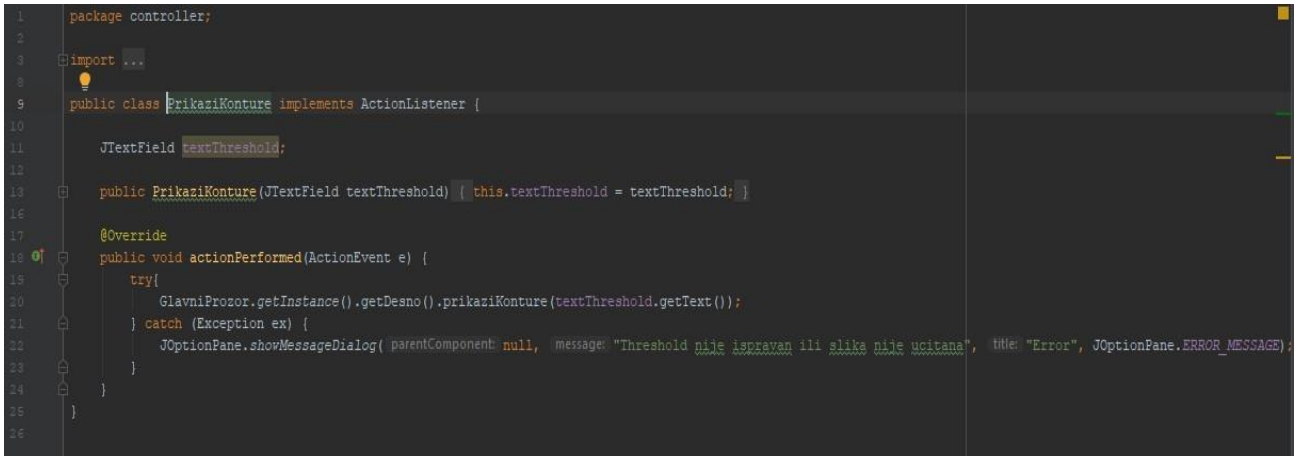
*3.**PrikaziOriginal***

У пакету *controller* су позване Java-ине компоненте javax.swing, java.awt.event.ActionListener, java.awt.event.ActionEvent, java.io.File које олакшавају програму решавање задатих проблема и поједностављују сам код.

Класа *DodajSliku* дефинише корисниково додавање слике, тако што кликом на дугме ''Dodaj'' позива искачући прозор у којем корисник треба да пронађе слику коју жели да унесе. Када корисник изабере слику, кликом на дугме ''Open'', програм проверава да ли је слика црно-бела и приказује је на екрану. У класи је коришћен Java-ин *FileChooser*(метода која омогућава кориснику да унесе слику).

Слика9(приказ класе DodajSliku)

Класа *PrikaziKonture* дефинише приказ слике на којој су издвојене контуре. Активира се кликом на дугме ’’Konture’’ и кориснику приказује на екрану слику. Метода типа *void* под називом *actionPerformed* се приликом клика на дугме активира и проверава да ли је унет treshold, уколико јесте, програм ће се извршити, уколико није, програм ће избацити грешку да унети treshold или слика нису исправни.



Слика10(приказ класе PrikaziKonture)

Класа *PrikaziOriginal* дефинише приказ унете слике кориснику. Уколико корисник жели да види како је изгледала унета слика и да упореди са сликом на којој су издвојене контуре, притиском на дугме ’’Original’’ програм кориснику приказује унету слику. Кликом на дугме се позива метода типа *void* под називом *actionPerformed* која враћа унету слику на екран.

Слика11(приказ класе PrikaziOriginal)

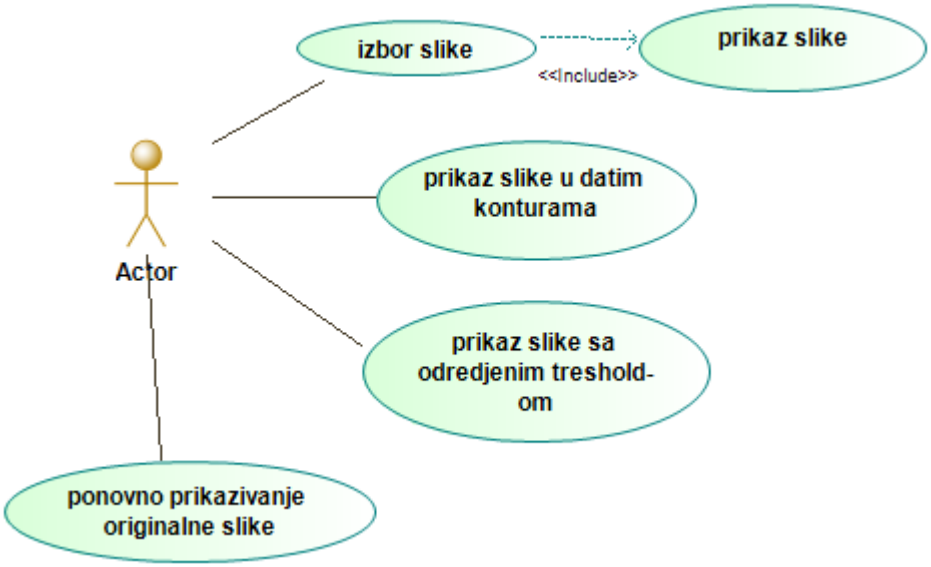
UML дијаграми

Undefined **M**odeling **L**anguage је стандардни графички језик за моделовање објектно-орјентисаног софтвера. UML је општи језик за моделовање помоћу ког се преко графичких симбола прави апстрактни модел система познат као UML модел.

USE Case Diagram

- Дијаграм случајева-коришћења приказује скуп случајева коришћења и актера
- Омогућавају крајњим корисницима да разумеју систем
- Поглед корисника на функционисање система

Дијаграм случајева пројекта, приказан је следећом сликом:

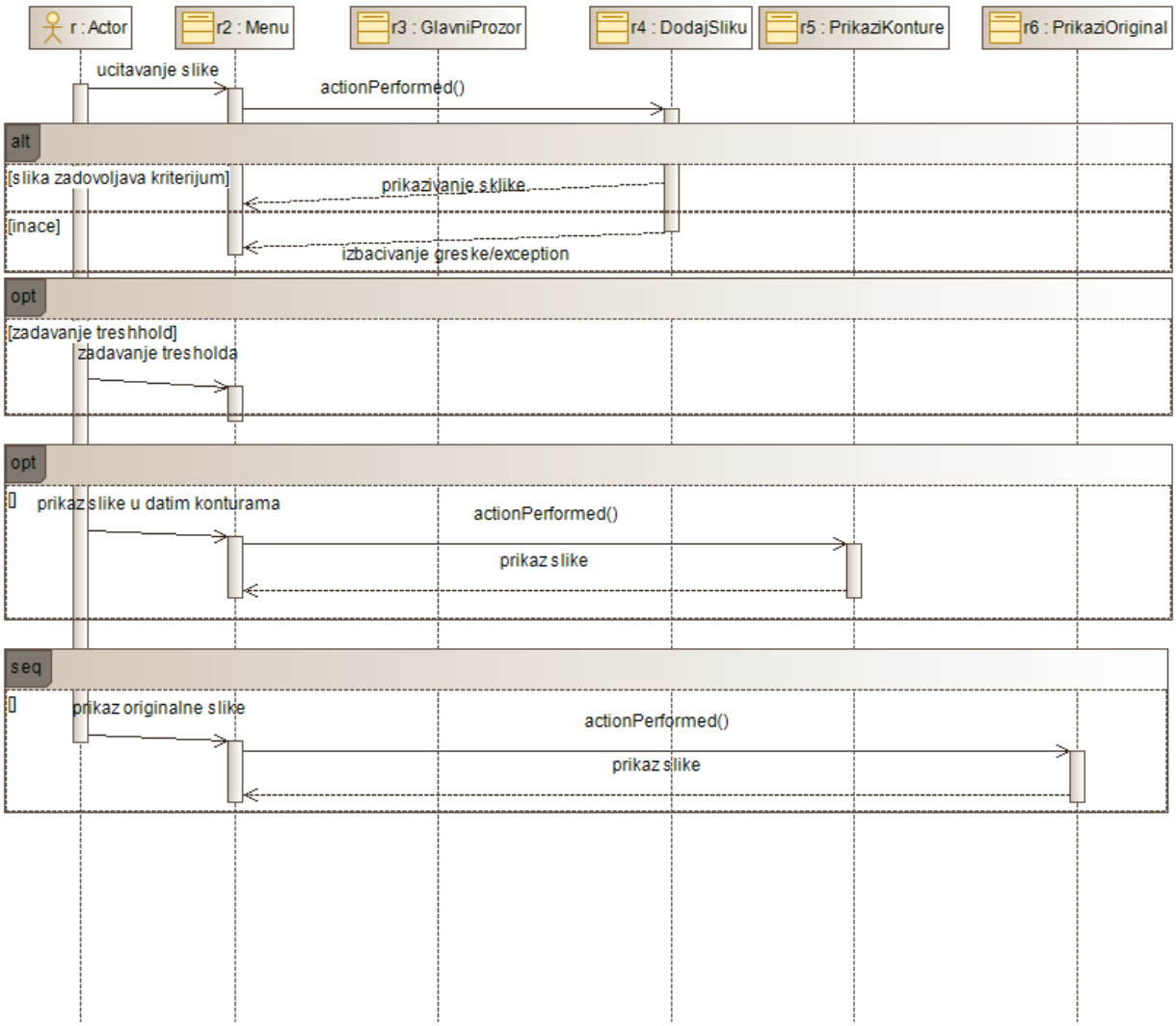


Слика12(Use Case Diagram)

Дијаграм секвенци

- Користе се да специфицирају интеракцију
- Моделирају конкретне сценарије
- Описујући комуникацијске секвенце на различитим нивоима детаља

Дијаграм секвенци пројекта, приказан је следећом сликом:



Слика13(Дијаграм секвенци)

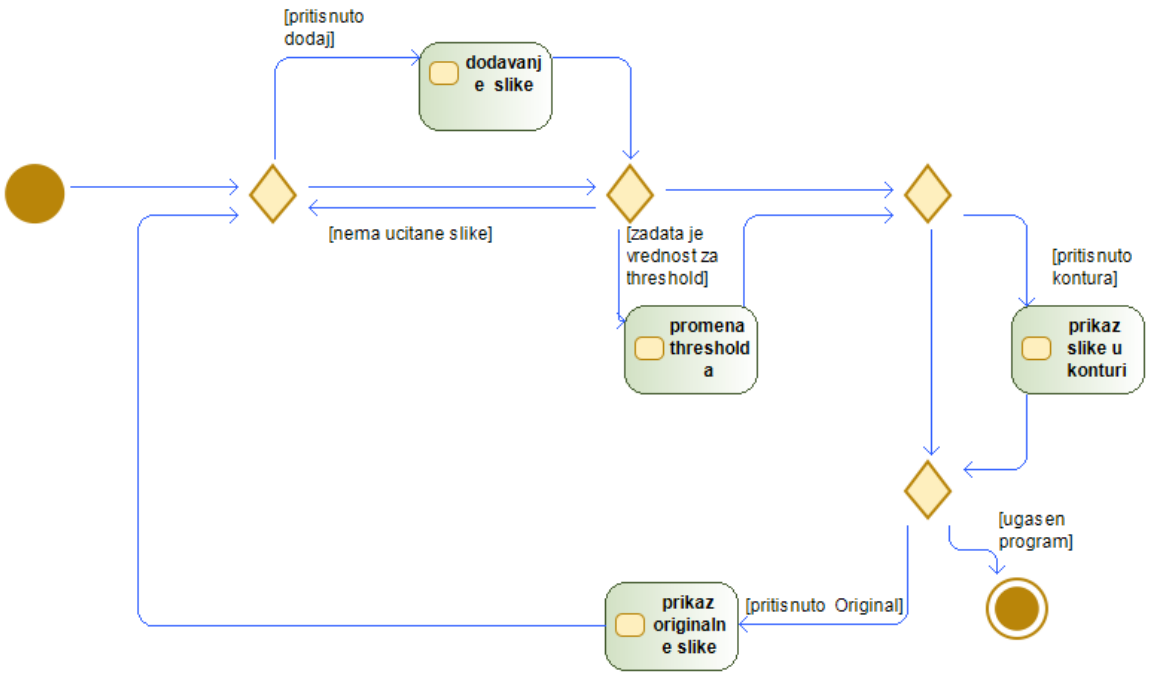
Дијаграм активности

Дијаграми активности су намењени моделирању динамичких аспеката(понашања) система.

Дијаграм активности приказује:

- Ток активности коју извршавају објекти
- Ток објеката између корака активности

Дијаграм активности пројекта, приказан је следећом сликом:



Слика14(Дијаграм активности)

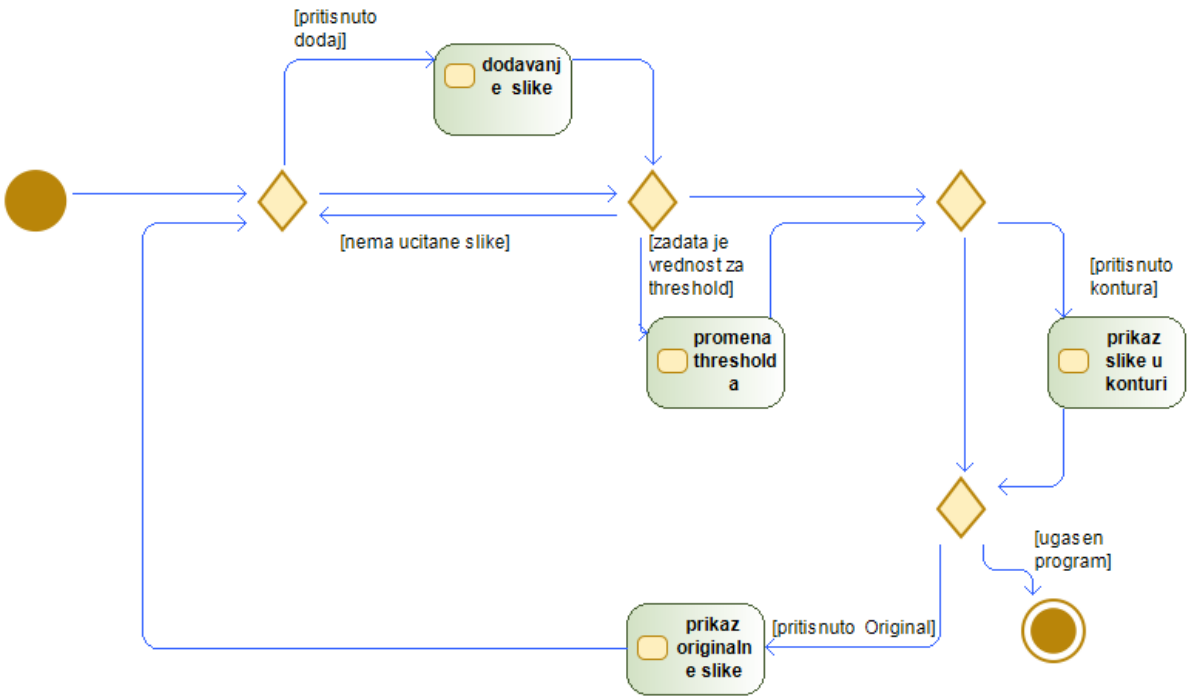
Дијаграм стања

Машина стања(state machine)

Дијаграм стања је граф који приказује аутомат стања:

- Чворови су стања
- Гране су прелази

Дијаграм стања пројекта, приказан је следећом сликом:



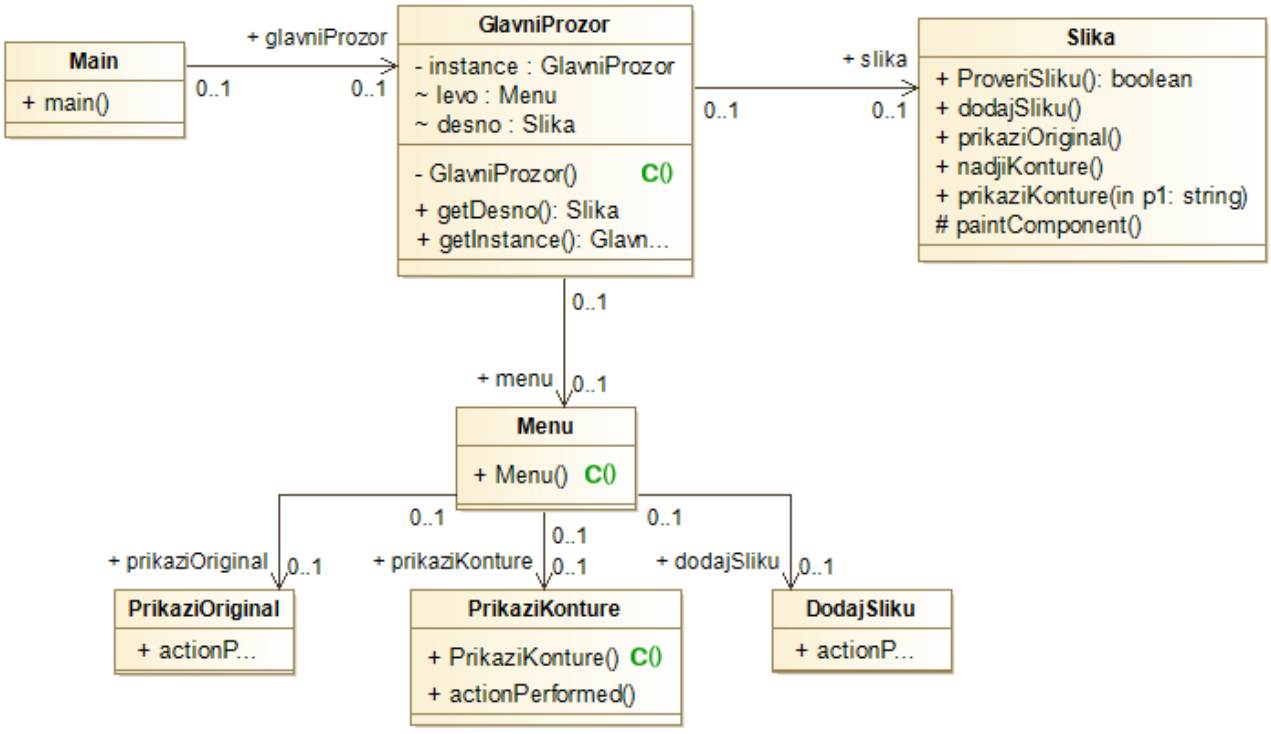
Слика17 (Дијаграм стања)

Дијаграм класа

Дијаграм класа приказује скуп класа, интерфејса, сарадњи и других ствари структуре, повезаним релацијама.

- Специфицира логичке и статичке аспекте модела
- Дијаграми класа су најчешћи у објектном моделирању
- Већина алата подржава генерисање скелета кода из дијаграма класа

Дијаграм класа пројекта, приказан је следећом сликом:



Слика18 (Дијаграм класа)

Литература

1. *Софтверски инжењеринг*. (2018). Преузето 20. Јула 2020 са сајта http://moodle.fink.rs/course/view.php?id=978
2. *UML* (2018). Преузето 5. Септембра 2020 са сајта https://en.wikipedia.org/wiki/Unified_Modeling_Language
3. *Java AWT Tutorial* . Преузето 2. Септембра 2020 са сајта https://www.javatpoint.com/java-awt
4. *Java SWING Tutorial.* Преузето 2.Септембра 2020 са сајта https://www.javatpoint.com/java-swing
5. *Treshold.java* Преузето 8. Септембра 2020 са сајта https://introcs.cs.princeton.edu/java/31datatype/Threshold.java.html
6. *Basic Tresholding Operations.* Преузето 8. Септембра 2020 са сајта https://docs.opencv.org/3.4/db/d8e/tutorial_threshold.html