UiO : Institutt for informatikk
Det matematisk-naturvitenskapelige fakultet

En kort introduksjon til JavaFX

Til bruk i IN1010

Dag Langmyhr

Våren 2019 (versjon 8. april 2019)

En kort introduksjon til JavaFX

Dag Langmyhr

Våren 2019 (versjon 8. april 2019)

1 Introduksjon

For å kunne lage programmer med grafiske brukergrensesnitt (GUI = Graphical User Interface), trenger vi et rammeverk. Det finnes flere, men i IN1010 skal vi bruke JavaFX.

1.1 Et minimalt eksempel

Svært mange introduksjoner til programmeringsspråk starter med et lite program som bare skriver «Hallo». Vi skal gjøre det samme, og denne koden resulterer i et vindu på dataskjermen som ser ut som vist i figur 1 på neste side.

```
1  import javafx.application.Application;
2  import javafx.stage.Stage;
3  import javafx.scene.Scene;
4  import javafx.scene.layout.Pane;
5  import javafx.scene.text.Font;
6  import javafx.scene.text.Text;
7
8  public class Hallo extends Application {
9      @Override
10     public void start(Stage teater) {
11         Text hilsen = new Text("Hallo, alle sammen!");
12         hilsen.setY(40);
13         hilsen.setFont(new Font(40));
14
15         Pane kulisser = new Pane();
16         kulisser.getChildren().add(hilsen);
17
18         Scene scene = new Scene(kulisser);
19
20         teater.setTitle("Velkommen til Java FX");
21         teater.setScene(scene);
22         teater.show();
23     }
24
25     public static void main(String[] args) {
26         launch(args);
27     }
28 }
```

Figur 1: Et minimalt eksempel på FX

1.2 Teatermetaforen

Når man konstruerer et GUI-program med JavaFX, er det nyttig å tenke seg at man er på et teater når teppet går opp:

- Vi har selve teateret med teaterscenen (klassen Stage).
- På teaterscenen kan vi ha ulike scenebilder (instanser av klassen Scene).
- Selv scenebildet består av diverse kulisser; vi kan tenke oss disse montert på en usynlig flate på scenen (klassen Pane).
- De enkelte kulissene lager vi etter behov (klassene Text, Rectangle, Circle etc).

Dette gjenspeiles i koden vist på side 1:

Stage (selve teateret): linje 10 og 20–22.

Scene (scenebildet): linje 18 og 21.

Pane (monteringsflaten for kulissene): linje 15–16 og 18.

Text (kulisse som viser en tekst): linje 11–13 og 16.

1.3 Posisjoneringen

Når vi har flere kulisser i et scenebilde, må vi posisjonere dem slik at de ikke havner oppå hverandre. Dette gjøres vanligvis med metodene setX og setY, men de kan hete noe annet for noen typer kulisser; se oversikten over de aktuelle klassene på side 11.

Forøvrige er det greit å merke seg at koordinatsystemet til JavaFX lar Y-aksen gå *nedover*; se figur 2. Enheter er *skjermpiksler* så en typisk skjerm vil ha 1000–2000 piksler i hver retning.

Figur 2: Koordinatsystemet i JavaFX

1.3.1 Rutenett

Det noen ganger lurt å plassere kulissene (eller i hvert fall noen av dem) i et rektangulært rutenett, slik som vist i figur 5 på side 7. Dette ordnes enkelt med klassen GridPane som vist i Tripp-trapp-tresko-eksemplet på side 7.

1.4 Annet å merke seg

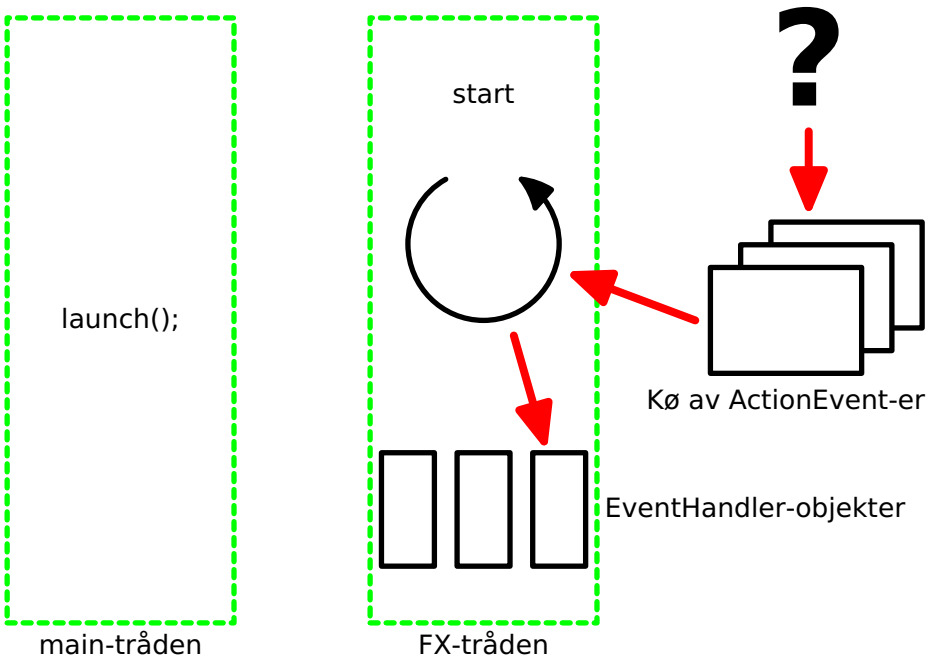
Følgende er nyttig å merke seg:

- Alt i JavaFX ligger i Java-biblioteket, og vi må importere alle de klassene vi trenger. (Det kan fort bli en 10–20 stykker.)
- Den offisielle dokumentasjonen av JavaFX-biblioteket finnes i https://docs.oracle.com/javase/8/javafx/api/toc.htm.
- Hvis du ikke har JavaFX på din private maskin, finner du informasjon om hvordan du kan installere den på https://www.oracle.com/technetwork/java/javafx/install-javafx-sdk-1-2-139156.html.
- All koden fra forelesningene og fra dette kompendiet finnes på https://www-adm.uio.no/studier/emner/matnat/ifi/IN1010/v19/programmer/GUI/.

2 Interaksjon med brukeren

I JavaFX benyttes *eventdrevet programmering* når programmet skal interagere med brukeren; se figur 3. Brukeren må

- deklarere egne subklasser av EventHandler der den redefinerte metoden handle angir hva som skal gjøres når hendelsen inntreffer
- opprette Button-objekter som plasseres blant FX-kulissene
- kople Button-objektene til riktig EventHandler-objekt.



Figur 3: Hendelsesløkken og -køen i JavaFX

2.1 Et enkelt telleverk

I dette programmet skal vi ha et telleverk som økes med 1 hver gang vi klikker på trykknappen merket «Øk teller»; i tillegg har vi to knapper for «Nullstill» og «Stopp».

Figur 4: Et enkelt telleverk

Følgende kode dreier seg om brukerinteraksjonen:

- Linje 12: Denne linjen oppretter feltet der tellerens verdi står. Dette feltet endres i linje 22 og 30.
- Linje 18–24: Klassen NulleBehandler håndterer klikk på knappen merket «Nullstill»; koplingen av de to skjer i linje 51–54.
- Linje 26–32: Klassen TelleBehandler tar seg av klikk på trykknappen merket «Øk teller»; koplingen skjer i linje 46–49.
- Linje 34–39: Klassen StoppBehandler er for klikk på «Stopp»-knappen; koplingen skjer i linje 56–59.

```
1  import javafx.application.Application;
2  import javafx.application.Platform;
3  import javafx.stage.Stage;
4  import javafx.scene.Scene;
5  import javafx.scene.control.Button;
6  import javafx.scene.layout.Pane;
7  import javafx.scene.text.*;
8  import javafx.event.*;
9
10 public class Telleverk extends Application {
11     int teller = 0;
12     Text tellerSomText = new Text("0");
13
14     public static void main(String[] arg) {
15         launch();
16     }
17
18     class NulleBehandler implements EventHandler<ActionEvent> {
19         @Override
20         public void handle(ActionEvent e) {
21             teller = 0;
22             tellerSomText.setText("0");
23         }
24     }
25
26     class TelleBehandler implements EventHandler<ActionEvent> {
27         @Override
28         public void handle(ActionEvent e) {
29             teller++;
30             tellerSomText.setText(""+teller);
31         }
32     }
33
34     class StoppBehandler implements EventHandler<ActionEvent> {
35         @Override
36         public void handle(ActionEvent e) {
37             Platform.exit();
38         }
39     }
40
```

```
41      @Override
42      public void start(Stage teater) {
43          tellerSomText.setFont(new Font(25));
44          tellerSomText.setX(30);  tellerSomText.setY(25);
45
46          Button telleknapp = new Button("Øk teller");
47          telleknapp.setLayoutX(10);  telleknapp.setLayoutY(50);
48          TelleBehandler tell = new TelleBehandler();
49          telleknapp.setOnAction(tell);
50
51          Button nulleknapp = new Button("Nullstill");
52          nulleknapp.setLayoutX(14);  nulleknapp.setLayoutY(80);
53          NulleBehandler nulle = new NulleBehandler();
54          nulleknapp.setOnAction(nulle);
55
56          Button stoppknapp = new Button("Stopp");
57          stoppknapp.setLayoutX(20);  stoppknapp.setLayoutY(110);
58          StoppBehandler stopp = new StoppBehandler();
59          stoppknapp.setOnAction(stopp);
60
61          Pane kulisser = new Pane();
62          kulisser.setPrefSize(90,150);
63          kulisser.getChildren().add(tellerSomText);
64          kulisser.getChildren().add(telleknapp);
65          kulisser.getChildren().add(nulleknapp);
66          kulisser.getChildren().add(stoppknapp);
67
68          Scene scene = new Scene(kulisser);
69
70          teater.setTitle("Teller");
71          teater.setScene(scene);
72          teater.show();
73      }
74  }
```

2.2 Tripp-trapp-tresko

Dette eksemplet viser det enkle spillet *Tripp-trapp-tresko* og lar brukeren spille mot datamaskinen. Datamaskinen er ikke spesielt intelligent programmert; den bare trekker tilfeldig blant de ledige rutene.

Figur 5: Tripp-trapp-tresko

```
1  import javafx.application.Application;
2  import javafx.application.Platform;
3  import javafx.stage.Stage;
4  import javafx.scene.Scene;
5  import javafx.scene.layout.Pane;
6  import javafx.scene.layout.GridPane;
7  import javafx.scene.control.Button;
8  import javafx.scene.text.Text;
9  import javafx.scene.text.Font;
10 import javafx.event.*;
11
12 import java.util.Random;
13
14 /* Brettet:
15       +---+---+---+
16       | 1 | 2 | 3 |
17       +---+---+---+
18       | 4 | 5 | 6 |
19       +---+---+---+
20       | 7 | 8 | 9 |
21       +---+---+---+
22
23    Spillerne:
24       X - maskinen
```

```
25          0 - brukeren
26      */
27
28  public class TTT extends Application {
29      Text statusinfo;
30      Rute brett[];
31      boolean ferdig = false;
32
33      class Rute extends Button {
34          char merke = ' ';
35
36          Rute() {
37              super(" ");
38              setFont(new Font(50));
39              setPrefSize(120, 120);
40          }
41
42          void settMerke(char c) {
43              setText(""+c);  merke = c;
44          }
45      }
46
47      class Klikkbehandler implements EventHandler<ActionEvent> {
48          @Override
49          public void handle(ActionEvent e) {
50              if (! ferdig)
51                  spill0((Rute)e.getSource());
52          }
53      }
54
55      class Stoppbehandler implements EventHandler<ActionEvent> {
56          @Override
57          public void handle(ActionEvent e) {
58              Platform.exit();
59          }
60      }
61
62      public static void main(String[] args) {
63          launch(args);
64      }
65
66      @Override
67      public void start(Stage teater) {
68          statusinfo = new Text("Velg en rute");
69          statusinfo.setFont(new Font(20));
70          statusinfo.setX(10);  statusinfo.setY(410);
71
72          Button stoppknapp = new Button("Stopp");
73          stoppknapp.setLayoutX(10);  stoppknapp.setLayoutY(450);
74          Stoppbehandler stopp = new Stoppbehandler();
75          stoppknapp.setOnAction(stopp);
```

```
76
77          brett = new Rute[9+1];
78          Klikkbehandler klikk = new Klikkbehandler();
79          for (int i = 1;  i <= 9;  i++) {
80              brett[i] = new Rute();
81              brett[i].setOnAction(klikk);
82          }
83
84          GridPane rutenett = new GridPane();
85          rutenett.setGridLinesVisible(true);
86          rutenett.add(brett[1], 0, 0);
87          rutenett.add(brett[2], 1, 0);
88          rutenett.add(brett[3], 2, 0);
89          rutenett.add(brett[4], 0, 1);
90          rutenett.add(brett[5], 1, 1);
91          rutenett.add(brett[6], 2, 1);
92          rutenett.add(brett[7], 0, 2);
93          rutenett.add(brett[8], 1, 2);
94          rutenett.add(brett[9], 2, 2);
95          rutenett.setLayoutX(10);  rutenett.setLayoutY(10);
96
97          Pane kulisser = new Pane();
98          kulisser.setPrefSize(400, 500);
99          kulisser.getChildren().add(rutenett);
100         kulisser.getChildren().add(statusinfo);
101         kulisser.getChildren().add(stoppknapp);
102
103         Scene scene = new Scene(kulisser);
104
105         teater.setTitle("Tripp-trapp-tresko");
106         teater.setScene(scene);
107         teater.show();
108
109         spillX();  // La X starte:
110     }
111
112     void spillO(Rute r) {
113         if (r.merke != ' ') {
114             statusinfo.setText("Ruten er opptatt; velg en annen");
115             return;
116         } else {
117             statusinfo.setText("Velg en rute");
118         }
119
120         r.settMerke('O');
121         if (harVunnet('O')) utropVinner('O');
122
123         if (! ferdig) spillX();
124     }
125
126     Random tilfeldig = new Random();
```

```
127     void spillX() {
128         int p;
129         do {
130             p = tilfeldig.nextInt(9)+1;
131         } while (brett[p].merke != ' ');
132         brett[p].settMerke('X');
133 
134         if (harVunnet('X')) utropVinner('X');
135         else if (erUavgjort()) utropUavgjort();
136     }
137 
138     boolean harVunnet(char c) {
139         return
140             trePaaRad(1, 2, 3, c) ||  // Vannrett
141             trePaaRad(4, 5, 6, c) ||
142             trePaaRad(7, 8, 9, c) ||
143             trePaaRad(1, 4, 7, c) ||  // Loddrett
144             trePaaRad(2, 5, 8, c) ||
145             trePaaRad(3, 6, 9, c) ||
146             trePaaRad(1, 5, 9, c) ||  // Diagonal
147             trePaaRad(3, 5, 7, c);
148     }
149 
150     boolean trePaaRad(int r1, int r2, int r3, char c) {
151         if (brett[r1].merke != c) return false;
152         if (brett[r2].merke != c) return false;
153         if (brett[r3].merke != c) return false;
154         return true;
155     }
156 
157     boolean erUavgjort() {
158         for (int i = 1;  i <= 9;  i++)
159             if (brett[i].merke == ' ') return false;
160         return true;
161     }
162 
163     void utropVinner(char c) {
164         statusinfo.setText(c + " har vunnet!");
165         ferdig = true;
166     }
167 
168     void utropUavgjort() {
169         statusinfo.setText("Det ble uavgjort!");
170         ferdig = true;
171     }
172 }
```

3 Utdrag fra Java FX-biblioteket

Dette er de viktigste klassene vi vil bruke i IN1010, men antallet metoder er kraftig redusert. Dessuten er det bevisst lagt inn noen forenklinger slik at det skal bli greiere å få oversikten.

3.1 class ActionEvent

Denne klassen representerer hendelser, for eksempel å klikke på en Button.

```
// import javafx.event.ActionEvent;

class ActionEvent {
    Object getSource() {
        /* Hvilken hendelse skjedde? */
    }
}
```

3.2 class Application

Denne klassen brukes til å hente inn hele FX-opplegget.

```
// import javafx.application.Application;
// import javafx.stage.Stage;

public abstract class Application {
    abstract void start(Stage primaryStage);
        /* Her plasseres GUI-initieringen. */

    public static void launch(String[] args) {
        /* Kalles fra 'main' for å starte FX. */
    }
}
```

3.3 class Button

Denne klassen definerer en klikkbar bryter.

```
// import javafx.scene.control.Button;
// import javafx.scene.text.Font;
// import javafx.event.*

class Button {
    Button(String text) {
        /* Lager en bryter med angitt tekst. */
    }

    void setFont(Font value) {
        /* Angi font for teksten. */
```

```
    }

    void setOnAction(EventHandler value) {
        /* Angi hvem som skal lytte etter trykk. */
    }

    void setPrefSize(double prefWidth, double prefHeight) {
        /* Angi ønsket størrelse for bryteren. */
    }

    void setText(String value) {
        /* Erstatt teksten med en ny. */
    }

    void setLayoutX(double value) {
        /* Angi x-koordinaten til første tegn i teksten. */
    }

    void setLayoutY(double value) {
        /* Angi y-koordinaten til grunnlinjen i teksten. */
    }
}
```

3.4 class Circle

Denne klassen lager en sirkel.

```
// import javafx.scene.shape.Circle;
// import javafx.scene.paint.Color;

class Circle {
    Circle(double radius) {
        /* En sirkel med gitt radius. */
    }

    void setFill(Color value) {
        /* Angi sirkelens farge. */
    }

    void setStroke(Color value) {
        /* Angi farge på sirkelens rand. */
    }

    void setStrokeWidth(double value) {
        /* Angi hvor tykk sirkelens rand skal tegnes. */
    }

    void setCenterX(double value) {
        /* Angi x-koordinaten til sentrum. */
```

```
    }

    void setCenterY(double value) {
        /* Angi y-koordinaten til sentrum. */
    }
}
```

3.5 class Color

Denne klassen brukes til å lage farger.

```
// import javafx.scene.paint.Color;

class Color {
    final static Color BLACK = rgb(0, 0, 0);
    final static Color WHITE = rgb(255, 255, 255);
        /* De to vanligste fargene. */

    static Color rgb(int red, int green, int blue) {
        /* Lager en farge basert på RGB-verdier. */
    }
}
```

3.6 interface EventHandler

Denne klassen brukes til å definere lyttere for ulike hendelser.

```
// import javafx.event.*

interface EventHandler {
    void handle(ActionEvent event);
        /* Angi hva som skal skje når hendelsen inntreffer. */
}
```

3.7 class FileChooser

Med metoden showOpenDialog i denne klassen kan vi åpne et eget vindu der brukeren kan velge en fil.

```
// import javafx.stage.FileChooser;
// import javafx.stage.Stage;
// import java.io.File;

class FileChooser {
    File showOpenDialog(Stage teater) {
        /* Åpner et dialogvindu for å velge en fil. */
```

```
    }
}
```

3.8 class Font

Lager et objekt som representerer en font.

```
// import javafx.scene.text.Font;

class Font {
    Font(double size) {
        /* Hent systemfonten i angitt størrelse. */
    }
}
```

3.9 class GridPane

Sett opp et rutenett der vi kan sette inn ulike kulisseelementer.

```
// import javafx.scene.layout.GridPane;
// import javafx.scene.layout.Pane;

class GridPane extends Pane {
    void add(Object elem, int col, int row) {
        /* Setter inn i posisjon [col,row] i rutenettet. */
    }

    void setGridLinesVisible(boolean value) {
        /* Angi om selve rutenettet skal tegnes. */
    }

    void setLayoutX(double value) {
        /* Angi ønsket x-koordinat for øvre venstre hjørne. */
    }

    void setLayoutY(double value) {
        /* Angi ønsket y-koordinat for øvre venstre hjørne. */
    }
}
```

3.10 class Line

Denne klassen lager en rett linje.

```
// import javafx.scene.shape.Line;
// import javafx.scene.paint.Color;
```

```
class Line {
    Line(double startX, double startY, double endX, double endY) {
        /* Lag en linje fra (startX,startY) til (endX,endY). */
    }

    void setStroke(Color value) {
        /* Angi farge på linjen. */
    }

    void setStrokeWidth(double value) {
        /* Angi hvor tykk linjen skal tegnes. */
    }
}
```

3.11 class Pane

Lager en kulisseflate der vi kan henge kulisseelementer.

```
// import javafx.scene.layout.Pane;
// import java.util.List;

class Pane {
    List getChildren() {
        /* Få listen der kulisseelementene ligger. */
    }

    void setPrefSize(double prefWidth, double prefHeight) {
        /* Angi et ønske om størrelsen. */
    }
}
```

3.12 class Platform

Denne klassen inneholder noen nyttige hjelpemetoder for Application.

```
// import javafx.application.Platform;

class Platform {
    static void exit() {
        /* Avslutter JavaFX på en pen måte. */
    }

    static void runLater(Runnable runnable) {
        /* Opprett en ny Event som vil bli kjørt en gang. */
    }
}
```

3.13 class Polygon

Denne klassen lager en mangekant (et polygon).

```
// import javafx.scene.shape.Polygon;
// import javafx.scene.paint.Color;

class Polygon {
    Polygon(double... points) {
        /* Lag et polygon av de angitt punktene. */
    }

    void setFill(Color value) {
        /* Angi polygonets farge. */
    }

    void setStroke(Color value) {
        /* Angi farge på polygonets rand. */
    }

    void setStrokeWidth(double value) {
        /* Angi hvor tykk polygonets rand skal tegnes. */
    }
}
```

3.14 class Rectangle

Denne klassen lager en rettvinklet firkant (dvs et rektangel).

```
// import javafx.scene.shape.Rectangle;
// import javafx.scene.paint.Color;

class Rectangle {
    Rectangle(double width, double height) {
        /* Opprett et rektangel med gitte dimensjoner. */
    }

    void setFill(Color value) {
        /* Angi rektangelets farge. */
    }

    void setStroke(Color value) {
        /* Angi farge på rektangelets rand. */
    }

    void setStrokeWidth(double value) {
        /* Angi hvor tykk rektangelets rand skal tegnes. */
    }

    void setX(double value) {
```

```
22          /* Angi x-koordinaten til øvre venstre hjørne. */
23      }
24
25      void setY(double value) {
26          /* Angi y-koordinaten til øvre venstre hjørne. */
27      }
28  }
```

3.15 class Scene

Definerer en scene der vi kan sette opp kulisser.

```
1   // import javafx.scene.Scene;
2   // import javafx.scene.layout.Pane;
3
4   class Scene {
5       Scene(Pane kulisser) {
6           /* Lag en scene med angitte kulisser. */
7       }
8   }
```

3.16 class Stage

Denne klassen gir hovedrammen for GUI-vinduet. Systemet vil opprette dette objektet for oss.

```
1   // import javafx.stage.Stage;
2   // import javafx.scene.Scene;
3
4   class Stage {
5       void setScene(Scene value) {
6           /* Angir hvilken scene som skal vises. */
7       }
8
9       void setTitle(String value) {
10          /* Angir tittel til GUI-vinduet. */
11      }
12
13      void show() {
14          /* Viser det som er i GUI-vinduet. */
15      }
16  }
```

3.17 class Text

Lager en tekst.

```
// import javafx.scene.text.Text;
// import javafx.scene.text.Font;

class Text {
    Text(String text) {
        /* En tekst med angitt innhold. */
    }

    void setFont(Font value) {
        /* Angi font for teksten. */
    }

    void setText(String value) {
        /* Erstatt teksten med en ny. */
    }

    void setX(double value) {
        /* Angi x-koordinaten til første tegn i teksten. */
    }

    void setY(double value) {
        /* Angi y-koordinaten til grunnlinjen i teksten. */
    }
}
```

Innhold