

# Uvod u JavaFX

---

# Sadržaj

---

Grafička sučelja u Javi

GridPane

Prednosti JavaFX-a

Prva JavaFX aplikacija

FXML

Organizatori rasporeda komponenti

HBox

VBox

FlowPane

BorderPane

# Grafička sučelja u Javi

---

- Aplikacije s grafičkim sučeljem (engl. *Graphical user interface* - GUI) omogućavaju korištenje svojih funkcionalnosti kroz grafičke elemente koji se nazivaju kontrole (engl. *controls*) ili widgeti (engl. *widgets* – *Window Gadgets*)
- Svaka GUI komponenta je također objekt koja se može koristiti pomoću miša i tipkovnice
- U Javu je od samih početaka uveden GUI *library* pod nazivom AWT (engl. *Abstract Window Toolkit*)
- Od Java SE 1.2 koristi se Swing *library* koji je bio primarni skup alata za razvoj grafičkih sučelja u Javi sve do Java SE 7
- Swing se trenutno nalazi samo u fazi održavanja, više se ne razvija, ali će ostati još neko vrijeme kao dio Jave radi unazadne kompatibilnosti

# Grafička sučelja u Javi

---

- JavaFX, grafički i multimedijski API (engl. *Application Programming Interface*) prvi put je objavljen 2007. godine kao konkurentna tehnologija Adobe Flashu i Microsoft Silverlightu
- Prva verzija 1.0 objavljena je 2008. godine
- Prije verzije 2.0 koristio se JavaFX Script koji je bio sličan JavaScriptu i prevodio je izvorni kod u *bytecode* i mogao se izvoditi na JVM-u
- Od verzije 2.0 koja je objavljena 2011. godine JavaFX je implementiran kao skupina librarya
- Objavom Jave 8 objavljena je inačica JavaFX 8

# Prednosti JavaFX-a

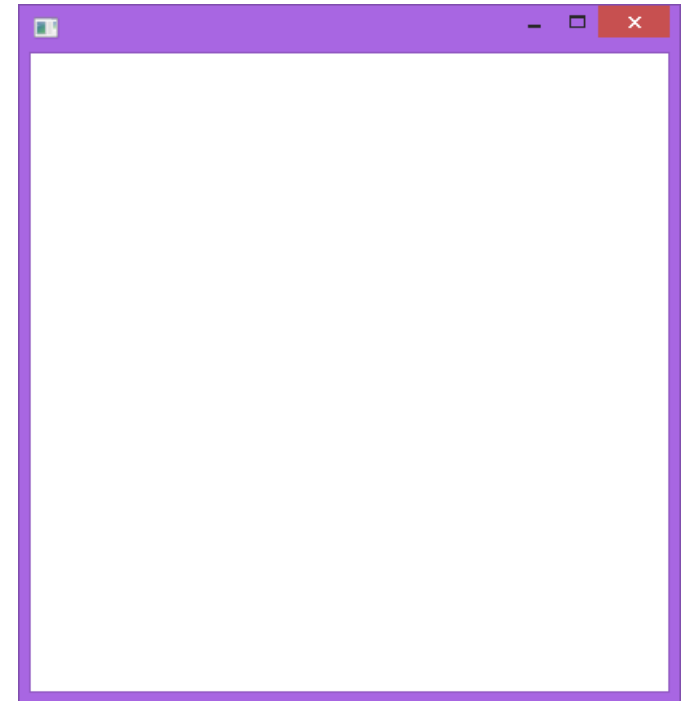
---

- Uključuje Scene Builder za dizajniranje grafičkog sučelja aplikacije koji može biti korišten odvojeno ili integriran unutar razvojnog okruženja
- Omogućava korištenje CSS-a (engl. *Cascading Style Sheets*) za oblikovanje komponenti grafičkog sučelja
- Omogućava napredno korištenje niti u sustavima koji imaju procesor s više jezgara
- Podržava razne transformacije nad komponentama i animacije kojima je moguće mijenjati svojstva grafičkih elemenata

# Prva JavaFX aplikacija

---

```
public class Main extends Application {  
    @Override  
    public void start(Stage primaryStage) {  
        try {  
            BorderPane root = new BorderPane();  
            Scene scene = new Scene(root, 400, 400);  
            scene.getStylesheets().add(getClass().getResource(  
                "application.css").toExternalForm());  
            primaryStage.setScene(scene);  
            primaryStage.show();  
        } catch (Exception e) {  
            e.printStackTrace();  
        }  
    }  
  
    public static void main(String[] args) {  
        launch(args);  
    }  
}
```



# Prva JavaFX aplikacija

---

- Svaka klasa koja predstavlja polaznu točku JavaFX aplikacije mora nasljeđivati klasu „javafx.application.Application” koja omogućava funkcionalnosti životnog ciklusa aplikacije kao što su inicijalizacija, pokretanje ili zaustavljanje izvođenja aplikacije
- Omogućava upravljanje aplikacijom odvojeno od glavne niti
- Sadrži metodu „launch” koja poziva nadjačanu metodu „start”
- Metoda „start” prima objekt klase „javafx.stage.Stage” koji omogućava pokretanje zasebne niti koja će se izvoditi na grafičkom sučelju aplikacije
- Objekt klase „Scene” potrebno je dodati u objekt klase „Stage” koji omogućava prikaz grafičkog sučelja
- Koncept JavaFX Scene Graph omogućava korištenje više različitih „Scene” objekata na jednom „Stage” objektu

# Prva JavaFX aplikacija

---

- Objekti klase „Scene” omogućavaju postavljanje veličine ekrana, naslova, kao i samog organizatora rasporeda komponenti (engl. *Layout*), npr.: „BorderPane”
- Osim toga omogućeno je definiranje lokacije i naziva CSS datoteke u kojoj se nalaze definicije stilova koji se koriste na grafičkom sučelju:

```
scene.getStylesheets().add(getClass().getResource(  
    "application.css").toExternalForm());
```

- CSS stilovi koji se koriste kod JavaFX-a moraju sadržavati „-fx” prefiks, a mogu se primjenjivati korištenjem metode „setStyle”, npr:

```
root.setStyle("-fx-background: rgb(225, 228, 203)");  
primaryStage.setScene(scene);
```



# FXML

---

- Scene Builder omogućava dizajniranje grafičkog sučelja aplikacije korištenjem posebno oblikovane XML datoteke koja ima ekstenziju „.fxml” – *FX Markup Language*
- Omogućava odvajanje prezentacijske logike od programske logike u JavaFX aplikacijama
- Da bi se mogao koristiti, FXML se najprije treba uključiti na samom grafičkom sučelju aplikacije, npr.:

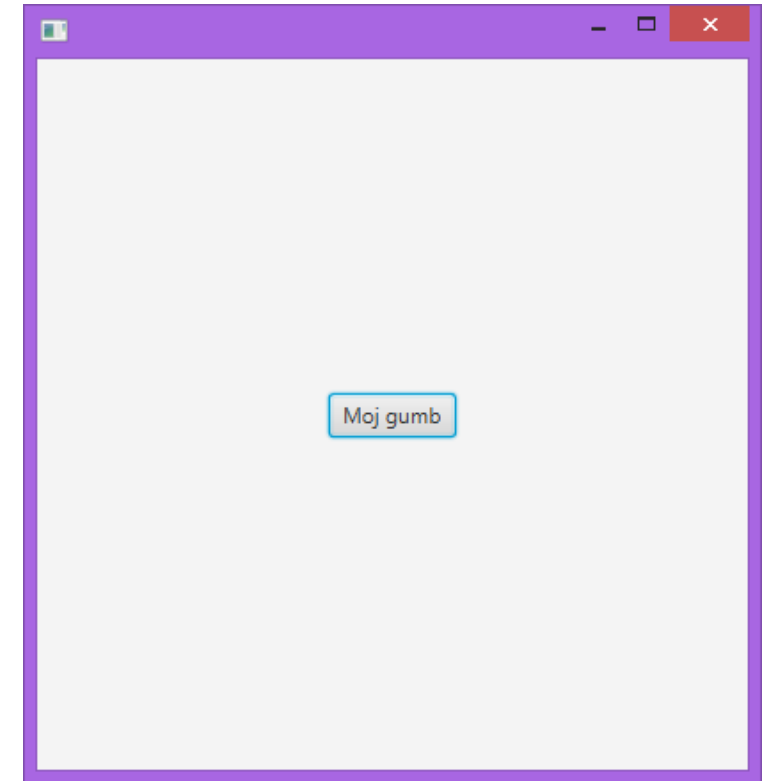
```
BorderPane root =  
    (BorderPane)FXMLLoader.Load(getClass().getResource("Sample.fxml"));
```

# FXML

---

- Primjer izgleda FXML datoteke koja sadrži samo jedan gumb:

```
<?xml version="1.0" encoding="UTF-8"?>
<?import javafx.scene.control.*?>
<?import java.lang.*?>
<?import javafx.scene.layout.*?>
<?import javafx.scene.layout.BorderPane?>
<BorderPane prefHeight="287.0" prefWidth="286.0"
xmlns:fx="http://javafx.com/fxml/1"
xmlns="http://javafx.com/javafx/8"
fx:controller="application.SampleController">
    <center>
        <Button mnemonicParsing="false" text="Moj gumb"
            BorderPane.alignment="CENTER" />
    </center>
</BorderPane>
```



# Organizatori rasporeda komponenti

---

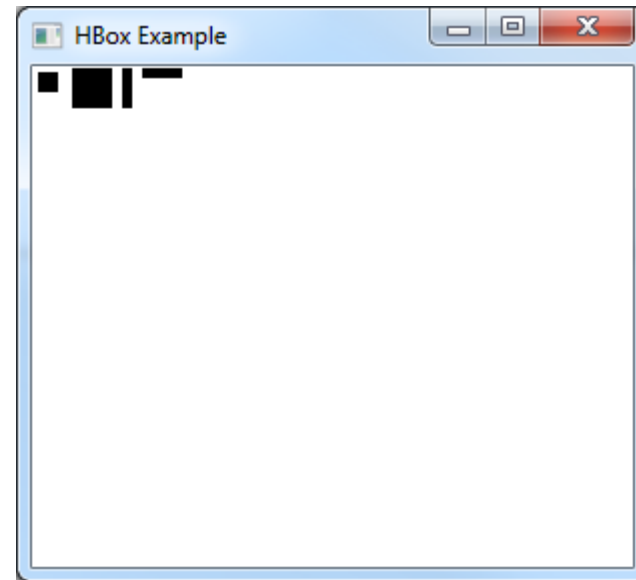
- Služe za postavljanje grafičkih elemenata na grafičko sučelje u određenom rasporedu
- Omogućavaju dinamičko povećanje prostora za prikazivanje komponenti u slučaju povećanja prozora na kojem se prikazuju
- Najčešće korišteni organizatori su:
  - `javafx.scene.layout.HBox`
  - `javafx.scene.layout.VBox`
  - `javafx.scene.layout.FlowPane`
  - `javafx.scene.layout.BorderPane`

# HBox

---

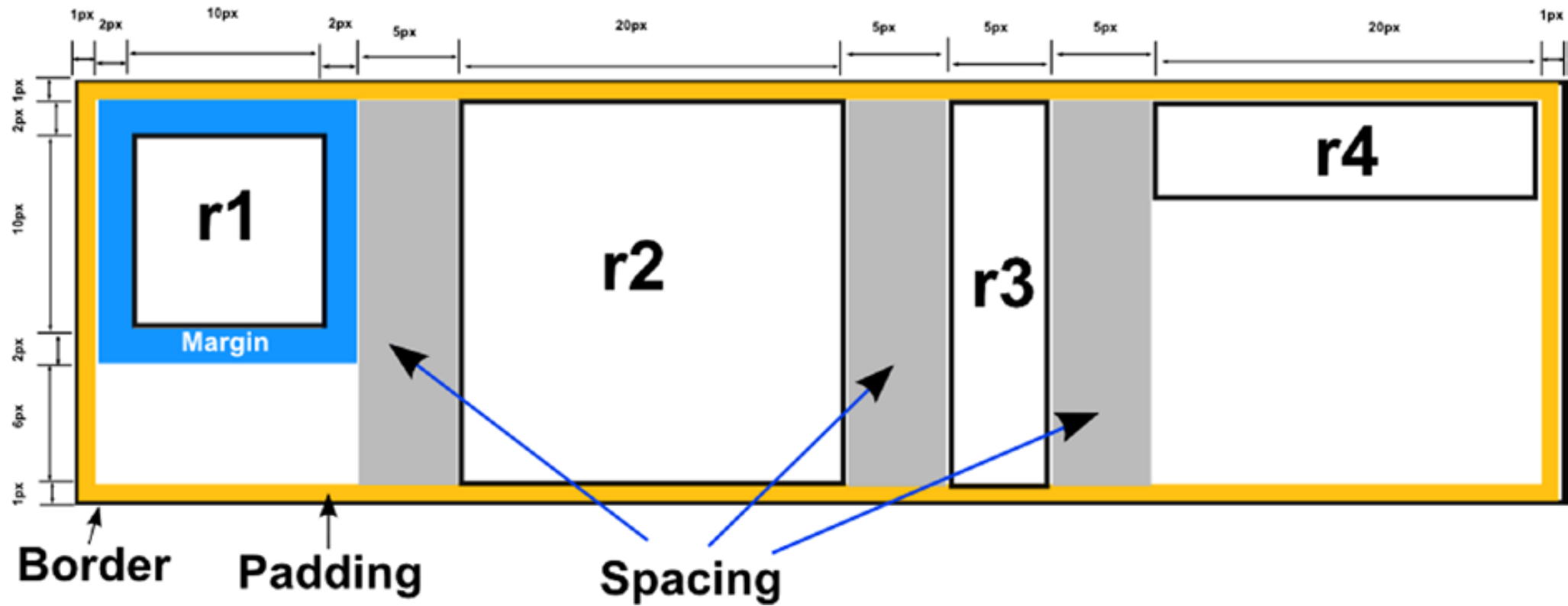
- Postavlja elemente u horizontalni redak, jedan iza drugog
- Omogućava definiranje razmaka između komponenata koje se mogu mijenjati u ovisnosti o veličini komponente (prozora u kojem se nalaze)
- Primjer korištenja:

```
HBox hbox = new HBox(5);  
hbox.setPadding(new Insets(1));  
Rectangle r1 = new Rectangle(10, 10);  
Rectangle r2 = new Rectangle(20, 20);  
Rectangle r3 = new Rectangle(5, 20);  
Rectangle r4 = new Rectangle(20, 5);  
HBox.setMargin(r1, new Insets(2,2,2,2));  
hbox.getChildren().addAll(r1, r2, r3, r4);
```



# HBox

- Definiranje razmaka između komponenata:

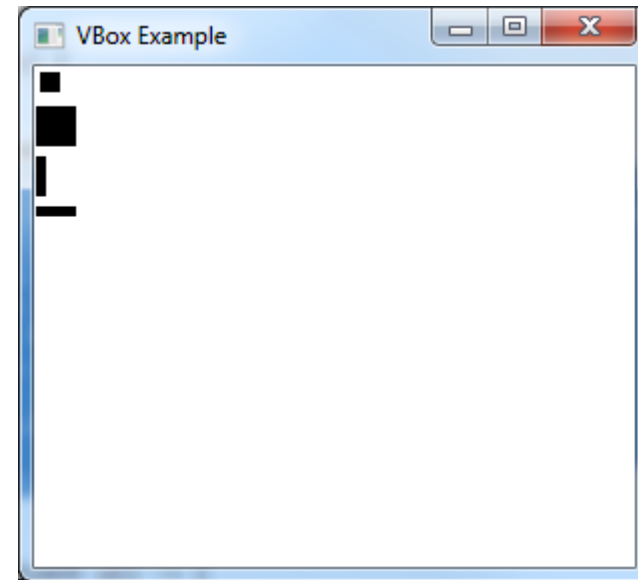


# VBox

---

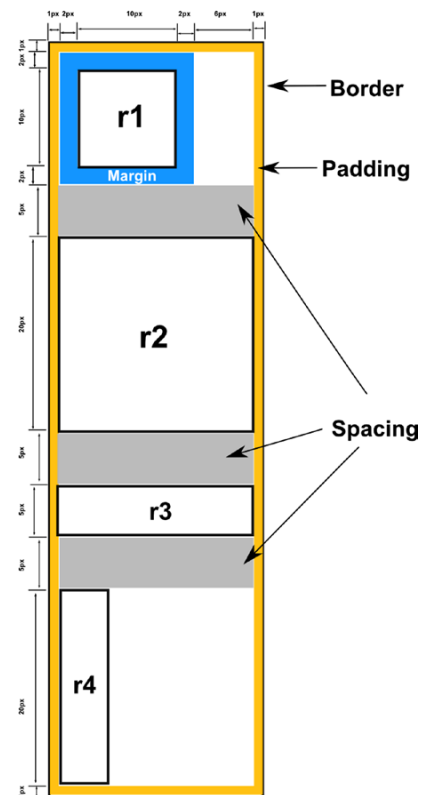
- Slično kao i u slučaju HBox komponente, postavlja elemente u stupac, jednog ispod drugog
- Primjer korištenja:

```
VBox vbox = new VBox(5);  
vbox.setPadding(new Insets(1));  
  
Rectangle r1 = new Rectangle(10, 10);  
Rectangle r2 = new Rectangle(20, 20);  
Rectangle r3 = new Rectangle(5, 20);  
Rectangle r4 = new Rectangle(20, 5);  
  
VBox.setMargin(r1, new Insets(2,2,2,2));  
  
vbox.getChildren().addAll(r1, r2, r3, r4);
```



# VBox

- Definiranje razmaka između komponenata:

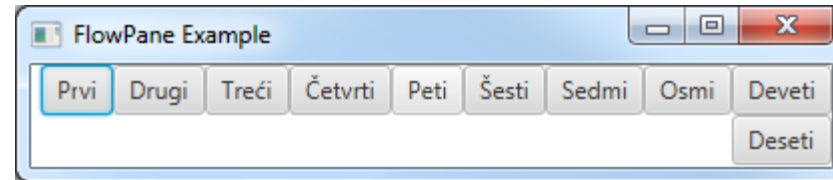


# FlowPane

---

- Omogućava postavljanje komponenti jedne iza druge do kraja raspoloživog retka, nakon čega nastavlja u sljedećem retku
- Primjer korištenja:

```
FlowPane flowPane = new FlowPane();
Button b1 = new Button("Prvi");
Button b2 = new Button("Drugi");
Button b3 = new Button("Treći");
Button b4 = new Button("Četvrti");
Button b5 = new Button("Peti");
Button b6 = new Button("Šesti");
Button b7 = new Button("Sedmi");
Button b8 = new Button("Osmi");
Button b9 = new Button("Deveti");
Button b10 = new Button("Deseti");
flowPane.setAlignment(Pos.TOP_RIGHT);
flowPane.getChildren().addAll(b1, b2, b3, b4, b5, b6, b7, b8, b9, b10);
```





# BorderPane

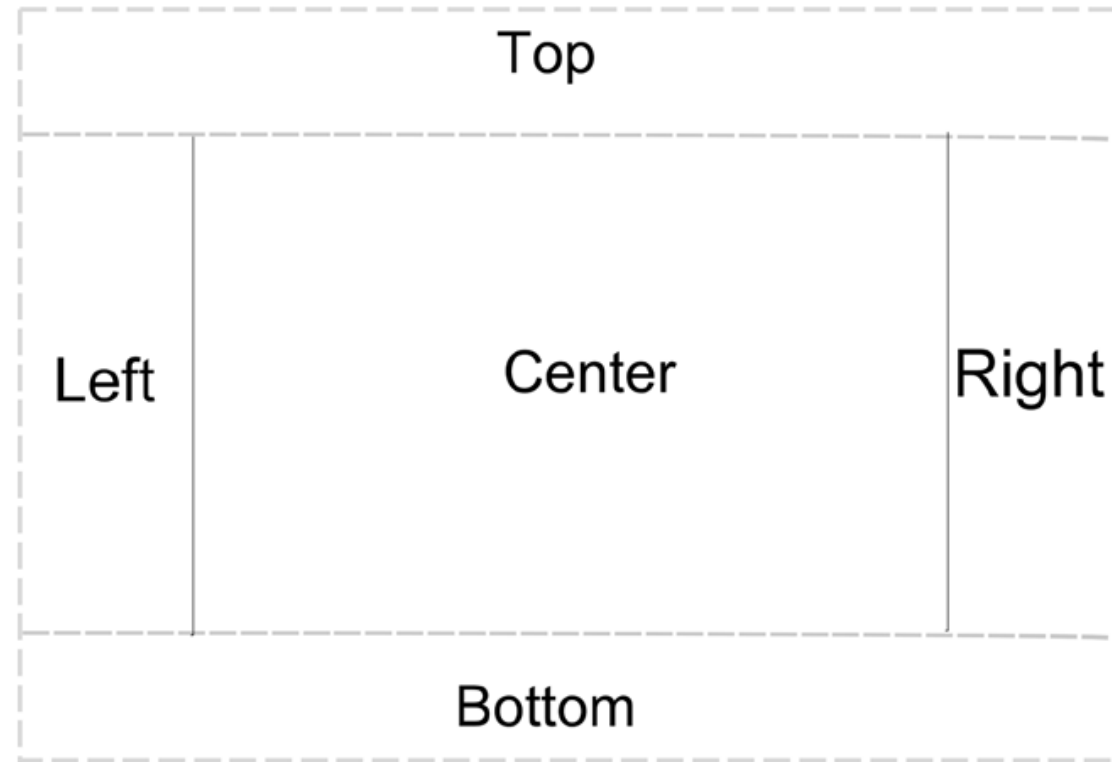
---

- Omogućava postavljanje komponenti u pet različitih područja:
  - TOP
  - LEFT
  - CENTER
  - RIGHT
  - BOTTOM
- Iako svako područje može imati samo jednu komponentu, moguće je u nju ubaciti drugi ugniježđeni organizator rasporeda komponenti
- Mogu se koristiti koncepti kao i kod dizajniranja web stranica

# BorderPane

---

- Dostupna područja za postavljanje elemenata:



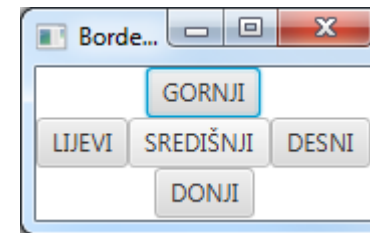
# BorderPane

---

- Primjer korištenja:

```
BorderPane borderPane = new BorderPane();
Button b1 = new Button("GORNJI");
Button b2 = new Button("LIJEVI");
Button b3 = new Button("SREDIŠNJI");
Button b4 = new Button("DESNI");
Button b5 = new Button("DONJI");

BorderPane.setAlignment(b1, Pos.TOP_CENTER);
borderPane.setTop(b1);
borderPane.setLeft(b2);
borderPane.setCenter(b3);
borderPane.setRight(b4);
BorderPane.setAlignment(b5, Pos.BOTTOM_CENTER);
borderPane.setBottom(b5);
```



# GridPane

---

- Omogućava umetanje elemenata na grafičko sučelje korištenjem „tablice” koja ima određen broj redaka i stupaca
- Kod dodavanja elementa na grafičko sučelje je potrebno definirati „koordinate” ćelije tablice u koju se dodaje
- Svakom stupcu moguće je definirati minimalnu, maksimalnu širinu u slučaju da se mijenja veličina samog ekrana u kojem se komponente nalaze
- Prikladan za kreiranje ekrana koji sadrže „forme” za unos podataka

# GridPane

---

- Primjer korištenja:

```
GridPane gridpane = new GridPane();

Label fNameLbl = new Label("First Name");
TextField fNameFld = new TextField();
Label lNameLbl = new Label("Last Name");
TextField lNameFld = new TextField();

Button saveButt = new Button("Save");

GridPane.setHalignment(fNameLbl, HPos.RIGHT);
gridpane.add(fNameLbl, 0, 0);
```

# GridPane

---

```
GridPane.setHalignment(lNameLbl, HPos.RIGHT);  
gridpane.add(lNameLbl, 0, 1);
```

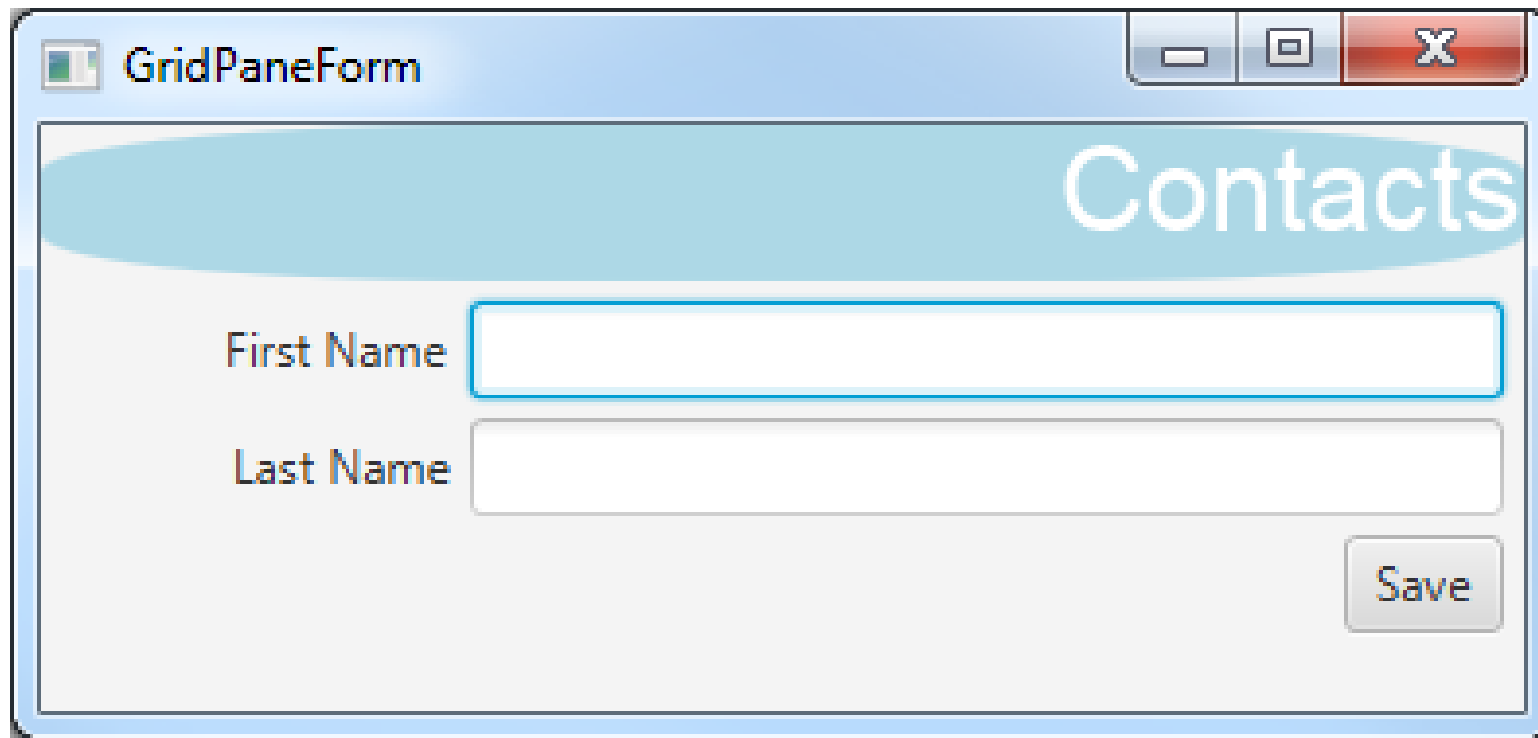
```
GridPane.setHalignment(fNameFld, HPos.LEFT);  
gridpane.add(fNameFld, 1, 0);
```

```
GridPane.setHalignment(lNameFld, HPos.LEFT);  
gridpane.add(lNameFld, 1, 1);
```

# GridPane

---

- Primjer izgleda prozora:



The image shows a Java Swing window titled "GridPaneForm". The window has a standard Mac OS-style title bar with minimize, maximize, and close buttons. The main content area has a light blue header with the word "Contacts" in white. Below the header, there are two text input fields. The first field is labeled "First Name" and the second is labeled "Last Name". Both labels are in a dark blue font. To the right of the input fields is a "Save" button with a light gray background and a dark gray border. The entire form is set against a light gray background.

# Pitanja?

---