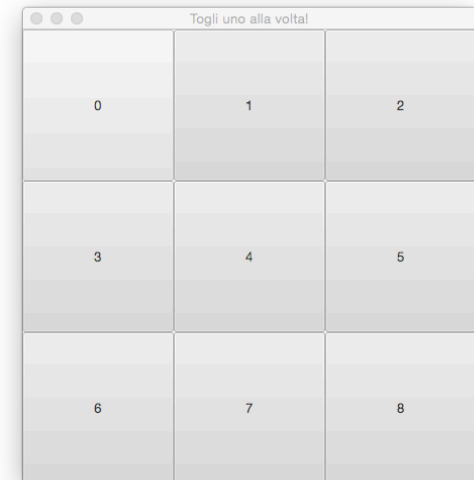


Esercizio 1

- Scrivere una applicazione che mostra 9 bottoni numerati da 0 a 8. Premendo un tasto numerico sulla tastiera, il bottone relativo deve essere rimosso, mentre gli altri devono restare al loro posto
- (ovviamente non devono essere generati errori premendo più volte lo stesso tasto, o premendo tasti non numerici)



```
package it.unitn.disi.lingProg.ronchet
```

```
Import...
```

```
public class AggiungiTogli extends Application {
```

```
    Button[] btn;
```

```
    final int NBUTTONS = 9;
```

```
    @Override
```

```
    public void start(Stage primaryStage) {
```

```
        // CREA BOTTONI
```

```
        // CREA PANNELLO (root)
```

```
        // GESTISCI GLI EVENTI
```

```
        // CREA STAGE
```

```
        // CREA SCENA
```

```
    }
```

```
    public static void main(String[] args) {
```

```
        Application.launch(args);
```

```
    }
```

```
    private Button createButton(int i) {...}
```

```
}
```

```
    btn = new Button[NBUTTONS];  
    for (int i = 0; i < NBUTTONS; i++) {  
        btn[i] = createButton(i);  
    }  
    // AGGIUNGERE IL METODO createButton
```

```
    Scene scene = new Scene(root);  
    primaryStage.setTitle("Togli uno alla volta!");  
    primaryStage.setScene(scene);  
    primaryStage.show();
```

```
    private Button createButton(int i) {  
        String text = "" + i;  
        Button btn = new Button(text);  
        btn.setPrefWidth(200);  
        btn.setPrefHeight(200);  
        return btn;  
    }
```

```
// CREA PANNELLO (root)
// GESTISCI GLI EVENTI
final Pane root = new TilePane();
root.setPrefTileWidth(100);
root.setPrefTileHeight(100);
root.setPrefColumns(3);
root.getChildren().addAll(btn);
root.addEventHandler(KeyEvent.KEY_PRESSED, new
    EventHandler<KeyEvent>() {
        public void handle(KeyEvent keyEvent) {
            KeyCode keyCode = keyEvent.getCode();
            int i = Integer.parseInt(keyCode.getName());
            System.out.println(i);
            boolean retval = root.getChildren().remove(btn[i]);
        }
    });
```

```
// CREA PANNELLO (root)
// GESTISCI GLI EVENTI
final Pane root = new TilePane();
root.setPrefTileWidth(100);
root.setPrefTileHeight(100);
root.setPrefColumns(3);
root.getChildren().addAll(btn);
root.addEventHandler(KeyEvent.KEY_PRESSED, new
    EventHandler<KeyEvent>() {
        public void handle(KeyEvent keyEvent) {
            KeyCode keyCode = keyEvent.getCode();
            // ATTENTIONPOINT 1
            if (!(keyCode.isDigitKey())) return;
            int i = Integer.parseInt(keyCode.getName());
            System.out.println(i);
            boolean retval = root.getChildren().remove(btn[i]);
        }
    });
```

```
// CREA PANNELLO (root)
// GESTISCI GLI EVENTI
final Pane root = new TilePane();
root.setPrefTileWidth(100);
root.setPrefTileHeight(100);
root.setPrefColumns(3);
root.getChildren().addAll(btn);
root.addEventHandler(KeyEvent.KEY_PRESSED, new
    EventHandler<KeyEvent>() {
        public void handle(KeyEvent keyEvent) {
            KeyCode keyCode = keyEvent.getCode();
            // ATTENTIONPOINT 1
            if (!(keyCode.isDigitKey())) return;
            int i = Integer.parseInt(keyCode.getName());
            System.out.println(i);
            boolean retval = root.getChildren().remove(btn[i]);
        }
    });
```

```
// CREA PANNELLO (root)
// GESTISCI GLI EVENTI
final Pane root = new TilePane();
root.setPrefTileWidth(100);
root.setPrefTileHeight(100);
root.setPrefColumns(3);
root.getChildren().addAll(btn);
root.addEventHandler(KeyEvent.KEY_PRESSED, new
    EventHandler<KeyEvent>() {
        public void handle(KeyEvent keyEvent) {
            KeyCode keyCode = keyEvent.getCode();
            // ATTENTIONPOINT 1
            if (!(keyCode.isDigitKey())) return;
            int i = Integer.parseInt(keyCode.getName());
            System.out.println(i);
            // ATTENTIONPOINT 2 - A
            root.getChildren().remove(btn[i]);
            root.getChildren().add(i, new Pane());
        }
    });
```

```
// CREA PANNELLO (root)
// GESTISCI GLI EVENTI
final Pane root = new TilePane();
root.setPrefTileWidth(100);
root.setPrefTileHeight(100);
root.setPrefColumns(3);
root.getChildren().addAll(btn);
root.addEventHandler(KeyEvent.KEY_PRESSED, new
    EventHandler<KeyEvent>() {
        public void handle(KeyEvent keyEvent) {
            KeyCode keyCode = keyEvent.getCode();
            // ATTENTIONPOINT 1
            if (!(keyCode.isDigitKey())) return;
            int i = Integer.parseInt(keyCode.getName());
            System.out.println(i);
            // ATTENTIONPOINT 2 - B
            boolean retval = root.getChildren().remove(btn[i]);
            System.out.println(retval);
            if (retval) {
                root.getChildren().add(i, new Pane());
            }
        }
    });
```

Ristrutturiamo il codice

```
class MyButton extends Button {  
    MyButton(int i) {  
        super("" + i);  
        String text = "" + i;  
        setPrefWidth(200);  
        setPrefHeight(200);  
    }  
}
```


Ristrutturiamo il codice

```
class MyPane extends TilePane {  
    MyPane() {  
        setPrefTileWidth(100);  
        setPrefTileHeight(100);  
        setPrefColumns(3);  
        addEventHandler(KeyEvent.KEY_PRESSED, new EventHandler<KeyEvent>() {  
            public void handle(KeyEvent keyEvent) {  
                KeyCode keyCode = keyEvent.getCode();  
                if (!(keyCode.isDigitKey())) return;  
                int i = Integer.parseInt(keyCode.getName());  
                boolean retval = getChildren().remove(btn[i]);  
                if (retval) getChildren().add(i, new Pane());  
            }  
        });  
    }  
}
```

Ristrutturiamo il codice

```
package it.unitn.disi.lingProg.ronchet
public class AggiungiTogli extends Application {
    Button[] btn;
    final int NBUTTONS = 9;
    @Override
    public void start(Stage primaryStage) {
        btn = new Button[NBUTTONS];
        for (int i = 0; i < NBUTTONS; i++) btn[i] = new MyButton(i);
        final Pane root = new MyPane();
        root.getChildren().addAll(btn);
        Scene scene = new Scene(root);
        primaryStage.setTitle("Togli uno alla volta!");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
    public static void main(String[] args) {
        Application.launch(args);
    }
    private Button createButton(int i) {...}
}
class MyButton ...
class MyPane ...
```

Lettura raccomandata!














Java Platform, Standard Edition (Java SE) 8

JavaFX: Working with Layouts in JavaFX

Table of Contents

<http://docs.oracle.com/javase/8/javafx/layout-tutorial/index.html>

Expand | Collapse

-  Title and Copyright Information
-  Preface
-  1 Using Built-in Layout Panes
-  2 Tips for Sizing and Aligning Nodes
 -  Sizing Nodes
 -  Aligning Content
 -  Additional Resources
 -  Application Files
-  3 Styling Layout Panes with CSS
 -  A LayoutSample.java
 -  B LayoutSizingAligning.java
 -  C LayoutSampleCSS.java
 -  D layoutstyles.css

Per chi vuole saperne di più...

Java Platform, Standard Edition (Java SE) 8

Home Client Technologies Embedded All Books

JavaFX

- Getting Started with JavaFX
 - What Is JavaFX
 - Get Started with JavaFX
 - Get Acquainted with JavaFX Architecture
 - Deployment Guide
- Graphics
 - Getting Started with JavaFX 3D Graphics
 - Use the Image Ops API
 - Work with Canvas
- User Interface Components
 - Work with UI Controls
 - Create Charts
 - Add Text
 - Add HTML Content
 - Work with Layouts
 - Skin Applications with CSS
 - Build UI with FXML
 - Handle Events
- Effects, Animation, and Media
 - Create Visual Effects
 - Add 2D & 3D Transformations
 - Add Transitions & Animation
 - Incorporate Media
- Application Logic
 - Work with the Scene Graph

Swing and 2D

- Getting Started with Swing
- Use Swing Components
- Use Concurrency in Swing
- Work with Advanced Swing Features
- Work with Swing Components Within a Window and a JComponent
- Write Swing Applications
- Work with Java 2D
- Getting Started with Graphics
- Work with Geometry
- Work with Text APIs
- Work with Images
- Print Graphics
- Learn Advanced Topics in Java 2D

JavaFX Scene Builder 2

- Getting Started with Scene Builder
- Work with Scene Builder
- Release Documentation
 - Install Scene Builder
 - Release Notes

<http://docs.oracle.com/javase/8/javase-clienttechnologies.htm>

Esercizio 2

- Scrivere una applicazione che mostra 8 bottoni numerati da 1 a 8 ed uno spazio vuoto. Premendo un tasto numerico sulla tastiera adiacente allo spazio vuoto, il tasto si sposta sullo spazio vuoto e viceversa.

Premendo un tasto non adiacente allo spazio vuoto, non accade nulla (opzionalmente: si sente un beep o un messaggio vocale di errore).

Situazioni anomale vanno gestite.

