A proposito dell’esame…

Esame - Materiale

1a prova:

- Orologio
- Penne
- Documento
- N.Matricola

2a prova:

- Tutto il materiale cartaceo che volete – solo per uso personale.
- API in locale, in Netbeans. Java Tutorials

Esame - Iscrizione

Assumete le vostre responsabilità!

- **Iscrizione (fino a 3 giorni prima)**
- **Dis-iscrizione (fino a 24 ore prima, eventualmente via mail. Last minute – only for serious reasons-, via mail.)**

No excuses. No exceptions.

Esame - Iscrizione

Assumete le vostre responsabilità!

- **Chi, essendosi iscritto, non si presenta senza una ottima giustificazione salta l'appello successivo**

No excuses. No exceptions.

Date degli esami

- Esse3, ma anche sul sito di questo corso

DATE ESAMI [display options]

L'esame complessivo (12 crediti) richiede il superamento di entrambi i moduli (1, tenuto dal prof. Ronchetti, e 2, tenuto dal prof. Kuper). I due moduli possono essere superati imn qualunque ordine, unico vincolo è che se uno dei due è superato in data X, l'altro deve essere superato entro X + un anno (con un minimo di tolleranza: se il primo è superato nel secondo appello estivo dell'anno solare A, il secondo va superato entro il secondo appello estivo dell'anno A+1.

Un volta superati entrambi i moduli, lo studente deve iscriversi al primo appello di registrazione esame disponibile. Non dimenticate di iscrivervi per la registrazione!

Modulo 1 (Ronchetti)

- Primo appello: 14 giugno
- Secondo appello: 1 luglio
- Terzo appello: 6 settembre
- Quarto appello: gennaio 2020, data da definire
- Quinto appello: febbraio 2020, data da definire

Appelli di registrazione esami:

- 30 giugno
- 10 agosto
- 13 settembre
- gennaio 2020, data da definire
- febbraio 2020, data da definire

Per comodità degli studenti, riportiamo qui anche le date degli appelli del prof. Kuper.

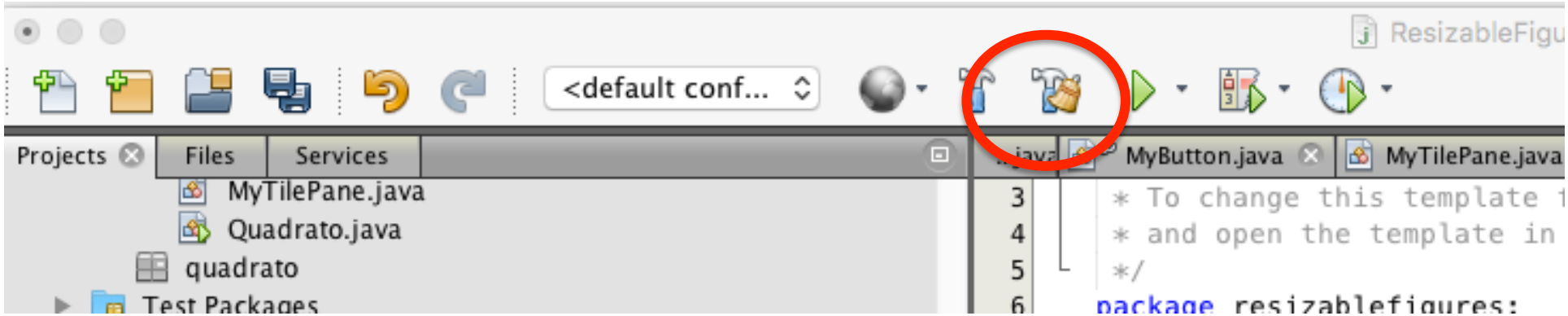
Modulo 2 (Kuper)

- Primo appello: 19 giugno
- Secondo appello: 24 luglio
- Terzo appello: 3 settembre
- Quarto appello: gennaio 2020, data da definire
- Quinto appello: febbraio 2020, data da definire

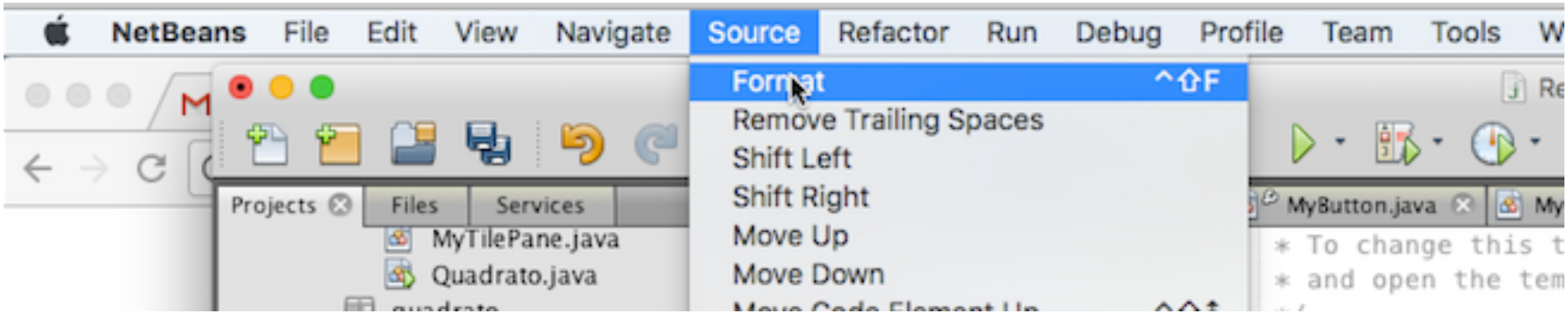
VERIFICARE COMUNQUE SU ESSE3 EVENTUALI VARIAZIONI!

Netbeans tricks

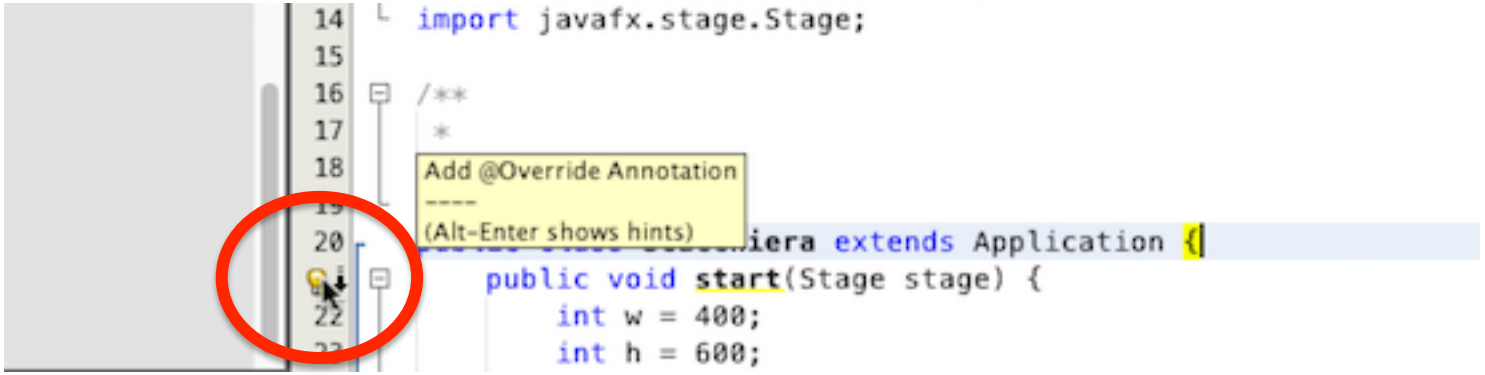
Always fully recompile! (Javafx)



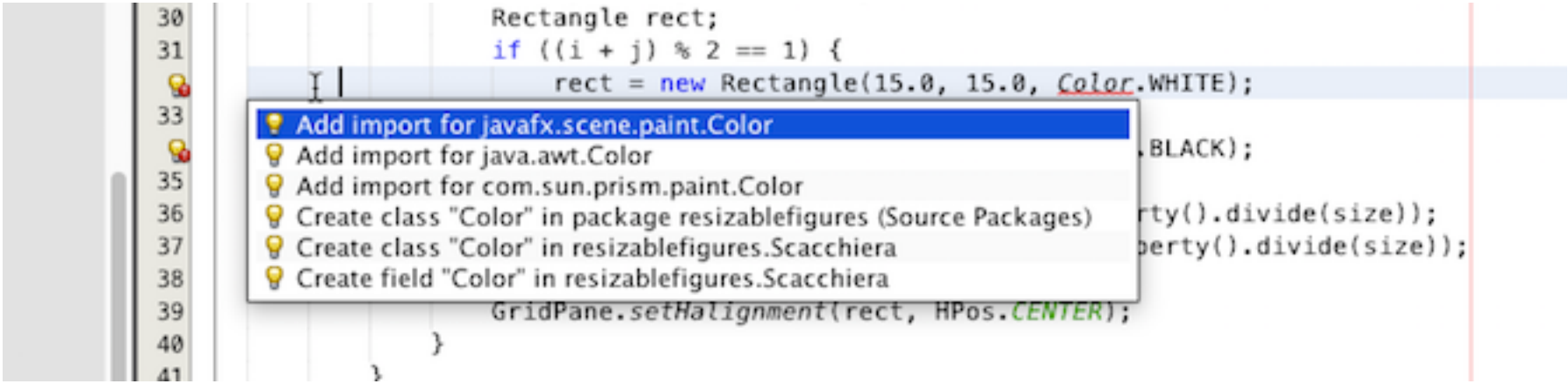
Format!



Read the suggestions!



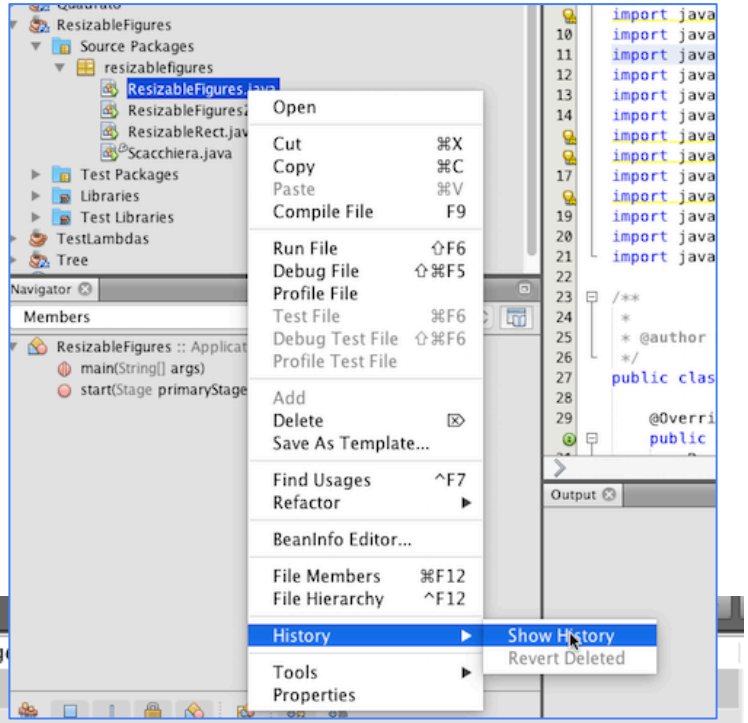
Be careful with the imports!

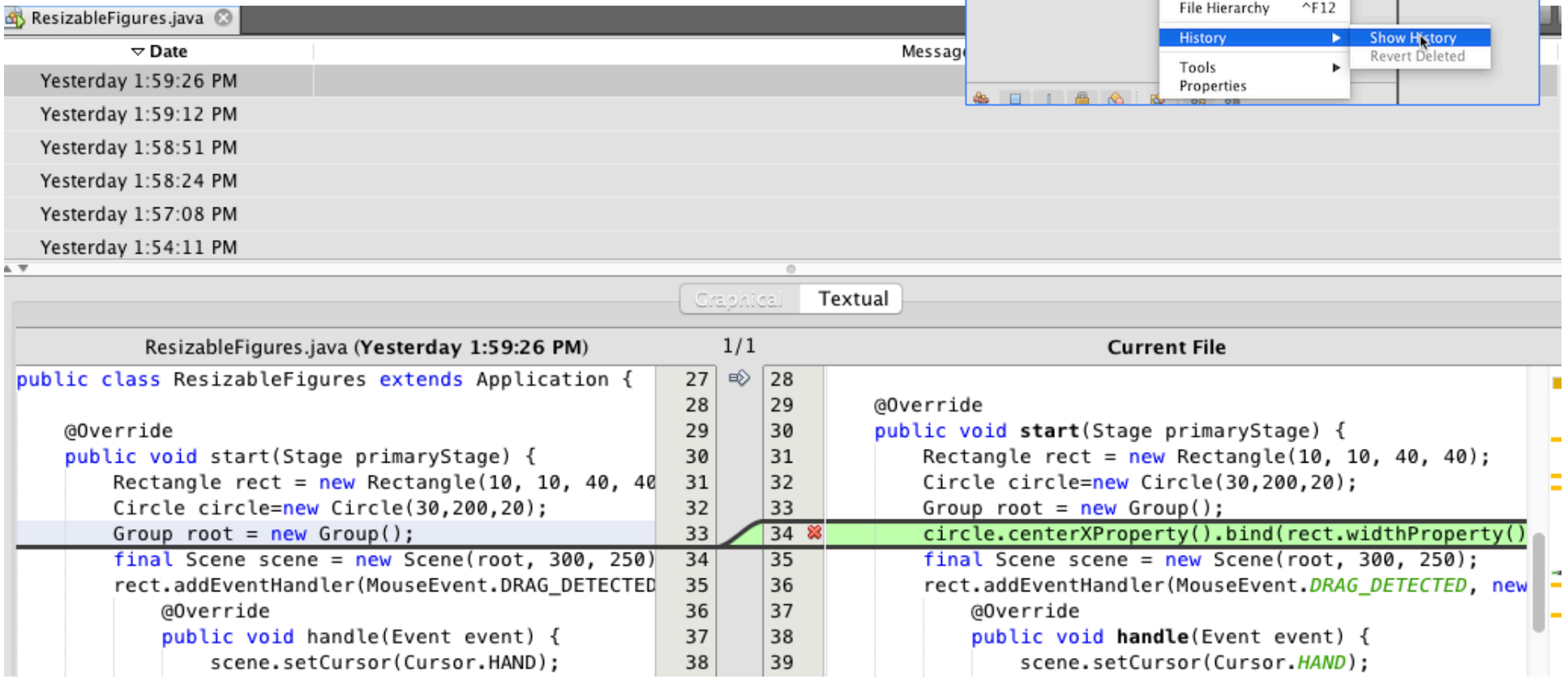


package names!

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

History

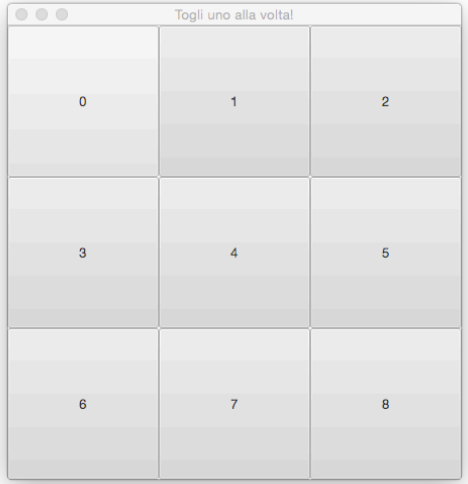

ResizableFigures
Source Packages
resizablefigures
ResizableFigures.java
ResizableRect.java
Scacchiera.java
Test Packages
Libraries
Test Libraries
TestLambdas
Tree
Navigator
Members
ResizableFigures :: Applicat
main(String[] args)
start(Stage primaryStage
Open
Cut ⌘X
Copy ⌘C
Paste ⌘V
Compile File F9
Run File ⇧F6
Debug File ⇧⌘F5
Profile File
Test File ⌘F6
Debug Test File ⇧⌘F6
Profile Test File
Add
Delete
Save As Template...
Find Usages ^F7
Refactor
BeanInfo Editor...
File Members ⌘F12
File Hierarchy ^F12
History
Tools
Properties
Show History
Revert Deleted
Output


ResizableFigures.java
Date
Message
Yesterday 1:59:26 PM
Yesterday 1:59:12 PM
Yesterday 1:58:51 PM
Yesterday 1:58:24 PM
Yesterday 1:57:08 PM
Yesterday 1:54:11 PM
Graphical
Textual
ResizableFigures.java (Yesterday 1:59:26 PM)
1/1
Current File
public class ResizableFigures extends Application {
@Override
public void start(Stage primaryStage) {
Rectangle rect = new Rectangle(10, 10, 40, 40);
Circle circle=new Circle(30,200,20);
Group root = new Group();
final Scene scene = new Scene(root, 300, 250);
rect.addEventHandler(MouseEvent.DRAG_DETECTED
@Override
public void handle(Event event) {
scene.setCursor(Cursor.HAND);
@Override
public void start(Stage primaryStage) {
Rectangle rect = new Rectangle(10, 10, 40, 40);
Circle circle=new Circle(30,200,20);
Group root = new Group();
circle.centerXProperty().bind(rect.widthProperty()
final Scene scene = new Scene(root, 300, 250);
rect.addEventHandler(MouseEvent.DRAG_DETECTED, new
@Override
public void handle(Event event) {
scene.setCursor(Cursor.HAND);

Esercizio

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 SCENA
       // MOSTRA STAGE
    }
     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_TYPED, new
        EventHandler<KeyEvent>() {
            public void handle(KeyEvent keyEvent) {
                String key = keyEvent.getCharacter();
                int i = Integer.parseInt(key);
                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_TYPED, new
        EventHandler<KeyEvent>() {
            public void handle(KeyEvent keyEvent) {
               String key = keyEvent.getCharacter();
               // ATTENTIONPOINT 1
               if (!("012345678".contains(key))) {
                        keyEvent.consume();
                       return;
               }
               int i = Integer.parseInt(key);
               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_TYPED, new
        EventHandler<KeyEvent>() {
            public void handle(KeyEvent keyEvent) {
               String key = keyEvent.getCharacter();
               // ATTENTIONPOINT 1
               if (!("012345678".contains(key))) {
                        keyEvent.consume();
                       return;
               }
               int i = Integer.parseInt(key);
               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_TYPED, new
        EventHandler<KeyEvent>() {
            public void handle(KeyEvent keyEvent) {
               String key = keyEvent.getCharacter();
               // ATTENTIONPOINT 1
               if (!("012345678".contains(key))) {
                        keyEvent.consume();
                       return;
               }
               int i = Integer.parseInt(key);
               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_TYPED, new
        EventHandler<KeyEvent>() {
            public void handle(KeyEvent keyEvent) {
                String key = keyEvent.getCharacter();
                // ATTENTIONPOINT 1
                if (!("012345678".contains(key))) {
                        keyEvent.consume();
                       return;
                }
                int i = Integer.parseInt(key);
                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: componenti personalizzate

```
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_TYPED, new
                      EventHandler<KeyEvent>() {
                public void handle(KeyEvent keyEvent) {
                    String key=keyEvent.getCharacter();
                    if (!("012345678".contains(key))) {
                        keyEvent.consume();
                        return;
                    }
                    int i = Integer.parseInt(key);
                    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 …
```

Reimplementiamo MyPane

```
class MyPane extends GridPane { final int NCOL = 3; final int NROW = 3;
        MyPane() {
            ColumnConstraints cc1 = new ColumnConstraints();
            cc1.setPercentWidth(100./ NCOL);
            getColumnConstraints().addAll(cc1, cc1, cc1);
            RowConstraints rc1 = new RowConstraints();
            rc1.setPercentHeight(100./ NROW);
            getRowConstraints().addAll(rc1, rc1, rc1);
            addEventHandler(KeyEvent.KEY_TYPED, new
                    EventHandler<KeyEvent>() {
                public void handle(KeyEvent keyEvent) {
                    String key = keyEvent.getCharacter();
                    if (!("012345678".contains(key))) {
                        keyEvent.consume(); return;
                    }
                    int i = Integer.parseInt(key);
                    System.out.println(i);
                    boolean retval = getChildren().remove(btn[i]);
                }
            });
        }
```

Reimplementiamo MyPane

```
/**
 * aggiunge un elemento alla griglia mappando l'indice sequenziale
 * in posizione (riga, colonna)
 *
 * @param index indice dell'elemento
 * @param o nodo da aggiungere
 */
 void add(int index, Node o) {
     add(o, index % NCOL, index / NCOL);
 }

 void addAll(Node[] n) {
       for (int i=0; i<n.length; i++) {
          add(i,n[i]);
       }
 }
```

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.

Esempi di esame

Vedere i files allegati alla lezione odierna

Esercizio per lunedi

Pensateci, poi in laboratorio
discuterete l'impianto della soluzione

e potrete fare domande

Slot Machine

1) Scrivere un'applicazione che implementi una slot machine. Tutto il codice deve essere documentato con Javadoc.
L'applicazione presenterà una finestra che ricordi vagamente la seguente immagine

2) I contatori sono due:

Credito (indica i soldi disponibili, espressi in centesimi, inizialmente è 0)

Punteggio (inizialmente è 0)

3) Le monete inizialmente sono 3. Sono dei cerchi su ciascuno dei quali è riportata la dicitura “1 Euro”.

4) Cliccando su una moneta, questa sparisce e il credito viene aumentato di 100.

5) I bottoni sono :

Nuova partita

Spin (disabilitato se il credito è zero)

Pay (disabilitato se il credito è zero)

6) Le ruote dei simboli sono tre, uguali tra loro. Ciascuna contiene gli stessi sei simboli (delle figure geometriche stilizzate: barra inclinata a destra, rombo, cerchio, ecc., scegliete voi). Ogni ruota mostra un solo simbolo alla volta.

7) Cliccando sul tasto “Nuova partita”, se il credito è inferiore a 100 appare una finestra di pop-up che dice “non hai credito sufficiente”. Altrimenti il credito viene diminuito di 100 e il punteggio viene settato a 128.

8) Se Il tasto Spin è abilitato, cliccandolo i simboli delle tre ruote vengono scelti in modo casuale. Ad ogni pressione del tasto “Spin” il punteggio viene dimezzato (ma se è 1 diventa 0).

9) Cliccando su una delle ruote dei simboli, il suo simbolo viene modificato (ma solo se il punteggio non è zero) scegliendolo in modo casuale (quelli delle altre ruote restano immutati). Il punteggio viene dimezzato.

10) Se i simboli mostrati dalle tre ruote sono uguali, appare una finestra di pop-up che dice “Hai vinto”, il credito viene incrementato di un valore pari al punteggio moltiplicato per 100, il punteggio diventa zero.

11) Cliccando sul tasto “Pay” appare un pop-up che dice “Hai vinto XX Euro”, dove XX è il credito diviso 100. Il sistema viene resettato nella condizione iniziale.