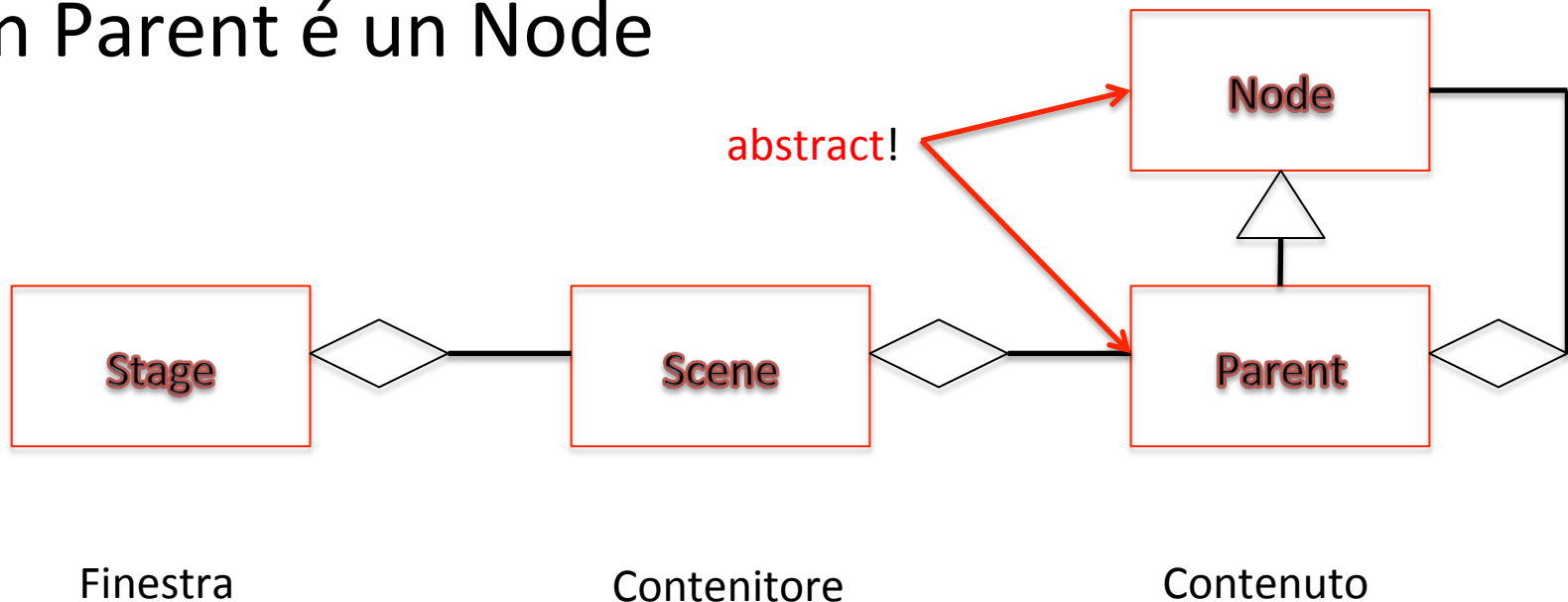


Grafica e non solo: Java FX

Stage/Scene/Parent/Node

Finestra == Stage

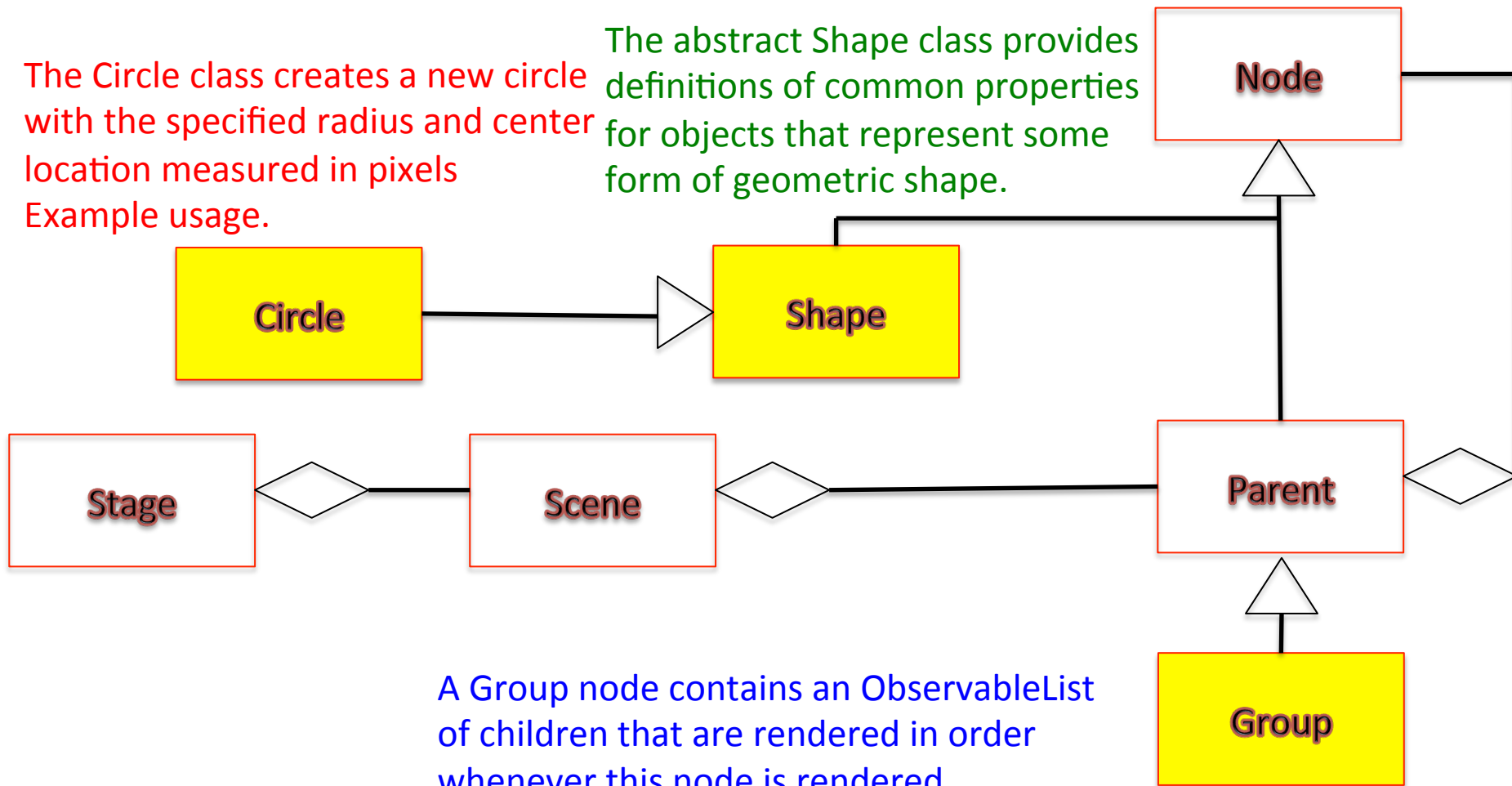
- Uno Stage contiene una Scene
- Una Scene ha un Parent
- Un Parent é un Node



Group – Shape - Circle

The Circle class creates a new circle with the specified radius and center location measured in pixels
Example usage.

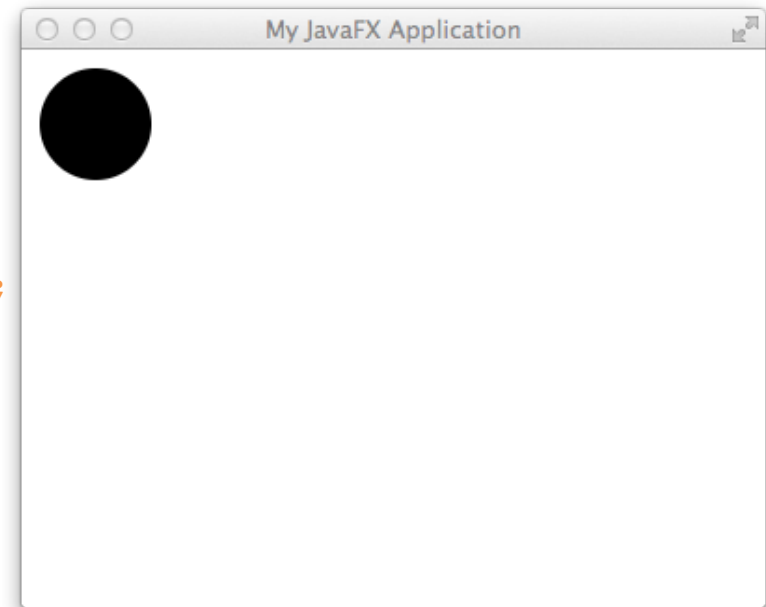
The abstract Shape class provides definitions of common properties for objects that represent some form of geometric shape.



A Group node contains an ObservableList of children that are rendered in order whenever this node is rendered.

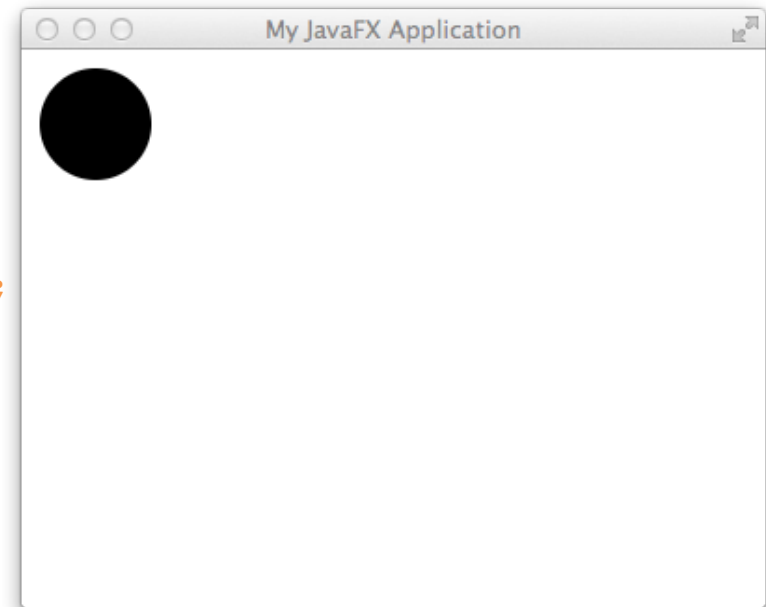
Applicazione minima

```
package it.unitn.disi.javafxapplication;
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.shape.Circle;
import javafx.stage.Stage;
public class MinimalApp extends Application {
    public void start(Stage stage) {
        Node circ = new Circle(40, 40, 30);
        Parent root = new Group(circ);
        Scene scene = new Scene(root, 400, 300);
        stage.setTitle("My JavaFX Application");
        stage.setScene(scene);
        stage.show();
    }
    public static void main(String[] args) {
        Application.launch(args);
    } }
```

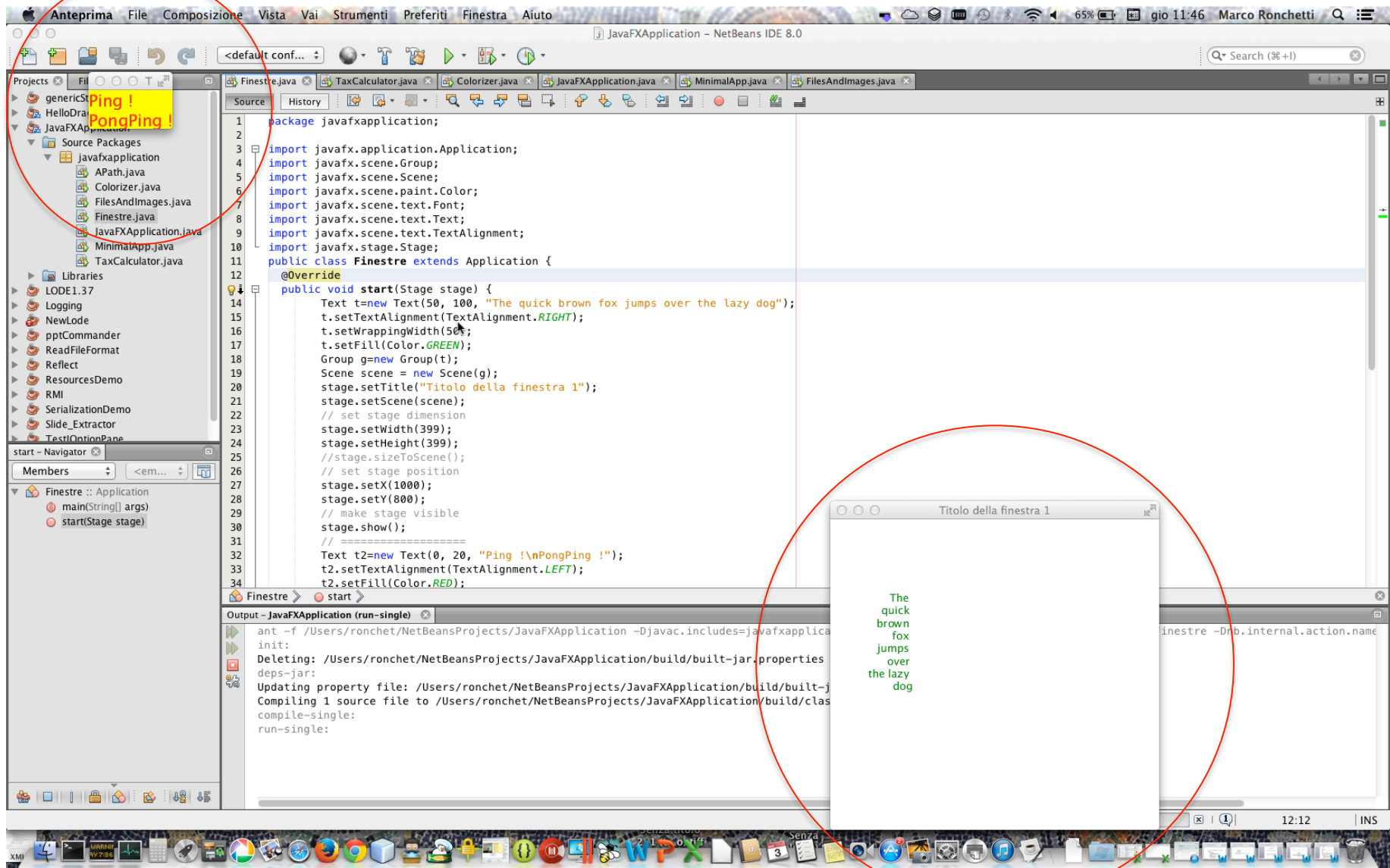


Applicazione minima

```
package it.unitn.disi.javafxapplication;
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.shape.Circle;
import javafx.stage.Stage;
public class MinimalApp extends Application {
    public void start(Stage stage) {
        Circle circ = new Circle(40, 40, 30);
        Group root = new Group(circ);
        Scene scene = new Scene(root, 400, 300);
        stage.setTitle("My JavaFX Application");
        stage.setScene(scene);
        stage.show();
    }
    public static void main(String[] args) {
        Application.launch(args);
    } }
```

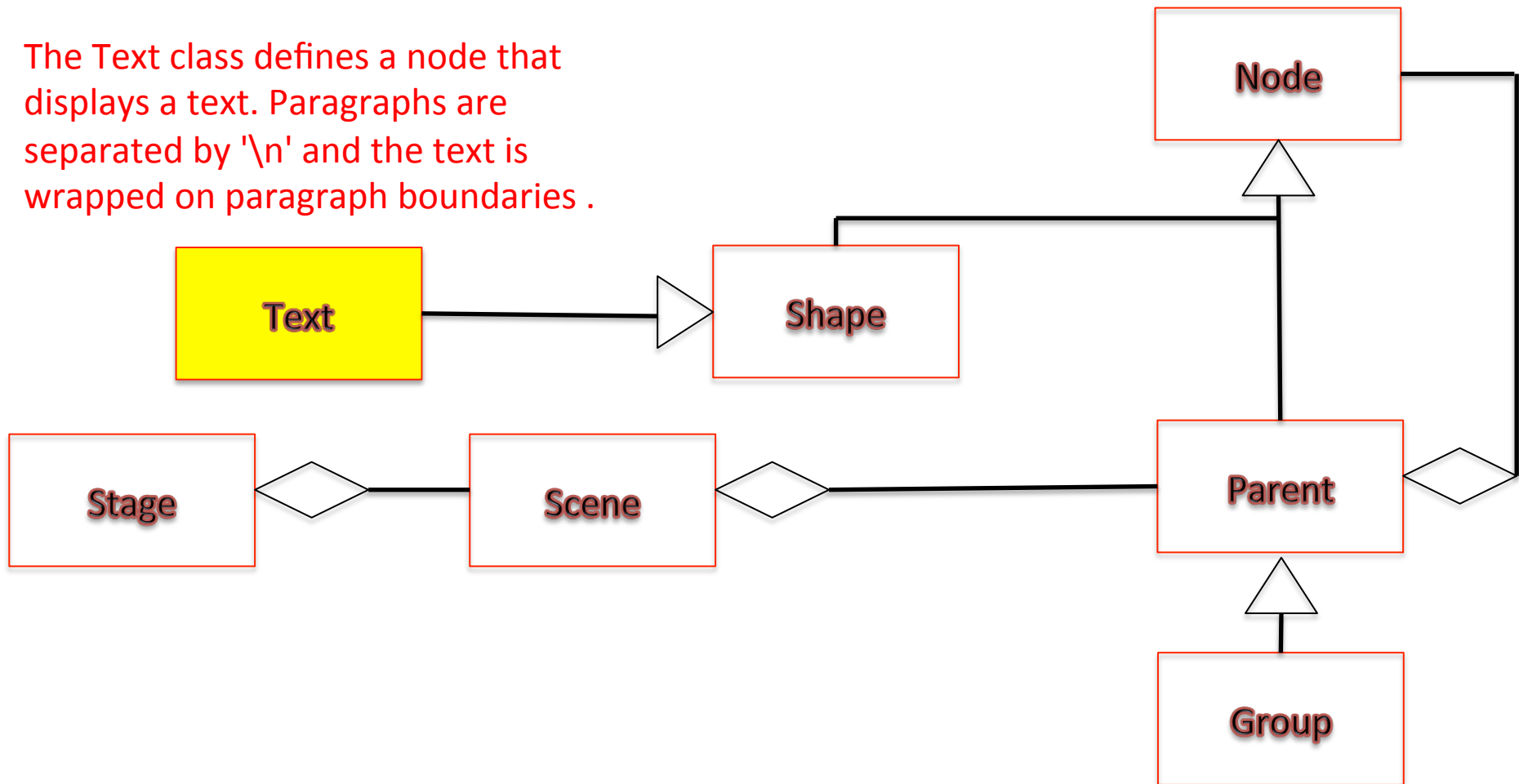


Finestre multiple



Group – Shape - Circle

The Text class defines a node that displays a text. Paragraphs are separated by '\n' and the text is wrapped on paragraph boundaries .



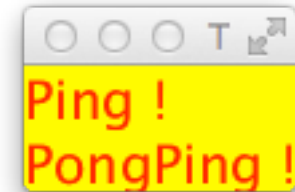
Finestre multiple: Prima finestra

```
public class Finestre extends Application {  
    public void start(Stage stage) {  
        Text t=new Text(50, 100, "The quick brown fox jumps over  
            the lazy dog");  
        t.setTextAlignment(TextAlignment.RIGHT);  
        t.setWrappingWidth(50);  
        t.setFill(Color.GREEN);  
        Group g=new Group(t);  
        Scene scene = new Scene(g);  
        stage.setTitle("Titolo della finestra 1");  
        stage.setScene(scene);  
        // set stage dimension  
        stage.setWidth(399);  
        stage.setHeight(399);  
        // set stage position  
        stage.setX(1000);  
        stage.setY(800);  
        // make stage visible  
        stage.show();  
    }  
}
```



Finestre multiple: Seconda finestra

```
Text t2=new Text(0, 20, "Ping !\nPongPing !");
t2.setTextAlignment(TextAlignment.LEFT);
t2.setFill(Color.RED);
t2.setFont(new Font(20));
Group g2=new Group(t2);
Scene scene2 = new Scene(g2);
scene2.setFill(Color.YELLOW);
Stage stage2=new Stage();
stage2.setTitle("Titolo della finestra 2");
stage2.setScene(scene2);
stage2.setX(100);
stage2.setY(80);
stage2.sizeToScene();
stage2.show();
}
public static void main(String[] args) {
    launch(args);
} }
```



Terminazione

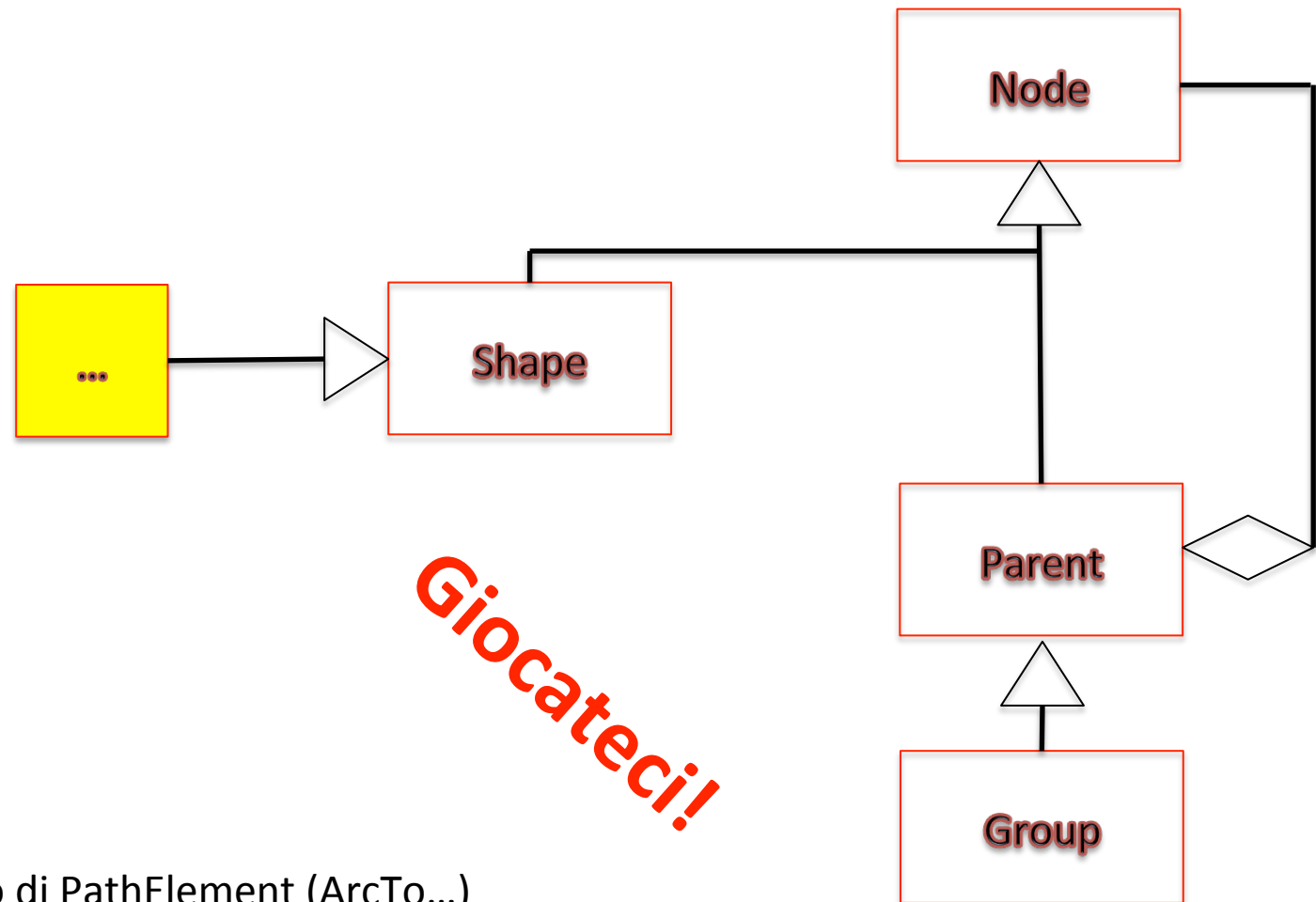
- Quando termina il processo?

(Un Processo é un Programma in esecuzione)

Shape hierarchy

Shape

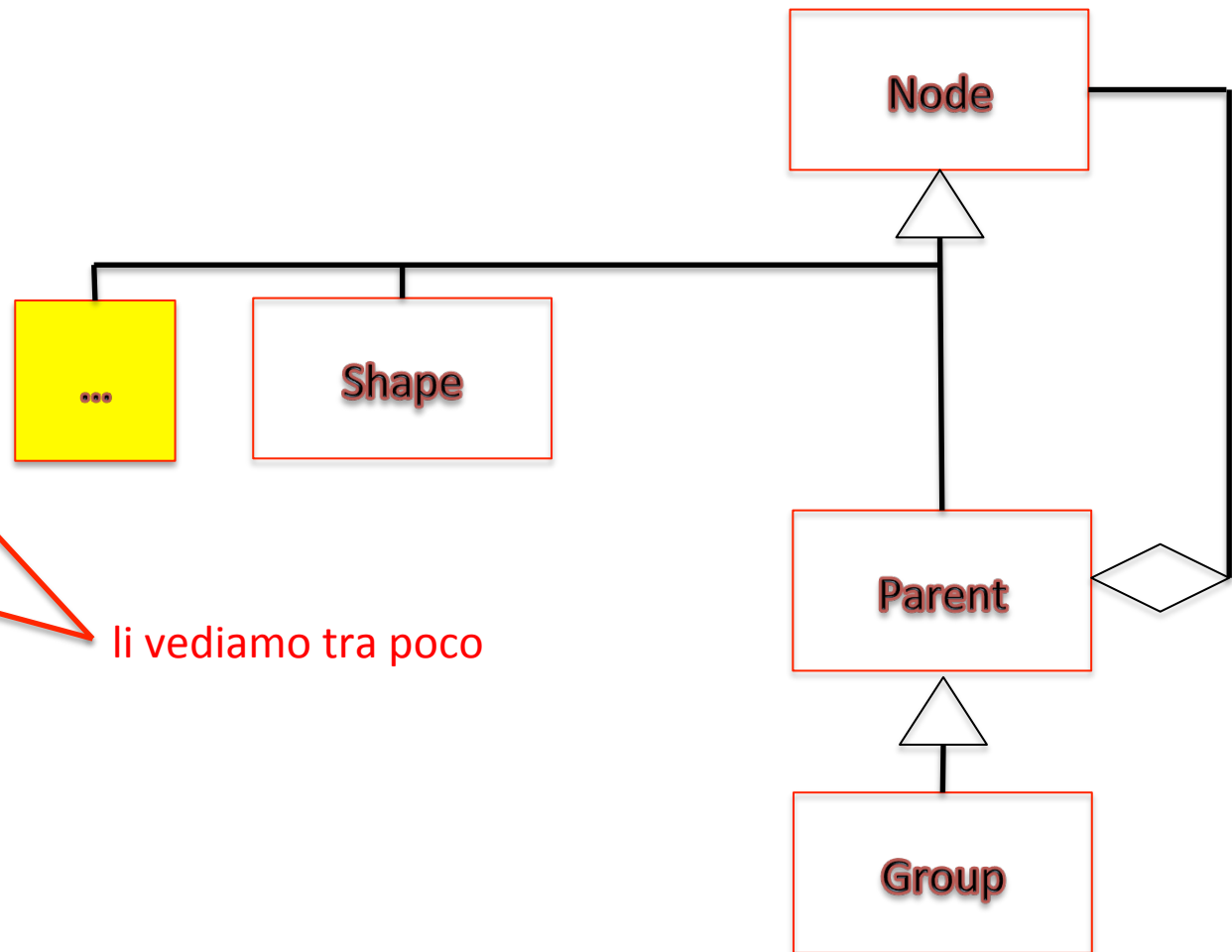
- Line
- Polyline
- Polygon
- Rectangle
- Arc
- Circle
- Ellipse
- QuadCurve
- CubicCurve
- Text
- SVGPath
- Path composto di PathElement (ArcTo...)



Node hierarchy

Node

- Parent
- Shape
- **ImageView**
- **MediaView**
- Canvas



Canvas

Una "tela del pittore" con un metodo per ottenere il suo **GraphicsContext** che ha varie primitive per disegnarci sopra:

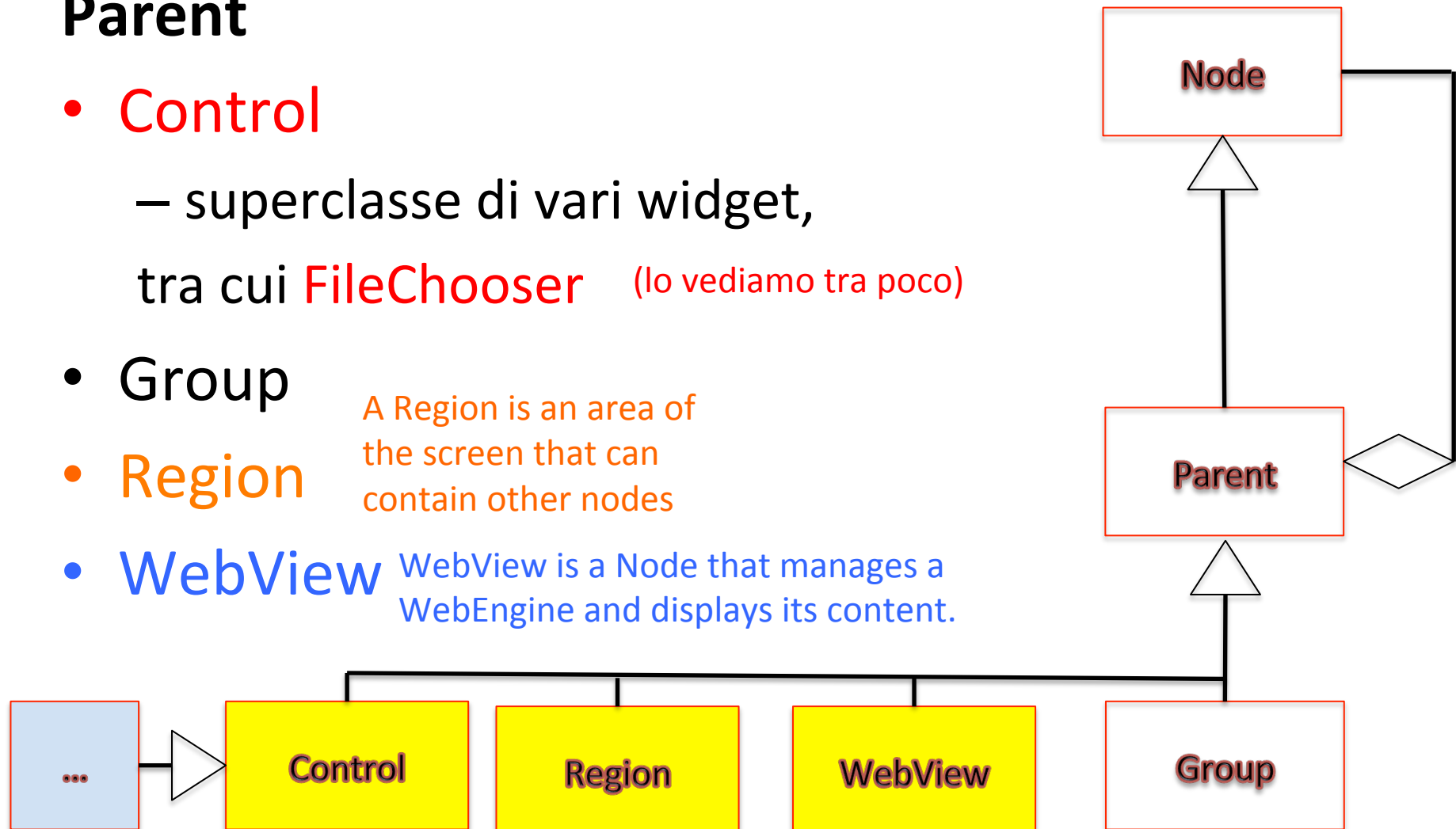
- fillArc()
- fillRect()
- drawImage()
- ...

<http://docs.oracle.com/javafx/2/canvas/jfxpub-canvas.htm>

Parent hierarchy

Parent

- **Control**
 - superclasse di vari widget,
tra cui **FileChooser** (lo vediamo tra poco)
- **Group**
- **Region** A Region is an area of the screen that can contain other nodes
- **WebView** WebView is a Node that manages a WebEngine and displays its content.



JavaFX UI Controls



Controls

Button:



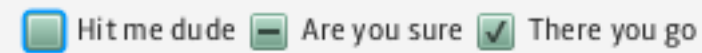
Toggle Button:



Hyperlink:

Hello I am a hyperlink  I can have an icon too

CheckBox:
Normal/Undefined/Selected



Radio Button:



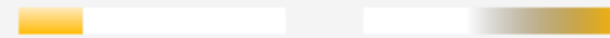
Menu Buttons:



ScrollView:



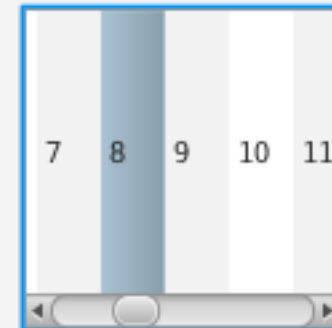
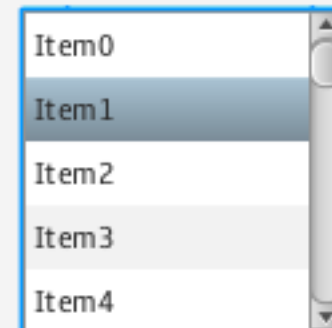
ProgressBar:



ProgressIndicator
Indeterminate:



ListView:



ImageView - Image

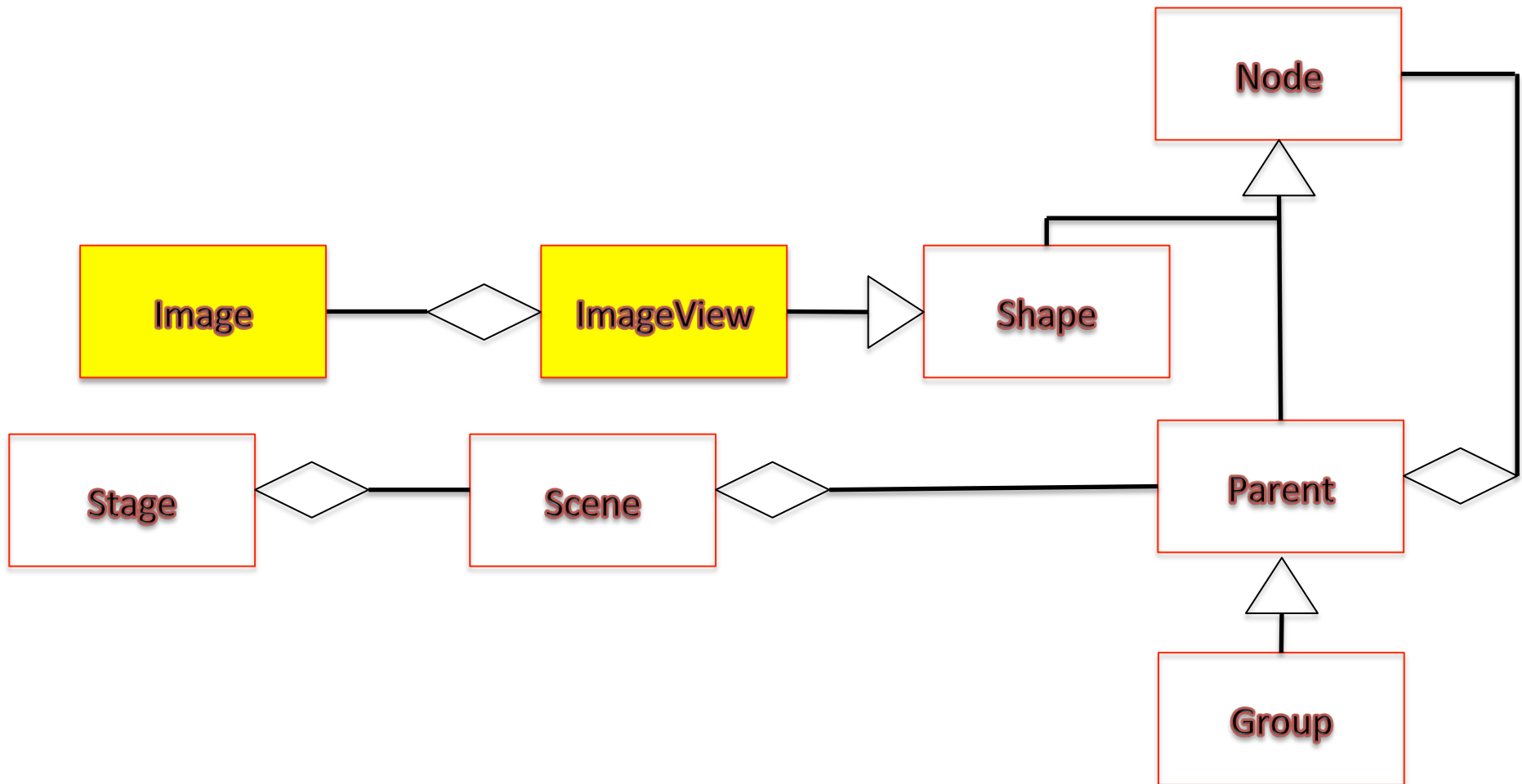
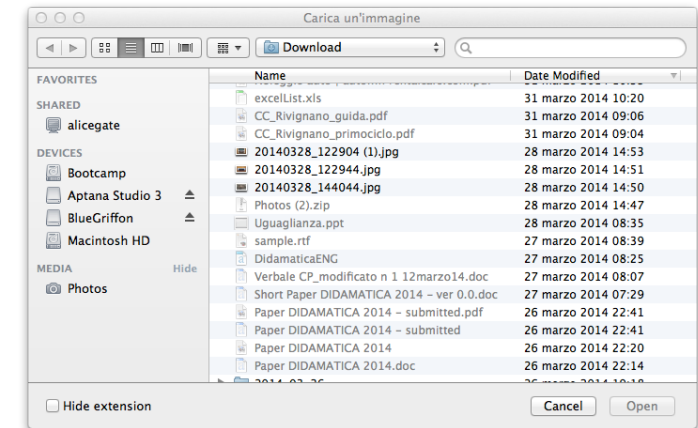


Image and File



```
public class FilesAndImages extends Application {
    public void start(Stage stage) {
        FileChooser fileChooser = new FileChooser();
        fileChooser.setTitle("Carica un'immagine");
        fileChooser.getExtensionFilters().addAll(
            new FileChooser.ExtensionFilter("JPG", "*.jpg"),
            new FileChooser.ExtensionFilter("PNG", "*.png")
        );
        String url = System.getProperty("user.home");
        File f=new File(url);
        fileChooser.setInitialDirectory(f); // bugged on MacOSX
        File file = fileChooser.showOpenDialog(stage);
        if (file == null) {
            System.out.println("No file chosen");
            System.exit(1);
        }
    }
}
```

Image and File

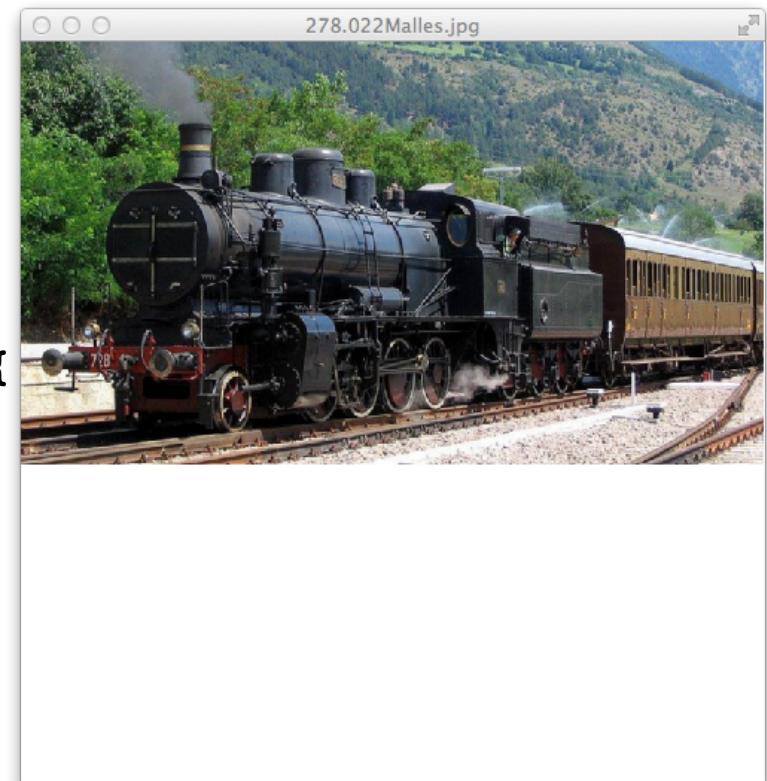
```
Image image = new Image("file://" +  
    file.getAbsolutePath(), 500, 500, true, true);  
ImageView iw = new ImageView(image);  
Group root = new Group(iw);  
Scene scene = new Scene(root, 500,500);  
stage.setTitle(file.getName());  
stage.setScene(scene);  
stage.sizeToScene();  
stage.show();
```

```
}
```

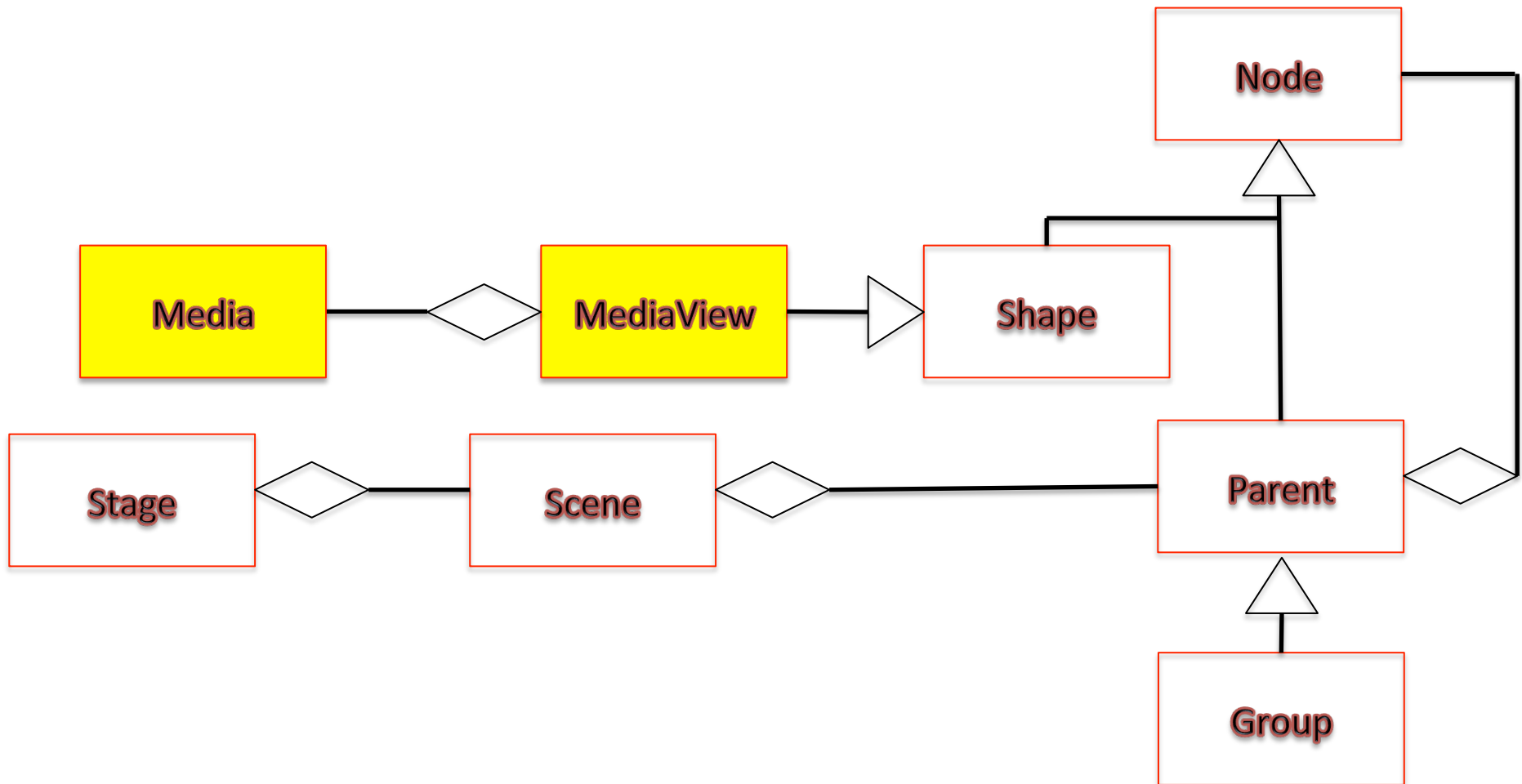
```
public static void main(String[] args) {  
    Application.launch(args);
```

```
}
```

```
}
```

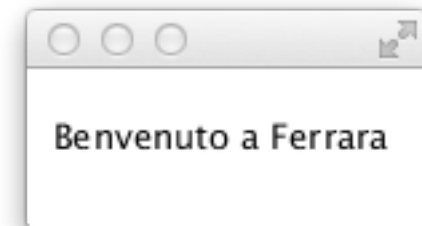


MediaView - Media



MediaView

```
public class Sounds extends Application{  
    public void start(Stage stage) {  
        Media media = new Media("http://www.ferraraterraeacqua.it/  
it/audioguide/audioguide-di-ferrara-citta-del-rinascimento/  
01_benvenuto-a-ferrara.mp3");  
        MediaPlayer mediaPlayer = new MediaPlayer(media);  
        mediaPlayer.setAutoplay(true);  
        // create mediaView and add media player to the viewer  
        MediaView mediaView = new MediaView(mediaPlayer);  
        Group root = new Group(mediaView);  
        root.getChildren().add(  
            new Text(10, 30, "Benvenuto a Ferrara"));  
        Scene scene = new Scene(root, 150, 60);  
        stage.setScene(scene);  
        stage.sizeToScene();  
        stage.show();  
    }  
    public static void main(String[] args) {  
        Application.launch(args);  
    }  
}
```



<http://docs.oracle.com/javafx/2/media/overview.htm>

Gestione di base degli eventi

Basic Events

```
public class Event0 extends Application {  
    public void start(Stage stage) {  
        Button btn = new Button();  
        btn.setText("Click me");  
        Listener a=new Listener();  
        btn.addEventHandler(Event.ANY, a);  
        Group root = new Group(btn);  
        Scene scene = new Scene(root, 300, 250);  
        stage.setScene(scene);  
        stage.sizeToScene();  
        stage.show();    }  
    public static void main(String[] args){  
        Application.launch(args); }  
}  
class Listener implements EventHandler{  
    int counter=0;  
    public void handle(Event t) {  
        System.out.println(++counter+" Ricevuto un evento di tipo "  
            +t.getEventType()); } }
```

1 Ricevuto un evento di tipo
INPUT_METHOD_TEXT_CHANGED
2 Ricevuto un evento di tipo MOUSE_ENTERED
3 Ricevuto un evento di tipo
MOUSE_ENTERED_TARGET
4 Ricevuto un evento di tipo MOUSE_MOVED
...
12 Ricevuto un evento di tipo MOUSE_MOVED
13 Ricevuto un evento di tipo MOUSE_PRESSED
14 Ricevuto un evento di tipo ACTION
15 Ricevuto un evento di tipo MOUSE_RELEASED
16 Ricevuto un evento di tipo MOUSE_CLICKED
17 Ricevuto un evento di tipo MOUSE_MOVED



Basic Events

```
public class Event0 extends Application {  
    public void start(Stage stage) {  
        Button btn = new Button();  
        btn.setText("Click me");  
        Listener a=new Listener();  
        btn.addEventHandler(ActionEvent.ACTION, a);  
        Group root = new Group(btn);  
        Scene scene = new Scene(root, 300, 250);  
        stage.setScene(scene);  
        stage.sizeToScene();  
        stage.show();    }  
    public static void main(String[] args){  
        Application.launch(args); }  
}  
class Listener implements EventHandler{  
    int counter=0;  
    public void handle(Event t) {  
        System.out.println(++counter+" Ricevuto un evento di tipo "  
            +t.getEventType()); } }
```

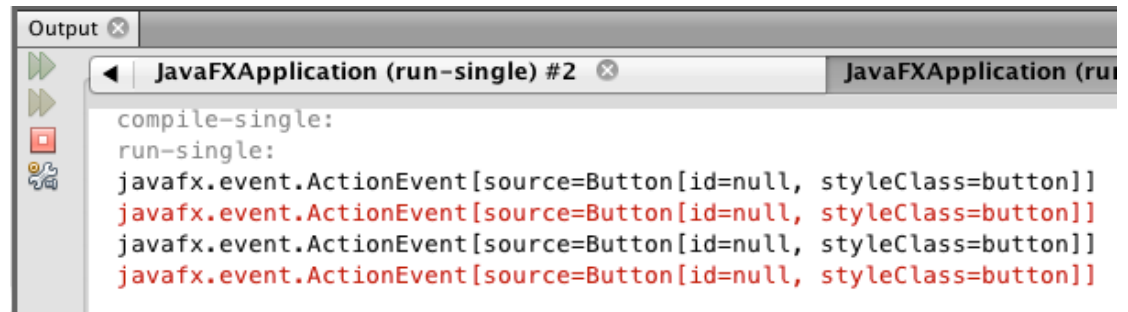


MultiListener

```
public class Event0 extends Application {  
    public void start(Stage stage) {  
        Button btn = new Button();  
        btn.setText("Click me");  
        Olistener o=new OListener();  
        Elistener e=new EListener();  
        btn.addEventHandler(ActionEvent.ACTION, o);  
        btn.addEventHandler(ActionEvent.ACTION, e);  
        Group root = new Group(btn);  
        Scene scene = new Scene(root, 300, 250);  
        stage (scene);  
        stage ();  
    }  
    public static void main(String[] args){  
        Application.launch(args);  
    }  
}
```

```
class OListener  
    implements EventHandler{  
        public void handle(Event t) {  
            System.out.println(t); }  
    }
```

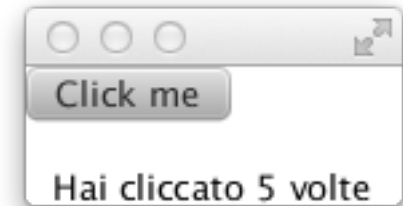
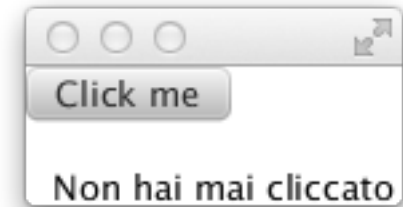
```
class EListener  
    implements EventHandler{  
        public void handle(Event t) {  
            System.err.println(t); }  
    }
```



Come organizzare gli
ascoltatori/osservatori

Listener Esterno

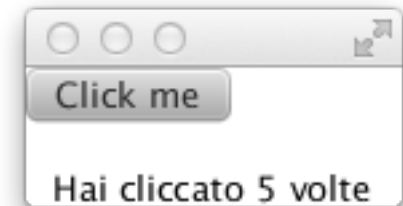
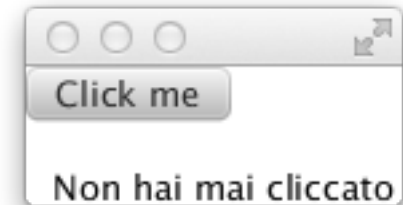
```
public class AppWithEvents1 extends Application {
    Text text=null;
    public void start(Stage stage) {
        text=new Text(10,50,"Non hai mai cliccato ");
        Button btn = new Button();
        btn.setText("Click me");
        Listener a=new Listener(this);
        btn.addEventHandler(ActionEvent.ACTION, a);
        Group root = new Group(btn);
        root.getChildren().add(text);
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }
    public void updateText(int n){
        text.setText("Hai cliccato "+n
            +" volte");
    }
    public static void main(String[] args) {
        Application.launch(args);
    }
}
```



```
class Listener
    implements EventHandler{
    AppWithEvents1 awe=null;
    int counter=0;
    Listener1(AppWithEvents1 a){
        awe=a;
    }
    public void handle(Event t) {
        awe.updateText(++counter);
    }
}
```

Listener Interno

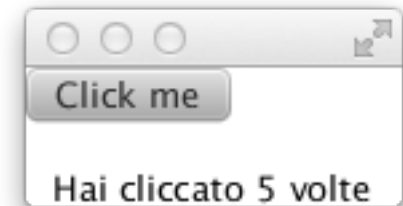
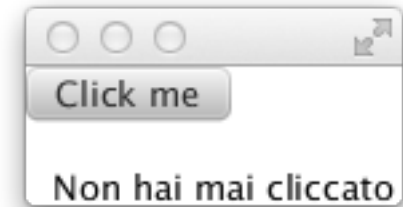
```
public class AppWithEvents1 extends Application {
    Text text=null;
    public void start(Stage stage) {
        text=new Text(10,50,"Non hai mai cliccato ");
        Button btn = new Button();
        btn.setText("Click me");
        Listener a=new Listener(this);
        btn.addEventHandler(ActionEvent.ACTION, a);
        Group root = new Group(btn);
        root.getChildren().add(text);
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }
    public void updateText(int n){
        text.setText("Hai cliccato "+n
            +" volte");
    }
    public static void main(String[] args) {
        Application.launch(args);
    }
}
```



```
class Listener
    implements EventHandler{
AppWithEvents1 awe=null;
    int counter=0;
Listener1(AppWithEvents1 a){
    awe=a;
}
    public void handle(Event t) {
        awe.updateText(++counter);
    }
}
```

Listener Interno

```
public class AppWithEvents1 extends Application {
    Text text=null;
    public void start(Stage stage) {
        text=new Text(10,50,"Non hai mai cliccato ");
        Button btn = new Button();
        btn.setText("Click me");
        Listener1 a=new Listener();
        btn.addEventHandler(ActionEvent.ACTION, a);
        Group root = new Group(btn);
        root.getChildren().add(text);
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }
    class Listener
        implements EventHandler{
            int counter=0;
            public void handle(Event t) {
                updateText(++counter);
            }
        }
}
```



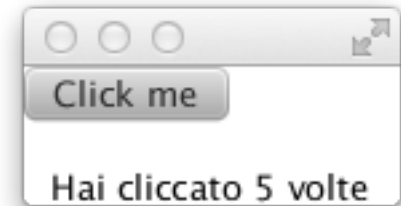
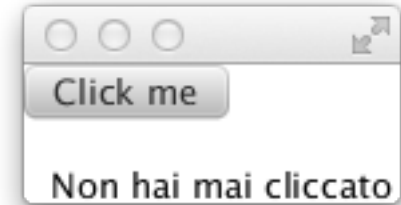
```
public void updateText(int n){
    text.setText("Hai cliccato"
        +n+" volte");
}
public static void main(
    String[] args) {
    Application.launch(args);
}
```

Listener Interno Anonimo

```
public class AppWithEvents1 extends Application {
    Text text=null;
    public void start(Stage stage) {
        text=new Text(10,50,"Non hai mai cliccato ");
        Button btn = new Button();
        btn.setText("Click me");
        Listener1 a=new EventHandler(){
            int counter=0;
            public void handle(Event t) {
                updateText(++counter);
            };
        };
        btn.addEventHandler(ActionEvent.ACTION, a);
        Group root = new Group(btn);
        root.getChildren().add(text);
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }
}

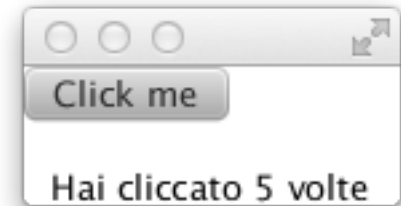
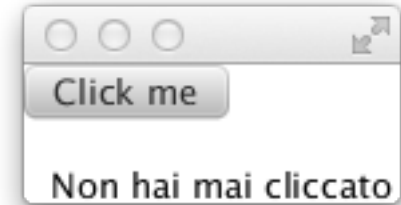
public void updateText(int n){
    text.setText("Hai cliccato"
        +n+" volte");
}

public static void main(
    String[] args) {
    Application.launch(args);
}}
```



Self Listener

```
public class AppWithEvents
    extends Application implements EventHandler {
    Text text=null;
    int counter=0;
    public void start(Stage stage) {
        text=new Text(10,50,"Non hai mai cliccato ");
        Button btn = new Button();
        btn.setText("Click me");
        btn.addEventHandler(ActionEvent.ACTION, this);
        Group root = new Group(btn);
        root.getChildren().add(text);
        Scene scene = new Scene(root);
        stage.setScene(scene);
        stage.show();
    }
    public void handle(Event t) {
        updateText(++counter);
    }
```

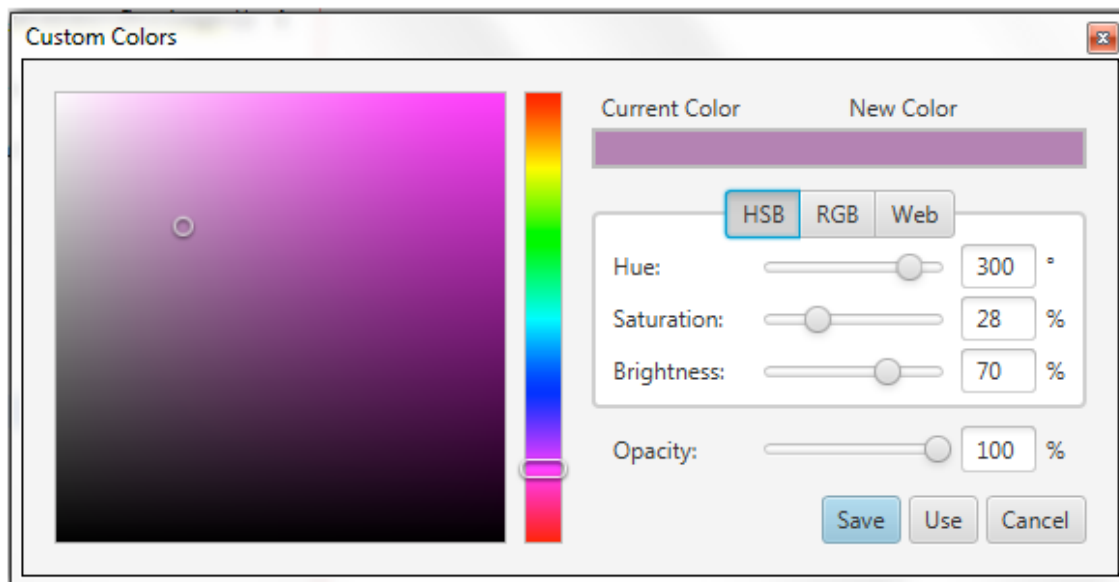
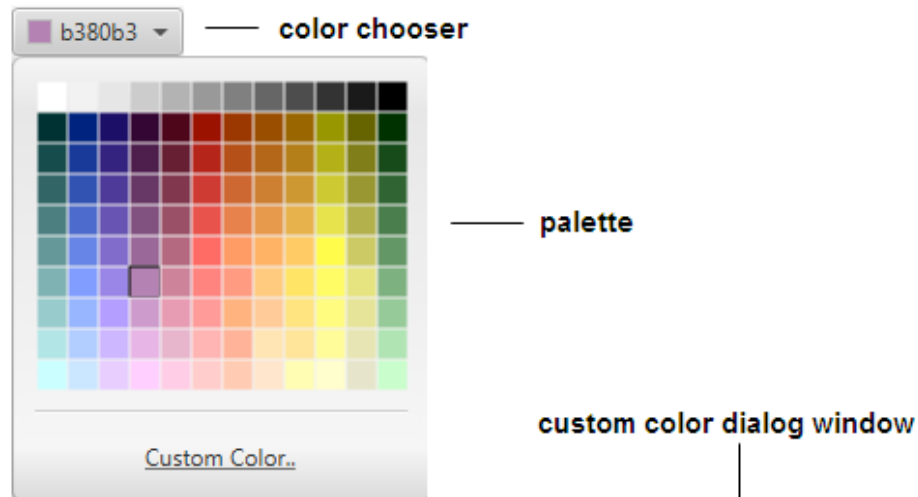


```
        public void updateText(int n){
            text.setText("Hai cliccato"
                +n+" volte");
        }
        public static void main(
            String[] args) {
            Application.launch(args);
        }
    }
```

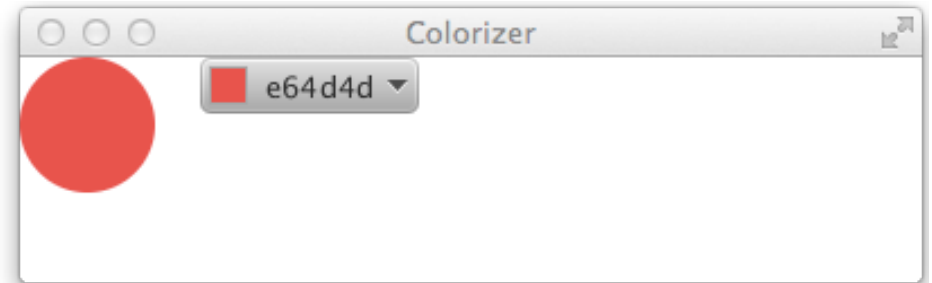
Due questioni:

- ColorPicker
- Convenience Methods

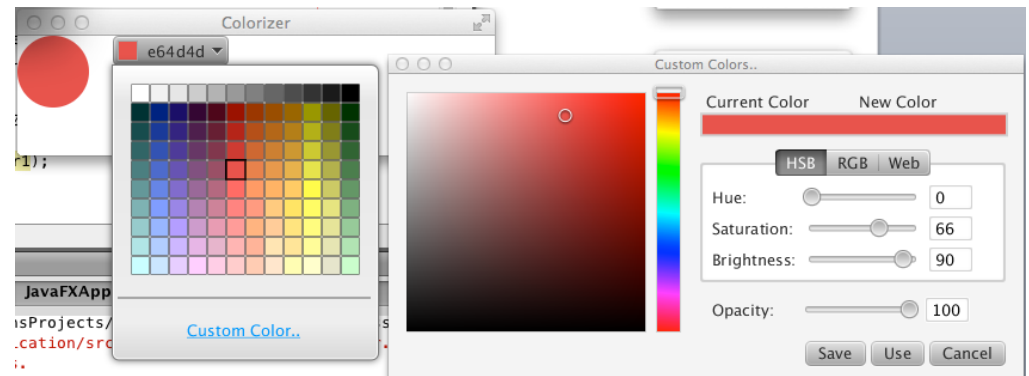
ColorPicker



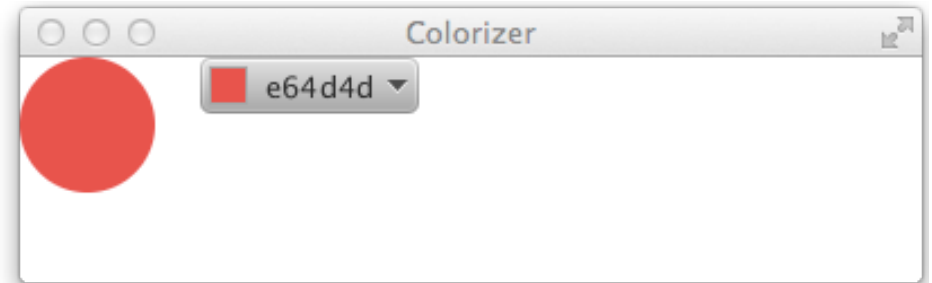
ColorPicker



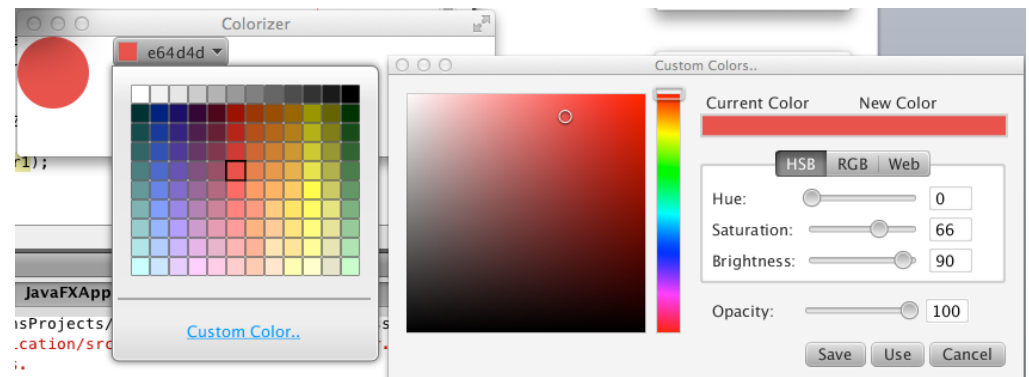
```
public class Colorizer extends Application {  
    public void start(final Stage stage) {  
        final Circle circ = new Circle(40, 40, 30);  
        final ColorPicker colorPicker1 = new ColorPicker(Color.BLACK);  
  
        colorPicker1.addEventHandler(ActionEvent.ACTION, new EventHandler() {  
            @Override  
            public void handle(Event t) {  
                System.out.println(t.getEventType());  
                circ.setFill(colorPicker1.getValue());  
            }  
        });  
        Scene scene = new Scene(new HBox(20), 400, 100);  
        HBox box = (HBox) scene.getRoot();  
        box.getChildren().addAll(circ, colorPicker1);  
        stage.setScene(scene);  
        stage.show();  
    }  
    ...  
}
```



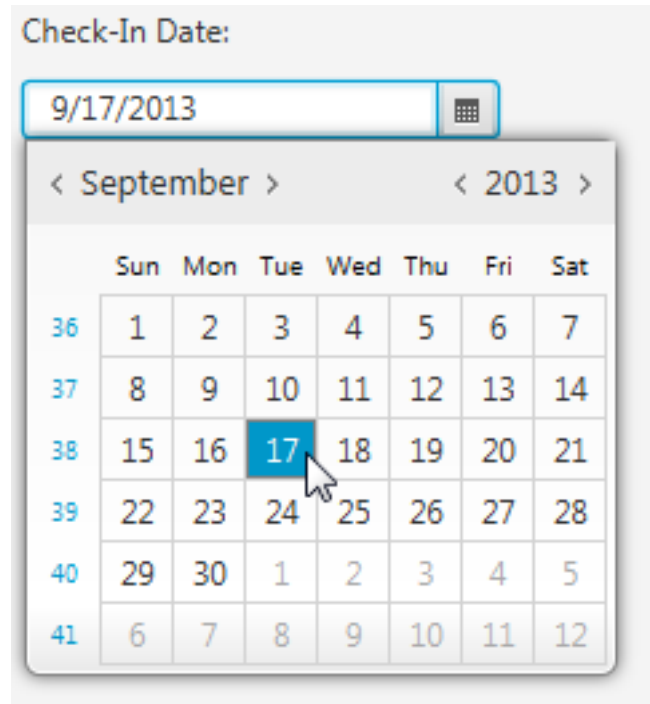
Convenience Methods



```
public class Colorizer extends Application {  
    public void start(final Stage stage) {  
        final Circle circ = new Circle(40, 40, 30);  
        final ColorPicker colorPicker1 = new ColorPicker(Color.BLACK);  
        colorPicker1.setOnAction(new EventHandler() {  
            // colorPicker1.addEventHandler(ActionEvent.ACTION, new EventHandler() {  
            @Override  
            public void handle(Event t) {  
                System.out.println(t.getEventType());  
                circ.setFill(colorPicker1.getValue());  
            }  
        });  
        Scene scene = new Scene(new HBox(20), 400, 100);  
        HBox box = (HBox) scene.getRoot();  
        box.getChildren().addAll(circ, colorPicker1);  
        stage.setScene(scene);  
        stage.show();  
    }  
    ...  
}
```



DatePicker



<http://docs.oracle.com/javase/8/javafx/user-interface-tutorial/date-picker.htm#CCHHJBEA>