SI3 - IHM 2017-2018

JavaFX

Styling and Animation

Université Nice Sophia Antipolis (Polytech)

19 Février 2018

Marco Winckler

Clément Duffau + Anne Marie Pinna-Dery

Université Nice Sophia (Polytech) | I3S | SPARKS team | bureau 446

winckler@i3s.unice.fr

http://www.i3s.unice.fr/~winckler/

Agenda

- Effets visuels avec CSS
- Animation

Effets visuel

GaussianBlur

InnerShadow

Reflection

SepiaTone

Effets visuel

- Drop Shadow – ajoute de l'ombre
- Reflection – ajoute du reflet sous le node
- Lighting – simule un éclairage sur un node pour donner un effet optique 3D

Effet visuel pas à pas

- Dans l'ide: créer un fichier .CSS
- Dans le .Java: associer le CSS au projet, ex.:

```
Scene scene = new Scene(grid, 300, 275);
primaryStage.setScene(scene);
scene.getStylesheets().add
(Login.class.getResource("Login.css").toExternalForm());
primaryStage.show();
OU
Parent root =
FXMLLoader.load(getClass().getResource("sample.fxml"));
root.getStylesheets().add(getClass().getResource("animation.css").toE
xternalForm());
```

Editer le CSS

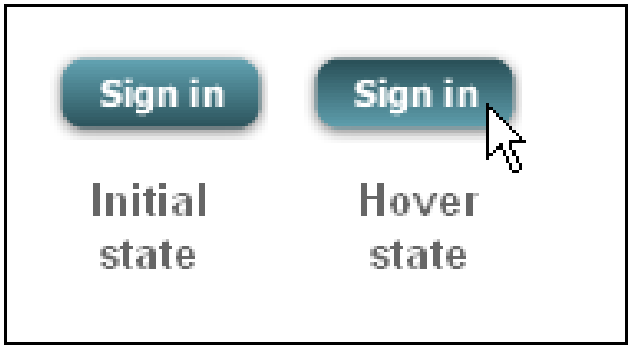
```
.root {
-fx-background-image: url("background.jpg");
}

.label {
-fx-font-size: 12px;
-fx-font-weight: bold;
-fx-text-fill: #333333;
-fx-effect: dropshadow( gaussian , rgba(255,255,255,0.5) , 0,0,0,1
);
}
```

Résultat


JavaFX Welcome!
Welcome
User Name:
Password:
Sign in


JavaFX Welcome
Welcome
User Name:
Password:
Sign in

Autre exemple




```
// CSS Drop Shadow Button
.button {
    -fx-text-fill: white;
    -fx-font-family: "Arial Narrow";
    -fx-font-weight: bold;
    -fx-background-color: linear-gradient(#61a2b1, #2A5058);
    -fx-effect: dropshadow( three-pass-box , rgba(0,0,0,0.6) , 5, 0.0 , 0 , 1 );
}
// CSS Button Hover Style
.button:hover {
    -fx-background-color: linear-gradient(#2A5058, #61a2b1);
}
```

Type de transformation

- Tous les nodes de la scène peuvent être transformé à partir de positions x,y
- Classes javafx.scene.tranform:
  - **translate** – change la position à partir de la position initial
  - **scale** – change la taille en fonction d'un facteur de échelle
  - **shear** – fait de rotation dans l'axe x et/ou y de façon indépendante
  - **rotate** – fait la rotation à partir d'un point central de la scène
  - **affine** – fait un mapping des coordonnées 2-D/3-D

Transformations

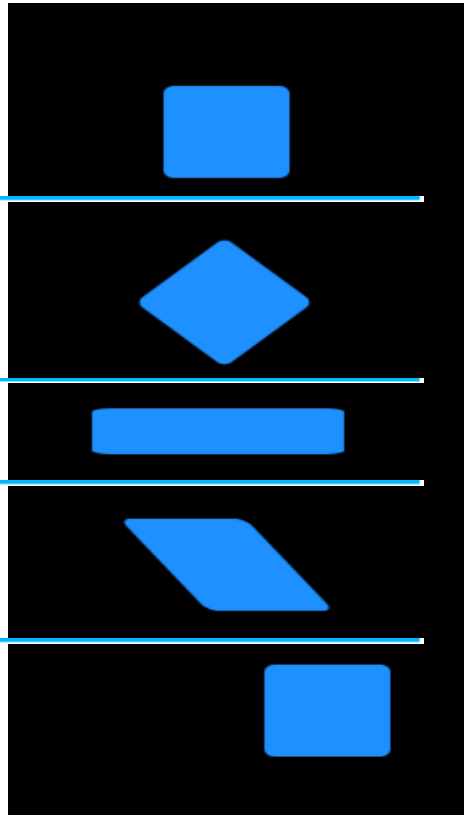
```
Rectangle rect=new Rectangle(0,0,60,60);
rect.setFill(Color.DODGERBLUE);
rect.setArcWidth(10);
rect.setArcHeight(10);
```

```
rect.setRotate(45);
```

```
rect.setScaleX(2);
rect.setScaleY(0.5);
```

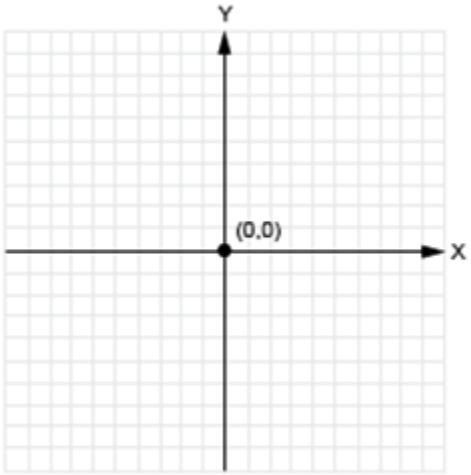
```
Shear shear = new Shear(0.7, 0);
rect.getTransforms().add(shear);
```

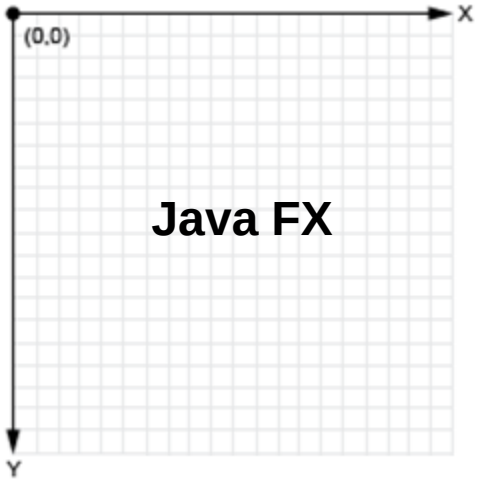
```
rect.setTranslateX(40);
rect.setTranslateY(10);
```



Animations

- Rappel: système de coordonnées (0,0)





Exemple: pas à pas

- Importer les classes animation /Contrôleur

  ```
  import javafx.animation.TranslateTransition;
  ```

- Initialiser l'animation

  ```
  TranslateTransition t = new TranslateTransition();
  ```

- Associer l'animation à l'objet de l'interface

  ```
  t.setNode(button);
  ```

- Définir les paramètre de l'animation

  ```
  t.setToY(-400); // déplacer à la vertical vers le haut
  t.setToX(-300); // déplacer à l'horisontal vers la gauche
  t.setAutoReverse(true); // fait des aller-retour
  t.setCycleCount(2); // nombre de boucles
  ```

- Executer l'animation
  - t.play();

Ajouter des évènements

- Gérer l’événement à la fin de l’animation

```
t.setOnFinished(event ->{
        Alert a = new Alert(Alert.AlertType.INFORMATION);
        a.setHeaderText("c’est fini!");
        a.show();
    } );
```

Exercices

- Identifier les animations à créer dans l'IHM pour:
  - Donner du feedback à l'utilisateur
  - Signaler les information important
  - Signaler les changements d'information
  - Faire de transitions pour les actions
  - Etc
- Implémenter les animations dans le projet