Cours R2.02
Introduction à l'Interaction Humain-Machine

Cours 4 : Widgets et événements (2/2)

Géry Casiez https://gery.casiez.net
Département informatique - IUT de Lille - Université de Lille

Plan du cours en 9 semaines

1. Introduction à l'interaction, placement
2. Programmation événementielle
3. Widgets et événements (1/2)
4. **Widgets et événements (2/2)**
5. Conception et prototypage (1/2)
6. Conception et prototypage (2/2)
7. Heuristiques et recommandations
8. Modèles et théories
9. Méthodes d'évaluation des IHM

Objectifs

Introduction à MVC

Utilisation des ListView

Dessiner avec Canvas

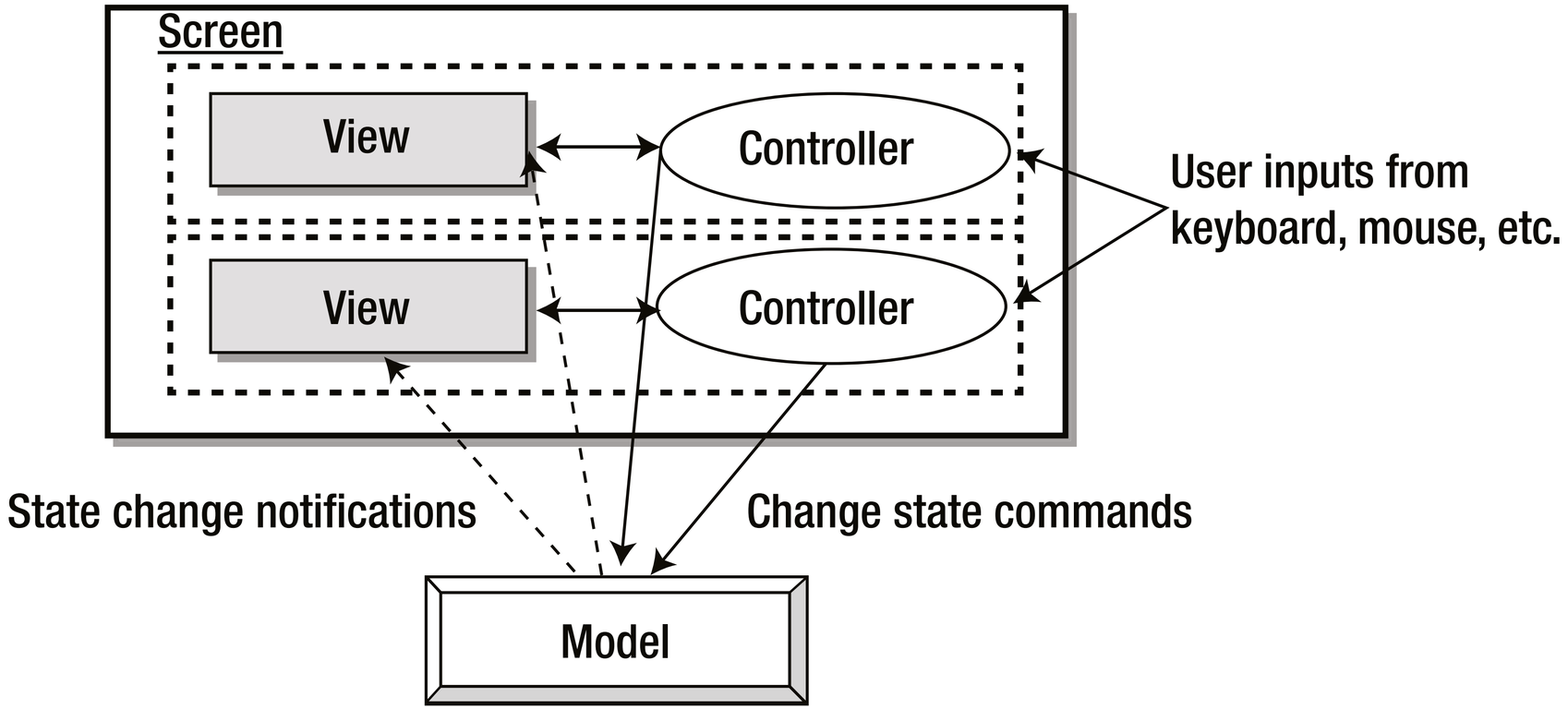
Une brève introduction à MVC

MVC : Modèle Vue Contrôleur

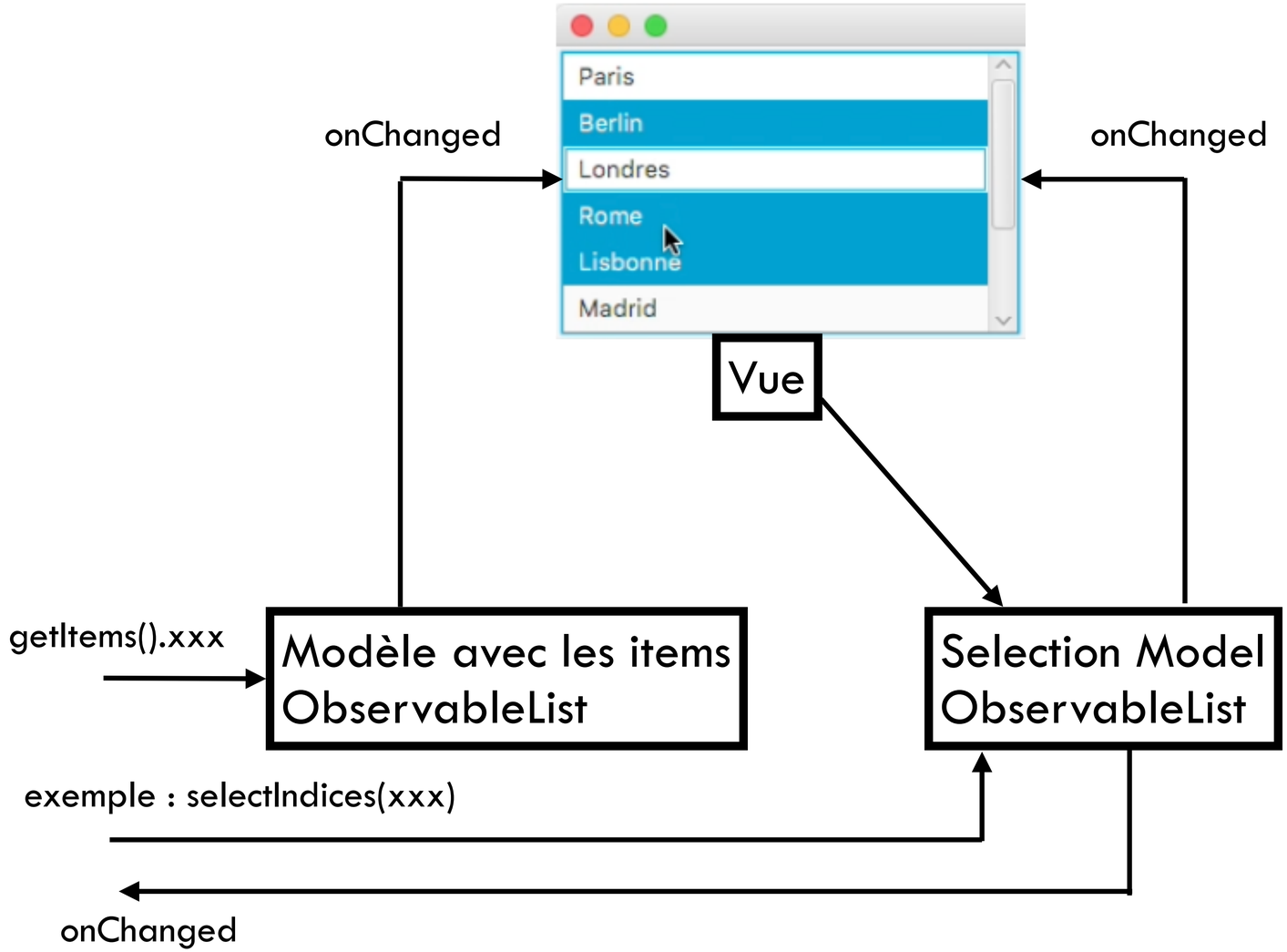
Idée : séparer les données d'une application de leur présentation

Le modèle informe la ou les vues de se mettre à jour

Une brève introduction à MVC



Exemple de ListView



ObservableList - ListChangeListener

void addListener(ListChangeListener<? super E> listener)

Interface ListChangeListener

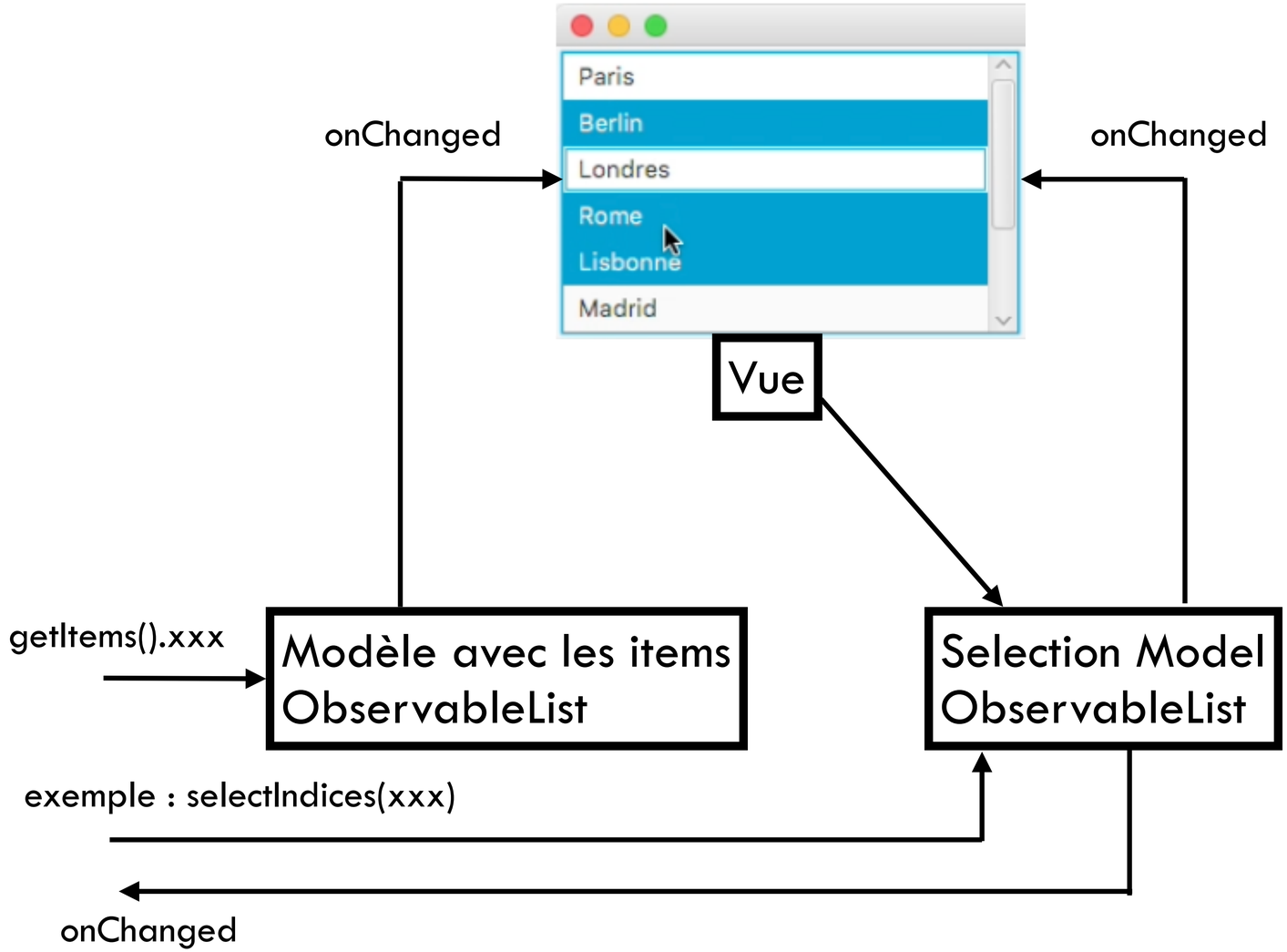
onChanged(ListChangeListener.Change<? super E> c)

Abonnement / notification

Abonnement aux changements d'état du modèle en utilisant **addListener**

Notification du changement d'état par l'appel de la méthode **onChanged**

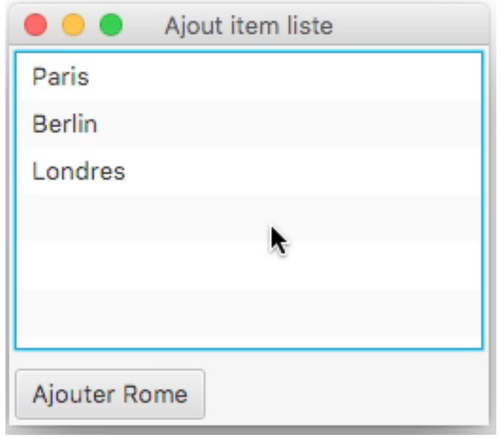
Exemple de ListView



ListView

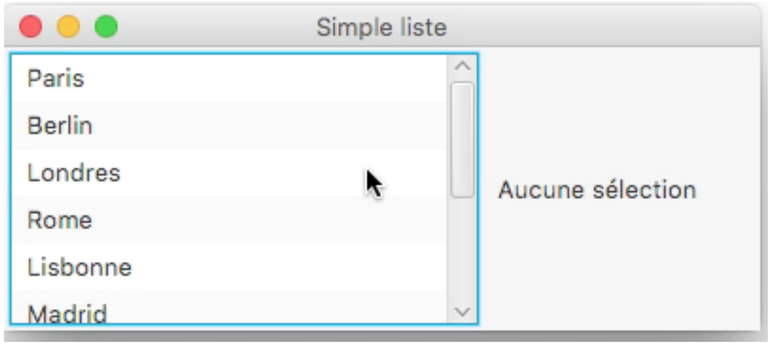
Illustration :

```
public class ListeViewAddItem extends Application {

    public void start(Stage stage) {
        ListView<String> list = new ListView<String>();
        list.getItems().addAll("Paris", "Berlin", "Londres");
        Button button = new Button("Ajouter Rome");
        button.setOnAction(e -> list.getItems().add("Rome"));

        VBox root = new VBox();
        root.setSpacing(10.0);
        root.setPadding(new Insets(3, 3, 3, 3));
        root.getChildren().addAll(list, button);

        Scene scene = new Scene(root);
        stage.setTitle("Ajout item liste");
        stage.setScene(scene);
        stage.show();
    }

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



ListView : gestion de la sélection simple


```
public class ListeSimple extends Application {
    Label label;

    class MonListChangeListener implements ListChangeListener<String> {
        public void onChanged(Change<? extends String> report) {
            label.setText("Sélection de " + report.getList());
        }
    }

    public void start(Stage stage) {
        label = new Label("Aucune sélection");
        ListView<String> list = new ListView<String>();
        list.getItems().addAll("Paris", "Berlin", "Londres", "Rome", "Lisbonne", "Madrid", "New York", "Tokyo", "Pékin");
        list.getSelectionModel().getSelectedItems().addListener(new MonListChangeListener());

        HBox root = new HBox();
        root.setAlignment(Pos.CENTER_LEFT);
        root.setSpacing(10.0);
        root.setPadding(new Insets(3, 3, 3, 3));
        root.getChildren().addAll(list,label);

        Scene scene = new Scene(root, 400, 150);
        stage.setTitle("Simple liste");
        stage.setScene(scene);
        stage.show();
    }

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



ListView : gestion de la sélection multiple


```
public class ListeSelectionMultiple extends Application {
    Label label;

    class MonListChangeListener implements ListChangeListener<String> {
        public void onChanged(Change<? extends String> report) {
            label.setText("Sélection de " + report.getList());
        }
    }

    public void start(Stage stage) {
        label = new Label("Aucune sélection");
        ListView<String> list = new ListView<String>();
        list.getItems().addAll("Paris", "Berlin", "Londres", "Rome", "Lisbonne", "Madrid", "New York", "Tokyo", "Pékin");
        list.getSelectionModel().setSelectionMode(SelectionMode.MULTIPLE);
        list.getSelectionModel().getSelectedItems().addListener(new MonListChangeListener());

        HBox root = new HBox();
        root.setAlignment(Pos.CENTER_LEFT);
        root.setSpacing(10.0);
        root.setPadding(new Insets(3, 3, 3, 3));
        root.getChildren().addAll(list,label);

        Scene scene = new Scene(root, 400, 150);
        stage.setTitle("Liste multi-sélection");
        stage.setScene(scene);
        stage.show();
    }

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

Résumé

MVC = Modèle, Vue, Contrôleur

MVC est utilisé par de nombreux widgets de JavaFX

- ListView possède 2 modèles (ObservableList) et une vue

Le modèle informe de son changement d'état par un mécanisme de listener (sytème d'abonnement)

- Utilisation d'un ListChangeListener pour ListView

Dessiner avec JavaFX

Utilisation de la classe **Canvas**

Pour dessiner des formes basiques, texte, chemins, images et pixels

Tout est dessiné dans le contexte graphique

canvas.getGraphicsContext2D()

Primitives de dessin

Primitives:

g.setFill(/*Color.UNE_COULEUR*/);

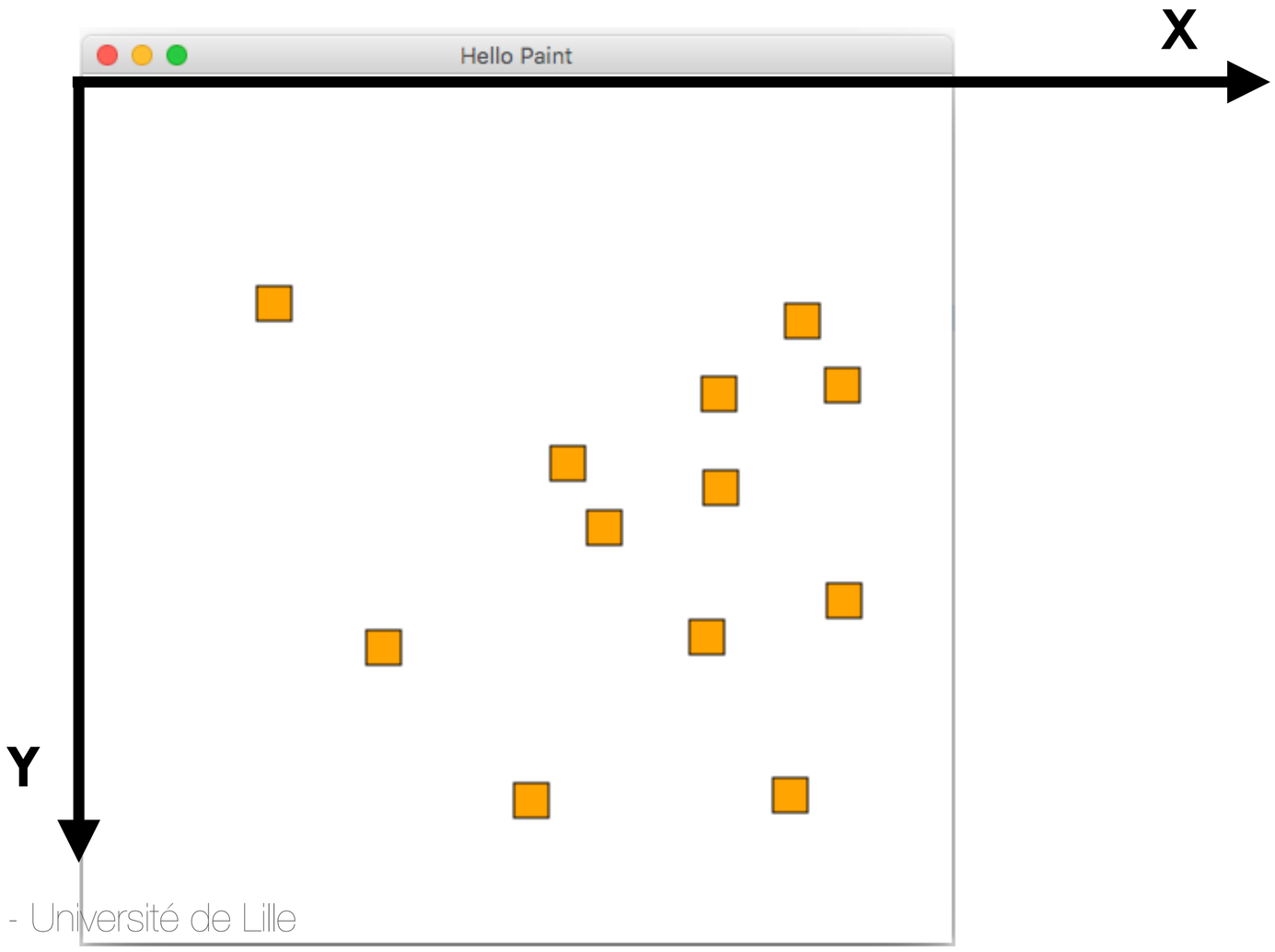
g.fillRect(...) ; g.fillOval(...) ; g.fillArc(...)

g.strokeRect(...) ; g.strokeOval(...) ; g.strokeArc(...) ;

g.strokeLine(...) ; g.drawImage(...) ; g.fillText(...)

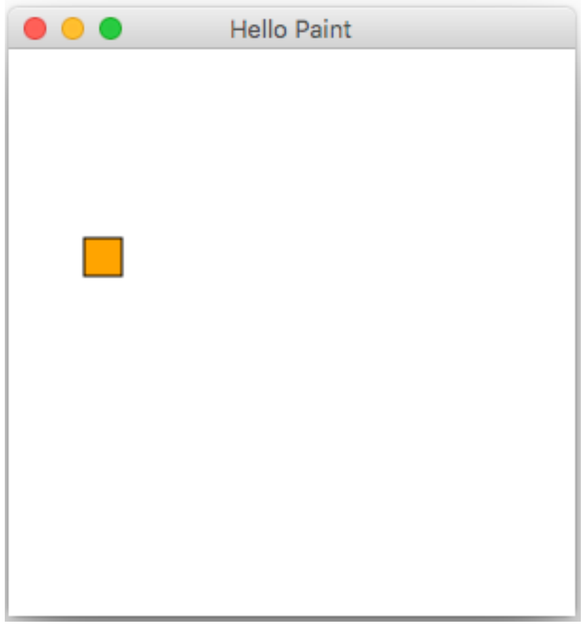
etc…

Système de coordonnées



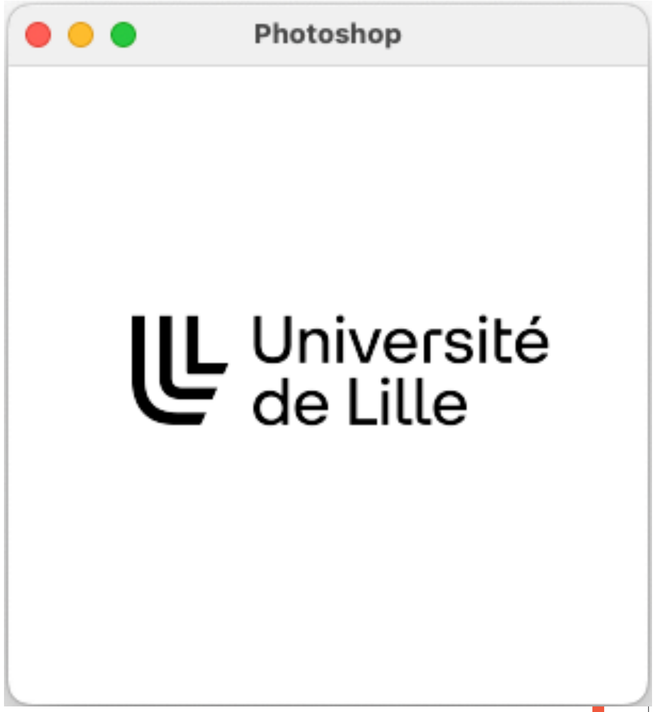
Exemple


```
public class HelloPaint extends Application {

    public void start(Stage stage) {
        VBox root = new VBox();
        Canvas canvas = new Canvas (300, 300);
        GraphicsContext gc = canvas.getGraphicsContext2D();
        gc.setFill(Color.ORANGE);
        gc.fillRect(40, 100, 20, 20);
        gc.setStroke(Color.BLACK);
        gc.strokeRect(40, 100, 20, 20);
        root.getChildren().add(canvas);

        Scene scene = new Scene(root);
        stage.setTitle("Hello Paint");
        stage.setScene(scene);
        stage.show();
    }

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



Afficher une image


```
public class Photoshop extends Application {

    public void start(Stage stage) {
        VBox root = new VBox();
        Canvas canvas = new Canvas (300, 300);
        GraphicsContext gc = canvas.getGraphicsContext2D();
        Image image = new Image("https://www.univ-lille.fr/fileadmin//user_upload/illustrations/logos/ULille.png", 250.0, 106.0, true, true);
        gc.drawImage(image, 30, 90);
        root.getChildren().add(canvas);

        Scene scene = new Scene(root);
        stage.setTitle("Photoshop");
        stage.setScene(scene);
        stage.show();
    }

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



Utilisation des styles

Personnalisation de l'apparence des éléments de l'interface

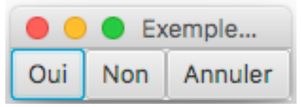
Même principe que les CSS en HTML

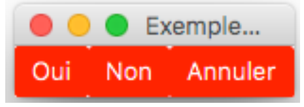
Exemple:

```
.button {
        -fx-background-color: red;
        -fx-text-fill: white;
}
```

Feuilles de styles : utilisation de fichiers CSS

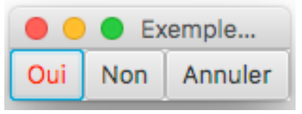

```
public class CSSExample extends Application {

    public void start(Stage stage) {

        Button yesBtn = new Button("Oui");
        Button noBtn = new Button("Non");
        Button cancelBtn = new Button("Annuler");
        HBox root = new HBox();
        root.getChildren().addAll(yesBtn, noBtn, cancelBtn);
        Scene scene = new Scene(root);

        scene.getStylesheets().add("resources/bouton.css");
        stage.setScene(scene);
        stage.setTitle("Exemple CSS");
        stage.show();
    }

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

}
```





Feuilles de style : avec du code


```
public class CSSExample extends Application {

    public void start(Stage stage) {

        Button yesBtn = new Button("Oui");
        yesBtn.setStyle("-fx-text-fill: red; -fx-font-weight: bold;");

        Button noBtn = new Button("Non");
        Button cancelBtn = new Button("Annuler");
        HBox root = new HBox();
        root.getChildren().addAll(yesBtn, noBtn, cancelBtn);
        Scene scene = new Scene(root);

        stage.setScene(scene);
        stage.setTitle("Exemple CSS");
        stage.show();
    }

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

}
```



Résumé

MVC

illustration avec ListView

dessin en JavaFX

feuilles de style