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

Cours 3 : Widgets et événements (1/2)

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

Compteur

Retour sur les interfaces en java

Les interfaces sont un type particulier de classe abstraite qui n'implémentent aucune méthode.

```
public interface MonInterface {
    void methode1();
}
```

```
public class ImplementMonInterface implements MonInterface {

    public void methode1() {
        System.out.println("Appel méthode 1");
    }
}
```

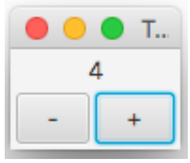
Interface EventHandler


```java
package javafx.event;

import java.util.EventListener;

// PENDING_DOC_REVIEW
/**
 * Handler for events of a specific class / type.
 *
 * @param <T> the event class this handler can handle
 * @since JavaFX 2.0
 */
@FunctionalInterface
public interface EventHandler<T extends Event> extends EventListener {
    /**
     * Invoked when a specific event of the type for which this handler is
     * registered happens.
     *
     * @param event the event which occurred
     */
    void handle(T event);
}
```

1ère solution : une classe externe par bouton

cours3/TestButtonIncDec.java


```
public class TestButtonIncDec extends Application {
    public void start(Stage stage) {
        Label label = new Label("0");
        Button bmoins = new Button("  -  ");
        Button bplus = new Button("  +  ");

        bplus.addEventHandler(ActionEvent.ACTION, new ClicListenerInc(label));
        bmoins.addEventHandler(ActionEvent.ACTION, new ClicListenerDec(label));

        VBox vbox = new VBox(3);
        vbox.setPadding(new Insets(3, 3, 3, 3));
        vbox.setAlignment(Pos.CENTER);
        HBox hbox = new HBox(3);
        hbox.getChildren().addAll(bmoins, bplus);
        vbox.getChildren().addAll(label, hbox);

        Scene scene = new Scene(vbox);
        stage.setTitle("TestButtonIncDec");
        stage.setScene(scene);
        stage.show();
    }

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




```
public class ClicListenerInc implements EventHandler<ActionEvent> {
    Label label;

    ClicListenerInc(Label lbl) {
        label = lbl;
    }

    public void handle(ActionEvent event) {
        int newValue = Integer.parseInt(label.getText()) + 1;
        label.setText("" + newValue);
    }
}
```

1ère solution : une classe externe par bouton

Ecritures équivalentes

```
bplus.addEventHandler(ActionEvent.ACTION, new ClicListenerInc(label));
```

```
ClicListenerInc myClicListenerInc = new ClicListenerInc(label);
bplus.addEventHandler(ActionEvent.ACTION, myClicListenerInc);
```

Classes internes en Java (inner class)

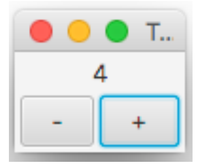
Une classe définie à l'intérieur d'une autre classe

Une classe interne peut accéder aux membres de sa classe englobante, même s'ils sont privés

2e solution : une classe interne

cours3/TestButtonIncDec2.java


```
public class TestButtonIncDec2 extends Application {
   Label label;
   Button bmoins, bplus;

   class ClicListenerIncDec implements EventHandler<ActionEvent> {
      public void handle(ActionEvent event) {
         int currentValue = Integer.parseInt(label.getText());
         if (event.getTarget() == bplus) {
            label.setText("" + (currentValue + 1));
         } else {
            label.setText("" + (currentValue - 1));
         }
      }
   }

   public void start(Stage stage) {
      label = new Label("0");
      bmoins = new Button("  -  ");
      bplus = new Button("  +  ");

      bplus.addEventHandler(ActionEvent.ACTION, new ClicListenerIncDec());
      bmoins.addEventHandler(ActionEvent.ACTION, new ClicListenerIncDec());

      VBox vbox = new VBox(3);
      vbox.setPadding(new Insets(3, 3, 3, 3));
      vbox.setAlignment(Pos.CENTER);
      HBox hbox = new HBox(3);
      hbox.getChildren().addAll(bmoins, bplus);
      vbox.getChildren().addAll(label, hbox);

      Scene scene = new Scene(vbox);
      stage.setTitle("TestButtonIncDec2");
      stage.setScene(scene);
      stage.show();
   }

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



Classes internes anonymes

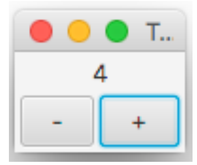
Une classe interne sans nom est une classe interne anonyme

Elles sont déclarées et instanciées en même temps

3e solution : deux classes internes anonymes

cours3/TestButtonIncDec3.java

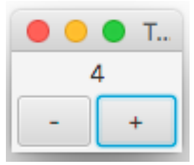

```
public class TestButtonIncDec3 extends Application {

   public void start(Stage stage) {
      Label label = new Label("0");
      Button bmoins = new Button("  -  ");
      Button bplus = new Button("  +  ");

      bplus.addEventHandler(ActionEvent.ACTION, new EventHandler<ActionEvent>(){
         public void handle(ActionEvent event) {
            int newValue = Integer.parseInt(label.getText()) + 1;
            label.setText("" + newValue);
         }
      });
      bmoins.addEventHandler(ActionEvent.ACTION, new EventHandler<ActionEvent>(){
         public void handle(ActionEvent event) {
            int newValue = Integer.parseInt(label.getText()) - 1;
            label.setText("" + newValue);
         }
      });

      VBox vbox = new VBox(3);
      vbox.setPadding(new Insets(3, 3, 3, 3));
      vbox.setAlignment(Pos.CENTER);
      HBox hbox = new HBox(3);
      hbox.getChildren().addAll(bmoins, bplus);
      vbox.getChildren().addAll(label, hbox);

      Scene scene = new Scene(vbox);
      stage.setTitle("TestButtonIncDec3");
      stage.setScene(scene);
      stage.show();
   }

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



4e solution : implémenter directement EventHandler<ActionEvent>

cours3/TestButtonIncDec4.java


```
public class TestButtonIncDec4 extends Application implements EventHandler<ActionEvent> {
   Label label;
   Button bmoins, bplus;

   public void start(Stage stage) {
      label = new Label("0");
      bmoins = new Button("  -  ");
      bplus = new Button("  +  ");

      bplus.addEventHandler(ActionEvent.ACTION, this);
      bmoins.addEventHandler(ActionEvent.ACTION, this);

      VBox vbox = new VBox(3);
      vbox.setPadding(new Insets(3, 3, 3, 3));
      vbox.setAlignment(Pos.CENTER);
      HBox hbox = new HBox(3);
      hbox.getChildren().addAll(bmoins, bplus);
      vbox.getChildren().addAll(label, hbox);

      Scene scene = new Scene(vbox);
      stage.setTitle("TestButtonIncDec4");
      stage.setScene(scene);
      stage.show();
   }

   public void handle(ActionEvent event) {
      int currentValue = Integer.parseInt(label.getText());
      if (event.getTarget() == bplus) {
         label.setText("" + (currentValue + 1));
      } else {
         label.setText("" + (currentValue - 1));
      }
   }

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



5e solution : expressions lambda

cours3/TestButtonIncDec5.java


```
public class TestButtonIncDec5 extends Application {

   public void start(Stage stage) {
      Label label = new Label("0");
      Button bmoins = new Button("  -  ");
      Button bplus = new Button("  +  ");

      bplus.addEventHandler(ActionEvent.ACTION, e -> {
         int currentValue = Integer.parseInt(label.getText());
         label.setText("" + (currentValue + 1));
      });
      bmoins.addEventHandler(ActionEvent.ACTION, e -> {
         int currentValue = Integer.parseInt(label.getText());
         label.setText("" + (currentValue - 1));
      });

      VBox vbox = new VBox(3);
      vbox.setPadding(new Insets(3, 3, 3, 3));
      vbox.setAlignment(Pos.CENTER);
      HBox hbox = new HBox(3);
      hbox.getChildren().addAll(bmoins, bplus);
      vbox.getChildren().addAll(label, hbox);

      Scene scene = new Scene(vbox);
      stage.setTitle("TestButtonIncDec5");
      stage.setScene(scene);
      stage.show();
   }

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

Syntaxe des expressions lambda


```
e -> {
    int currentValue = Integer.parseInt(label.getText());
    label.setText("" + (currentValue - 1));
}
```

```
e -> System.out.println(e.getEventType())
```

Expressions lambda

cours3/LambdaDemo.java

Raccourci de syntaxe aux classes internes anonymes

Ne peut être utilisé que pour les interfaces qui possèdent une seule méthode

```java
public interface Operation {
    public int runOperation(int a, int b);
}

public class Addition implements Operation {
    public int runOperation(int a, int b) {
        return a + b;
    }
}
```

```java
public class LambdaDemo {
    public int doOperation(Operation op, int a, int b) {
        return op.runOperation(a, b);
    }

    public static void main(String[] args) {
        LambdaDemo demo = new LambdaDemo();

        // Utilisation de l'interface implémentée
        Addition myOp = new Addition();
        int res1 = demo.doOperation(myOp, 3, 4);
        System.out.println(res1);

        // Utilisation d'une expression lambda
        int res2 = demo.doOperation((a, b) -> a + b, 3, 4);
        System.out.println(res2);
    }
}
```

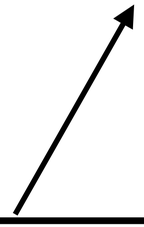
Portée des variables pour les lambdas

cours3/LambdaDemo2.java cours3/LambdaDemo3.java

Une expression lambda peut avoir accès aux variables définies dans son contexte englobant mais uniquement celles dont la valeur ne change pas

```
public class LambdaDemo2 {
    public int doOperation(Operation op, int a, int b) {
        return op.runOperation(a, b);
    }

    public static void main(String[] args) {
        LambdaDemo2 demo = new LambdaDemo2();

        int c = 1;
        int res = demo.doOperation((a, b) -> a + b + c, 3, 4);
        System.out.println(res);
    }
}
```

```
public class LambdaDemo3 {
    public int doOperation(Operation op, int a, int b) {
        return op.runOperation(a, b);
    }

    public static void main(String[] args) {
        LambdaDemo3 demo = new LambdaDemo3();

        for (int c = 1; c < 5; c++) {
            int res = demo.doOperation((a, b) -> a + b + c, 3, 4);
            System.out.println(res);
        }
    }
}
```



Local variable c defined in an enclosing scope must be final or effectively final

Abonneurs / abonnés

Deux catégories d'entités :

Un abonneur, la source d’information, le lieu où sont gérés les événements.

Un **ou des** abonnés, utilisant l’information produite.

Résumé des différentes étapes :

Abonnement de l’abonné auprès de l’abonneur.

Un événement se produit au niveau de l’abonneur.

L’abonneur notifie l’ensemble des abonnés en leur transmettant un événement.

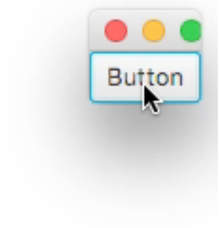
Chaque abonné peut traiter l’événement.

Illustration

cours3/PlusieursHandlers.java


```
public class PlusieursHandlers extends Application {
   public void start(Stage stage) {
      VBox root = new VBox();
      Button b = new Button("Button");
      b.addEventHandler(ActionEvent.ACTION, e -> System.out.println("Event handler 1"));
      b.addEventHandler(ActionEvent.ACTION, e -> System.out.println("Event handler 2"));
      b.addEventHandler(ActionEvent.ACTION, e -> System.out.println("Event handler 3"));
      root.getChildren().add(b);

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

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

```
Event handler 1
Event handler 2
Event handler 3
```



Utilisation de setOnXXX

cours3/TestButtonIncDec6.java

Méthode setOnXXX pour enregistrer **un** event handler sur un noeud

```
public class TestButtonIncDec6 extends Application {

    public void start(Stage stage) {
        Label label = new Label("0");
        Button bmoins = new Button("  -  ");
        Button bplus = new Button("  +  ");

        bplus.setOnAction(e -> {
            int currentValue = Integer.parseInt(label.getText());
            label.setText("" + (currentValue + 1));
        });

        bmoins.setOnAction(e -> {
            int currentValue = Integer.parseInt(label.getText());
            label.setText("" + (currentValue - 1));
        });

        // ...
    }
}
```

Utilisation de setOnXXX

De la même manière:

setOnMouseClicked

setOnMouseDragged

setOnKeyPressed

setOnTouchPressed

…

Recommandations

Privilégier l'utilisation des méthodes setOnXXX

Peu de code : utilisation des lambda expressions

Ne pas utiliser les classes internes anonymes

Plus de code : classe interne ou implémentation EventHandler

Encore plus de code : classe externe

Toujours privilégier la lisibilité du code

Dessiner avec JavaFX

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

Le contexte graphique définit l'ensemble des paramètres (couleur des traits, de remplissage, police courante, changements de repère…) qui sont appliqués lors du dessin d'un nouvel élément.

canvas.getGraphicsContext2D()

Primitives de dessin

Primitives :

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

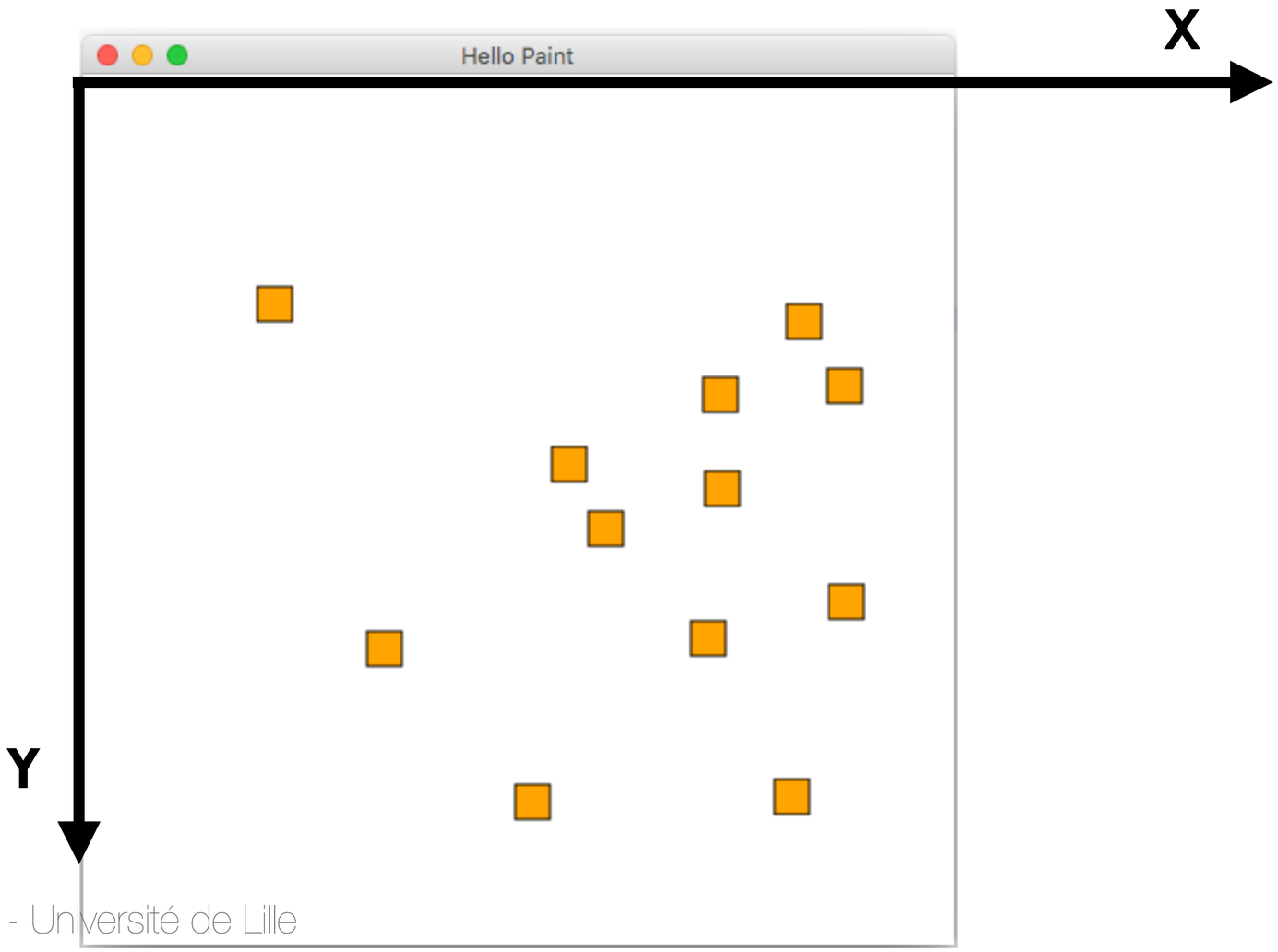
g.fillRect(...) ; g.fillOval(...) ; g.fillArc(...)

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

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

etc…

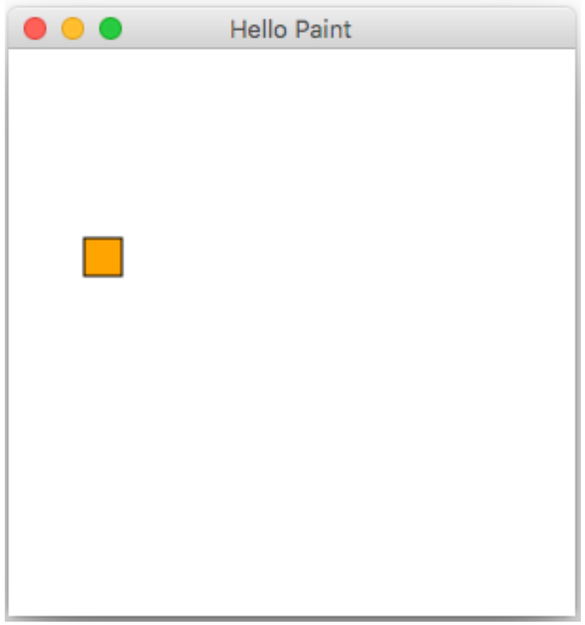
Système de coordonnées



Exemple

cours3/HelloPaint.java


```
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

cours3/Photoshop.java


```
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

Utilisation des styles

cours3/Photoshop.java

Personnalisation de l'apparence des éléments de l'interface

Même principe que les CSS en HTML

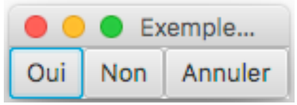
Exemple:

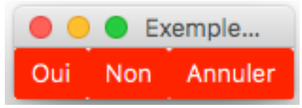
```
.button {
        -fx-background-color: red;
        -fx-text-fill: white;
}
```

Feuilles de styles : utilisation de fichiers CSS

cours3/CSSExampleStyleSheet.java


```
public class CSSExampleStyleSheet 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("cours3/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

cours3/CSSExample.java


```
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);
    }
}
```