TP3 : Développement d'IHM avec JavaFX
«Gestion des évènements dans JavaFX»

Illustrations: Propagation de l'événement dans l' « EventDispatchRoot » et fonctionnement de l' « EventFilter » et du « Eventhandler »

1. Taper le code suivant, cliquer une fois sur le bouton et une autre sur la scène, observer le résultat.

```
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.input.MouseEvent;
import javafx.stage.Stage;

public class TestEventFilter extends Application {
               public void start(Stage primaryStage) {
                 Group root = new Group();
                 primaryStage.setTitle("Illustration propagation evenements ");
                 primaryStage.setMinWidth(400);
                 primaryStage.setMinHeight(300);
                 Scene scene=new Scene(root);
                 // ajouter un bouton à la scene
                 Button btn = new Button("Bouton Test");
                 btn.setLayoutX(140);
                 btn.setLayoutY(100);
              // le bouton capte l'évènement
                 btn.addEventFilter(MouseEvent.MOUSE_CLICKED, (MouseEvent e) ->
{
                     System.out.println("Evenement traité par le bouton, Phase
de capture de l'événement");
                 });
            // le Group capte l'évènement
                 root.addEventFilter(MouseEvent.MOUSE_CLICKED, (MouseEvent e) ->
{
                  System.out.println("Evenement traité par le Group, Phase de
capture de l'événement");
                 });
               //la scène capte l'évènement
               scene.addEventFilter(MouseEvent.MOUSE_CLICKED, (MouseEvent e) ->
{ System.out.println("Evenement traité par la scene, Phase de capture de
l'événement");
                 });
               //le stage capte l'évènement
                 primaryStage.addEventFilter(MouseEvent.MOUSE_CLICKED,
(MouseEvent e) -> {
                 System.out.println("Evenement traité par le stage, Phase de
capture de l'événement");
                 });
                 root.getChildren().add(btn);
                 primaryStage.setScene(scene);
                 primaryStage.show();
               }
               public static void main(String[] args) {
Application.launch(args);         } }
```

2. Maintenant, appeler la méthode **e.consume()** juste après la définition d'un filtre (au choix dans le filtre du stage, scene, Group et bouton). observer le résultat.
3. Observer le comportement du handler et filter. Cliquer une fois sur le bouton et une autre fois sur la scène. Expliquer le comportement du handler par rapport au filter.

```
import javafx.application.Application;
import javafx.scene.Group;
import javafx.scene.Scene;
import javafx.scene.control.Button;
import javafx.scene.input.MouseEvent;
import javafx.stage.Stage;

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

                @Override
                public void start(Stage primaryStage) {
                  Group root = new Group();
                  primaryStage.setTitle("Illustration propagation evenements ");
                  primaryStage.setMinWidth(400);
                  primaryStage.setMinHeight(300);
                  Scene scene=new Scene(root);
                  // ajouter un bouton à la scene
                  Button btn = new Button("Bouton Test");
                  btn.setLayoutX(140);
                  btn.setLayoutY(100);
                  root.getChildren().add(btn);
               // le bouton capte l'évènement
                  btn.addEventFilter(MouseEvent.MOUSE_CLICKED, (MouseEvent e) -> {
                      System.out.println("Evenement traité par le bouton, Phase de
capture de l’événement");
                  });
                  // le Group capte l'évènement
                  root.addEventFilter(MouseEvent.MOUSE_CLICKED, (MouseEvent e) ->
{
                      System.out.println("Evenement traité par le Group, Phase de
capture de l’événement");
                  });
                //la scène capte l'évènement
                  scene.addEventFilter(MouseEvent.MOUSE_CLICKED, (MouseEvent e) ->
{
                  System.out.println("Evenement traité par la scene, Phase de
capture de l’événement");
                 //e.consume();
                  });
                //le stage capte l'évènement
                  primaryStage.addEventFilter(MouseEvent.MOUSE_CLICKED,
(MouseEvent e) -> {
                  System.out.println("Evenement traité par le stage, Phase de
capture de l’événement");
                 //e.consume();
                  });

                  // le bouton capte l'evenement
                   btn.addEventHandler(MouseEvent.MOUSE_CLICKED, (MouseEvent e) ->
{
                      System.out.println("Evenement traité par le bouton, Phase de
```

```
remonter de l’événement");
                    });
                // le Group capte l'évènement
                        root.addEventHandler(MouseEvent.MOUSE_CLICKED, (MouseEvent
e) -> {
                            System.out.println("Evenement traité par le Group,
Phase de remonter de l’événement");
                        });

                       //la scene capte l'evenement
                        scene.addEventHandler(MouseEvent.MOUSE_CLICKED,
(MouseEvent e) -> {
                        System.out.println("Evenement traité par la scene, Phase
de remonter de l’événement");
                        });
                      // le stage capte l'evenement
                        primaryStage.addEventHandler(MouseEvent.MOUSE_CLICKED,
(MouseEvent e) -> {
                            System.out.println("Evenement traité par le stage,
Phase de remonter de l’événement");
                         });

                  primaryStage.setScene(scene);
                  primaryStage.show();
                }
}
```

4. Observer le comportement des handler et filter aprés avoir rajouter le code suivant au programme.

```
btn.setSkin(new ButtonSkin(btn) {
                        {
                            this.consumeMouseEvents(false);
                        }
                    });
```