Gruppe 1

Projektideen: Business Programming 2

Charlotte Grollmisch
Jersey Otabor Izedonwmen

WIRTSCHAFTS UNIVERSITÄT WIEN VIENNA UNIVERSITY OF ECONOMICS AND BUSINESS

Idee: Race car game

Bei dieser Idee handelt es sich um ein interaktives Spiel, in dem der Spieler sich durch verschiedene Hindernisse durchschlingelt und Punkte sammelt. Das Interface wird hierbei mit Hilfe von javafx implementiert, BSF wird genutzt um javafx und anderwertige Komponenten überhaupt anzusteuern und AOO/OOO liefert ein Zertifikat und/oder wird genutzt, um Spielerstatistiken (Graphiken, usw.) zu speichern.

Anwendung:

- JavaFX
- BSF
- Jsoup or AOO/OOO

JavaFX und BSF

Start:

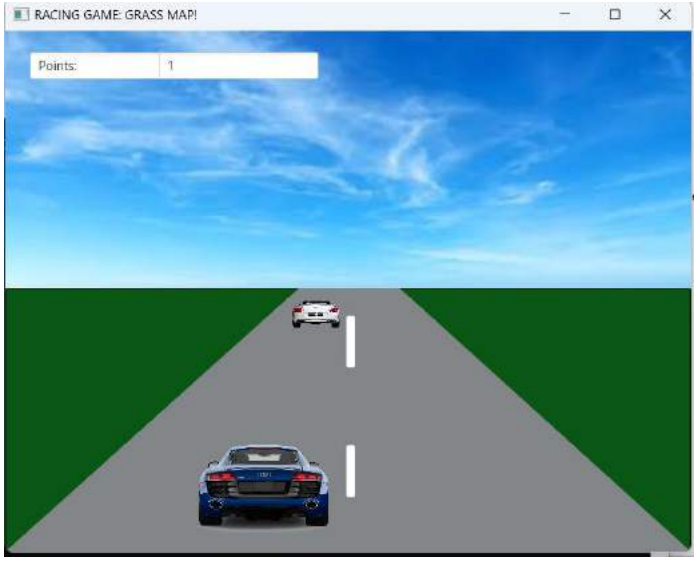
```
::method start          -- Rexx method "start" implements the abstract method
  use arg primaryStage  -- fetch the primary stage (window)
  primaryStage~setTitle("RACING GAME: GRASS MAP!")

   -- create an URL for the FMXLDocument.fxml file (hence the protocol "file:")
  fxmlUrl=.bsf~new("java.net.URL", "file:fxml_01.fxml")
   -- use FXMLLoader to load the FXML and create the GUI graph from its definitions:
  rootNode=bsf.loadClass("javafx.fxml.FXMLLoader")~load(fxmlUrl)
  rodeNode = rootNode~lookup("#testing1")
  duration =bsf.loadClass("javafx.util.Duration")~millis(1000)
  scene=.bsf~new("javafx.scene.Scene", rootNode)     -- create a scene for our document
  primaryStage~setScene(scene)   -- set the stage to our scene
  primaryStage~show              -- show the stage (and thereby our scene)
```


```
::routine startGame public
  /* @get(testing1 car1 car2 car3 pane1 pane2 pane3) */
  pane3~setVisible(.false)
  pane1~setVisible(.true)
  pane1~requestFocus
  rodeNode = testing1
  duration =bsf.loadClass("javafx.util.Duration")~millis(1000)
  rt= .bsf~new("javafx.animation.TranslateTransition", duration ,rodeNode)
  rt ~~setByY(308) ~~setCycleCount(100) ~~setAutoReverse(.false)~~setRate(1.5) ~~play
  .CrashController~new
  car2Node = car2 --rootNode~lookup("#car2")
  car3Node = car3 --rootNode~lookup("#car3")
  duration2 =bsf.loadClass("javafx.util.Duration")~millis(100)
```

JavaFX und BSF

Steuerung:

```
::routine buttonClicked public
  slotDir=arg(arg())  -- note: last argument is the slotDir argument from BSF4ooRexx
  /* @get(button1 car1) */
  myKey = slotDir~scriptcontext~getAttribute("event")~getCode~getName
  SELECT
    WHEN myKey="Left" THEN
      DO
        say GOING LEFT
        --say car1
        car1~X = car1~getX - 10.0
        --say car1~getX
      END
    WHEN myKey="Right" THEN
      DO
        say GOING RIGHT
        car1~X = car1~getX + 10.0
        --say car1~getX
      END
    OTHERWISE say "Wrong Key";
  END
```



Punkte:

```
::method handle
  car1 = arg(1)~getSource~getNode~scene~~root~lookup("#car1")
  crashed = arg(1)~getSource~getNode~getBoundsInParent~intersects(car1~getBoundsInParent)
  IF crashed THEN
  DO
    self~gameRun = .false
  END
  IF self~points > 25 THEN
  DO
    self~gameRun = .false
  END
  IF self~gameRun THEN
  DO
    self~points = self~points + 1
    arg(1)~getSource~getNode~scene~~root~lookup("#points1")~setText(self~points)
```

JavaFX und BSF

Crash:

```
para2  = .bsf~new("javafx.animation.ParallelTransition",car3Node)
para2~getChildren~~add(rt3)~~add(scale2)
crash1 = .CrashController~new(para,para2)
crash12=BSFCreateRexxProxy(crash1, ,"javafx.event.EventHandler")

para~setOnFinished(crash12)
para~~play

para2~setOnFinished(crash12)
--para2~~play
```

Ergebnisse:

```
::routine certify public
    /* @get(points1) */
  --say points1
  say "Creating a certification using the player's game stats"
  call createCertificate points1~getText, "0", "Grass Map"

 ::requires "Ergebnisse.rex"
```

AOO/OOO – LibreOffice Writer

Titel:

```
   /* create and insert a RectangleShape for displaying text */
shape = xDMsf~createInstance("com.sun.star.drawing.RectangleShape")
xShape = Shape~xShape
size = .bsf~new("com.sun.star.awt.Size")
size~Height = 1500
size~Width = 17000
xShape~setSize(size)
xShape~xPropertySet~setPropertyValue("blue", box("int", "C0 C0 C0"x ~c2d))
xText~insertTextContent(xTextCursor, shape~xTextContent, .false)

   /* insert text into the shape */
xShapeText = xShape~XText
xShapeText~setString("Ergebnisse: Car Race Game")
```

Grafik:

```
 /* create and insert a GraphicObjectShape with picture */
oGraph = xDMsf~createInstance("com.sun.star.drawing.GraphicObjectShape")
xGraph = oGraph~XShape
size = .bsf~new("com.sun.star.awt.Size")
size~Height = 5500
size~Width = 7500
xGraph~setSize(size)
xGraph~xPropertySet~setPropertyValue("GraphicURL", uno.convertToUrl(thisPath"\car-30984_1280.png"))
xText~insertTextContent(xTextCursor, oGraph~xTextContent , .false)
```

Tabelle :

```
maxRows=2
xTextTable = xDMsf~createInstance("com.sun.star.text.TextTable")~XTextTable ~~initialize(maxRows, 3)
xText~insertTextContent(xTextCursor, xTextTable, .false)


   /* insert colored heading text in the TextTable (first row) */
call setColoredCellText xTextTable, "A1", "Points Achieved!"
call setColoredCellText xTextTable, "B1", "Lives left:"
call setColoredCellText xTextTable, "C1", "Map"
xTextTable~getCellByName("A2")~XText~setString(arg(1))
xTextTable~getCellByName("B2")~XText~setString(arg(2))
xTextTable~getCellByName("C2")~XText~setString(arg(3))
```

Ergebnisse: Car Race Game

Points Achieved!	Lives left:	Map
1	0	Grass Map

Congratulations!