جامعة طرابلس
كلية تقنية المعلومات

البرمجة المرئية
Visual Programming

ITSE423

أستاذ المادة – حسن علي حسن
h.ebrahem@uot.edu.ly

المحاضرة الثالثة

مواضيع المحاضرة

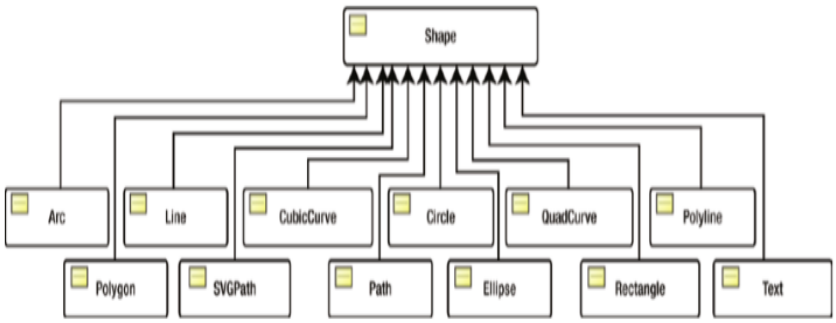
- الأشكال ثنائية الأبعاد 2D
  - كيفية رسم خط Line
  - كيفية رسم مستطيل Rectangle
  - كيفية رسم الدائرة Circle
  - كيفية رسم الشكل البيضاوي Ellipse
  - كيفية رسم المضلع Polygons
  - كيفية رسم الأقواس Arcs
  - كيفية رسم المنحنيات Quadratic Curves

الأشكال ثنائية الأبعاد 2D

- أي شكل يمكن رسمه في بعدين two-dimensional يسمى الشكل ثنائي الأبعاد.
- تقدم JavaFX مجموعة متنوعة من nodes لرسم أنواع مختلفة من الأشكال (خطوط ، دوائر ، مستطيلات ، إلخ).
- يمكنك إضافة الأشكال إلى scene graph.
- يمكن أن تكون الأشكال ثنائية الأبعاد 2D أو ثلاثية الأبعاد 3D. في هذا المحاضرة، سنناقش الأشكال ثنائية الأبعاد.

الأشكال ثنائية الأبعاد 2D

- جميع فئات الأشكال موجودة في الحزمة javafx.scene.shape يتم وراثة الفئات التي تمثل الأشكال ثنائية الأبعاد من فئة Shape كما هو موضح في الشكل التالي:



الأشكال ثنائية الأبعاد 2D

- الشكل shape له حجم size وموقع position يتم تحديهما عن طريق خصائصهما. على سبيل المثال، تحدد خصائص width وheight حجم المستطيل، كما تحدد خاصية نصف القطر radius حجم الدائرة.
- تحدد الخصائص x و y موضع الزاوية اليسرى العلوية للمستطيل، كما تحدد الخصائص centerX وcenterY مركز الدائرة ، إلخ.
- لا يتم تغيير حجم الأشكال من قبل وآبائهم parents أثناء التخطيط. يتغير حجم الشكل فقط عندما تتغير خصائصه المتعلقة به.

Hassan Ebrahem

5

كيفية رسم خط Line في 2D

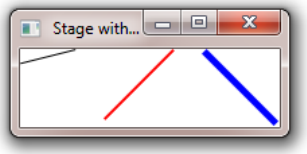
- الجزء التالي يصف بالتفصيل كيفية استخدام فئات JavaFX لرسم الأشكال ثنائية الأبعاد .
- يتم عرض الكائن line node من الفئة Line class.
- بشكل افتراضي ، يتم تعيين الخاصية fill الخاصة به على قيمة خالية Null.
- القيمة الافتراضية للون هي Color.BLACK والقيمة المبدئية للعرض strokeWidth هي 1.0.
- تحتوي فئة Line class على أربع خصائص:
- startX ، startY، endX، endY.

Hassan Ebrahem

6

مثال رسم خط Line في 2D

```
public class Shape2D extends Application {
    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle(" Stage with 2D ");
        Line line1 = new Line(10, 10, 50, 0);
        line1.setStrokeWidth(1.0);
        Line line2 = new Line(0, 50, 50, 0);
        line2.setStrokeWidth(2.0);
        line2.setStroke(Color.RED);
        Line line3 = new Line(0, 0, 50, 50);
        line3.setStrokeWidth(5.0);
        line3.setStroke(Color.BLUE);
        HBox root = new HBox(line1, line2, line3);
        root.setSpacing(20);
        Scene scene = new Scene(root);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
```



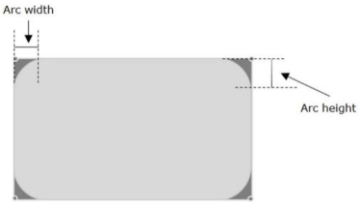
كيفية رسم مستطيل Rectangle في 2D

- يتم تمثيل rectangle node من Rectangle class.
- لرسم مستطيل نحتاج إلى ستة خصائص لتحديد المستطيل وهي:

- x
- y
- width
- height
- arcWidth
- arcHeight

كيفية رسم مستطيل Rectangle في 2D

- تمثل الخاصيتان x و y هما إحداثيات الركن الأيسر العلوي للمستطيل في نظام الإحداثيات للعقدة.
- الخاصيتان width و height هما تحددان عرض وارتفاع المستطيل، على التوالي.
- لرسم المربع قم بتحديد نفس قيم العرض والارتفاع.
- الخاصيتان arcWidth و arcHeight تحددان حواف المستطيل.
- القيمة المبدئية لجميع الخصائص تساوي 0.



كيفية رسم مستطيل Rectangle في 2D

- تحتوي فئة المستطيل Rectangle class على عدة أنواع لبناء المستطيل .
- دوال بناء المستطيل constructors هي:

- Rectangle()
- Rectangle(double width, double height)
- Rectangle(double x, double y, double width, double height)
- Rectangle(double width, double height, Paint fill)

- القيمة المبدئية لخصائص للدالة الأولى تساوي 0. وبالتالي لان يكون واضحاً عند التنفيذ. يجب اضافة التالي للبرنامج:
  - import javafx.scene.layout.HBox;
  - import javafx.scene.paint.Color;
  - import javafx.scene.shape.Rectangle;

مثال رسم مستطيل Rectangle في 2D

```
public class Rectangle2D extends Application {

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle(" Rectangle Stage with 2D ");
        Rectangle rectangle = new Rectangle(100,200,100,200);
        Rectangle rectangle2 = new Rectangle(100,50);
        Rectangle rectangle3 = new Rectangle(300,150, Color.RED);
        rectangle.setArcWidth(50);
        rectangle.setArcHeight(50);
        rectangle.setStrokeWidth(5.0);
        rectangle.setStroke(Color.BLUE);
        HBox root = new HBox(rectangle, rectangle2, rectangle3);
        root.setSpacing(20);
        Scene scene = new Scene(root);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
```

كيفية رسم الدائرة Circle في 2D

- يتم تمثيل circle node من Circle class. يستخدم class ثلاث خصائص لتحديد الدائرة وهي:
  - centerX
  - centerY
  - radius
- تمثل خاصيتي centerX و centerY إحداثيات x و y لمركز الدائرة.
- الخاصية الأخيرة تمثل نصف قطر الدائرة radius.
- القيم الافتراضية لهذه الخصائص هي صفر.

كيفية رسم الدائرة Circle في 2D

- تحتوي فئة الدائرة على عدة دوال constructors لبناء الدائرة وهي:

- Circle()
- Circle(double radius)
- Circle(double centerX, double centerY, double radius)
- Circle(double centerX, double centerY, double radius, Paint fill)
- Circle(double radius, Paint fill)

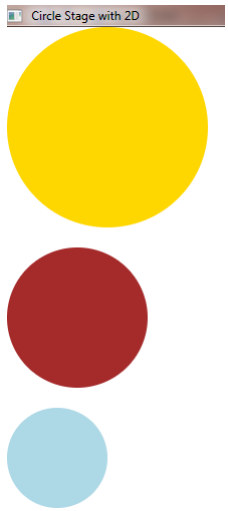
- يجب اضافة التالي للبرنامج:

- import javafx.scene.layout.VBox;
- import javafx.scene.paint.Color;
- import javafx.scene.shape.Circle;

مثال رسم الدائرة Circle في 2D

```
public class Circle2D extends Application {

    @Override
    public void start(Stage primaryStage) {
        primaryStage.setTitle(" Circle Stage with 2D ");
        Circle circle1 = new Circle(100, 50, 100);
        circle1.setFill(Color.GOLD);
        Circle circle2 = new Circle(70);
        circle2.setFill(Color.BROWN);
        Circle circle3 = new Circle(50, Color.LIGHTBLUE);
        VBox root = new VBox(circle1, circle2, circle3);
        root.setSpacing(20);
        Scene scene = new Scene(root, 300, 300);
        primaryStage.setScene(scene);
        primaryStage.show();
    }
```



كيفية رسم الشكل البيضاوي Ellipse في JavaFx 2D

- يمثل تمثيل ellipse node من الفئة Ellipse class. يستخدم class أربع خصائص لتعريف الشكل البيضاوي:
  - centerX
  - centerY
  - RadiusX
  - RadiusY
- تمثل خاصيتي centerX و centerY إحداثيات x و y لمركز الدائرة.
- الخاصيتان radiusX و radiusY تمثل نصف قطر القطع الناقصة في الاتجاهيين الافقي والراسي.
- القيم الافتراضية لهذه الخصائص هي صفر.

كيفية رسم الشكل البيضاوي Ellipse في JavaFx 2D

- يحتوي الشكل البيضاوي على عدة أنواع لبنائه constructors وهي:

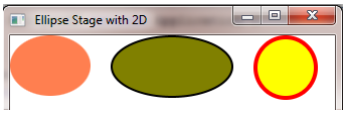
- Ellipse()
- Ellipse(double radiusX, double radiusY)
- Ellipse(double centerX, double centerY, double radiusX, double radiusY)

- يجب اضافة التالي في بداية للبرنامج:
  - import javafx.scene.layout.HBox;
  - import javafx.scene.paint.Color;
  - import javafx.scene.shape.Ellipse;

مثال رسم الشكل البيضاوي Ellipse في 2D

```
public void start(Stage primaryStage) {
    primaryStage.setTitle(" Ellipse Stage with 2D ");
    Ellipse e1 = new Ellipse(40, 30);
    e1.setFill(Color.CORAL);
    Ellipse e2 = new Ellipse(60, 30);
    e2.setFill(Color.OLIVE);
    e2.setStroke(Color.BLACK);
    e2.setStrokeWidth(2.0);
    Ellipse e3 = new Ellipse(30, 30);
    e3.setFill(Color.YELLOW);
    e3.setStroke(Color.RED);
    e3.setStrokeWidth(4.0);

    HBox root = new HBox(e1, e2, e3);
    root.setSpacing(20);
    Scene scene = new Scene(root, 300, 300);
    primaryStage.setScene(scene);
    primaryStage.show();
}
```

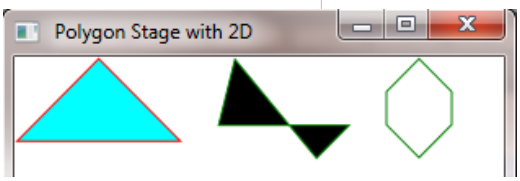


كيفية رسم المضلع Polygons في 2D

- يتم تمثيل polygon node من الفئة Polygon class. لا يوجد عدد محدد للخصائص.
- يتيح لك رسم المضلع باستخدام مصفوفة إحداثيات (x ، y) والتي تحدد رؤوس المضلع.
- يحتوي الشكل المضلع على دالتين لآنشائه constructors:
  - Polygon()
  - Polygon(double... points)
- يجب اضافة التالي في بداية للبرنامج:
  - import javafx.scene.shape.Polygon;

مثال رسم المضلع Polygons في 2D

```
public void start(Stage primaryStage) {
    primaryStage.setTitle(" Polygon Stage with 2D ");
    Polygon p1 = new Polygon();
    p1.getPoints().addAll(50.0, 0.0, 0.0, 50.0, 100.0, 50.0);
    p1.setFill(Color.AQUA);
    p1.setStroke(Color.RED);
    Polygon p2 = new Polygon();
    p2.getPoints().addAll(30.0, 10.0, 20.0, 50.0, 100.0, 50.0, 80.0, 70.0);
    p2.setFill(Color.BLACK);
    p2.setStroke(Color.GREEN);
    Polygon p3 = new Polygon();
    p3.getPoints().addAll(100.0, 0.0, 120.0, 20.0, 120.0, 40.0, 100.0, 60.0, 80.0 , 40.0, 80.0, 20.0);
    p3.setFill(Color.WHITE);
    p3.setStroke(Color.GREEN);
    HBox root = new HBox(p1, p2, p3);
    root.setSpacing(20);
    Scene scene = new Scene(root, 300, 300);
    primaryStage.setScene(scene);
    primaryStage.show();
}
```



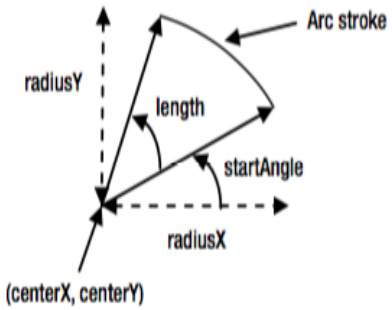
Hassan Ebrahem

19

كيفية رسم الأقواس Arcs في 2D

- يتم تمثيل القوس من الفئة Arc class.
- يتم تمثيل الاقواس Arc كجزء من الشكل البيضاوي.
- يتم استخدام سبع خصائص لتعريف الأقواس وهي:

- centerX
- centerY
- radiusX
- radiusY
- startAngle
- length
- Type

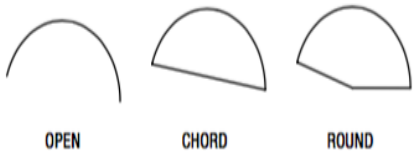


Hassan Ebrahem

20

كيفية رسم الأقواس Arcs في 2D

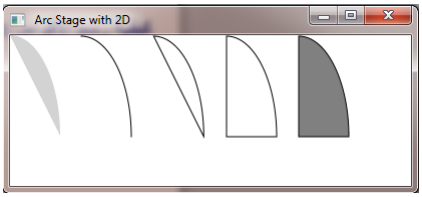
- تحدد خاصية النوع Type طريقة إغلاق القوس. وتأخذ الخاصية ArcType أحد القيم التالية:
- ArcType.OPEN لكي يكون القوس مفتوح.
- ArcType.CHORD لكي يكون القوس مغلق بربط نقطتي البداية والنهاية بخط مستقيم.
- ArcType.ROUND لكي يكون القوس مغلق بربط نقطة البداية والنهاية بمركز القطع الناقص.



كيفية رسم الأقواس Arcs في 2D

- تحتوي الفئة Arc class على دالتين constructors لبنائه وهم:
  - Arc()
  - Arc(double centerX, double centerY, double radiusX, double radiusY, double startAngle, double length)
- يجب اضافة التالي في بداية للبرنامج:
  - import javafx.scene.layout.HBox;
  - import javafx.scene.paint.Color;
  - import javafx.scene.shape.Arc;
  - import javafx.scene.shape.ArcType;

مثال رسم الأقواس Arcs في 2D

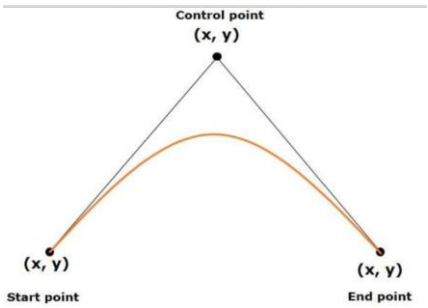



```
public void start(Stage primaryStage) {
    primaryStage.setTitle(" Arc Stage with 2D ");
    Arc A1 = new Arc(0, 0, 50, 100, 0, 90);
    A1.setFill(Color.LIGHTGRAY);
    Arc A2 = new Arc(0, 0, 50, 100, 0, 90);
    A2.setFill(Color.TRANSPARENT);
    A2.setStroke(Color.BLACK);
    Arc A3 = new Arc(0, 0, 50, 100, 0, 90);
    A3.setFill(Color.TRANSPARENT);
    A3.setStroke(Color.BLACK);
    A3.setType(ArcType.CHORD);
    Arc A4 = new Arc(0, 0, 50, 100, 0, 90);
    A4.setFill(Color.TRANSPARENT);
    A4.setStroke(Color.BLACK);
    A4.setType(ArcType.ROUND);
    Arc A5 = new Arc(0, 0, 50, 100, 0, 90);
    A5.setFill(Color.GRAY);
    A5.setStroke(Color.BLACK);
    A5.setType(ArcType.ROUND);
    HBox root = new HBox(A1, A2, A3, A4, A5);
    root.setSpacing(20);
    Scene scene = new Scene(root, 400, 150);
    primaryStage.setScene(scene);
    primaryStage.show();
}
```


كيفية رسم المنحنيات Quadratic Curves

- تستخدم QuadCurve لرسم المنحنيات حيث يتقاطع الخط مع نقطتين محددتين باستخدام نقطة تحكم محددة.
- يتم استخدام ستة خصائص لتعريف المنحنيات وهي:

- startX
- startY
- controlX
- controlY
- endX
- endY



JavaFx Quadratic Curves كيفية رسم المنحنيات

- تحتوي المنحنيات على دالتين constructors لبنائها وهي:

- QuadCurve()
- QuadCurve(double startX, double startY, double controlX, double controlY, double endX, double endY)

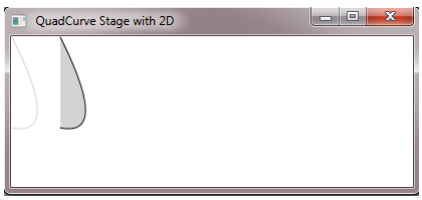
- يجب اضافة التالي في بداية للبرنامج:

- import javafx.scene.layout.HBox;
- import javafx.scene.paint.Color;
- import javafx.scene.shape.QuadCurve;

JavaFx Quadratic Curves مثال رسم المنحنيات

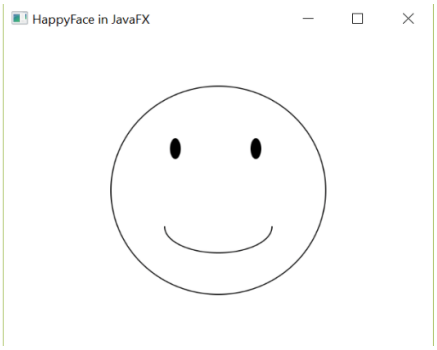
```
public void start(Stage primaryStage) {
    primaryStage.setTitle(" QuadCurve Stage with 2D ");

    QuadCurve qc1 = new QuadCurve(0, 0, 50, 100, 0, 90);
    qc1.setFill(Color.LIGHTGRAY);
    qc1.setStroke(Color.BLACK);
    QuadCurve qc2 = new QuadCurve(0, 0, 50, 100, 0, 90);
    qc2.setFill(Color.TRANSPARENT);
    qc2.setStroke(Color.LIGHTGRAY);
    HBox root = new HBox(qc2, qc1);
    root.setSpacing(20);
    Scene scene = new Scene(root, 400, 150);
    primaryStage.setScene(scene);
    primaryStage.show();
}
```



واجب

- اكتب التعليمات البرمجية المصدر لتنفيذ واجهة المستخدم التالية:



نهاية المحاضرة