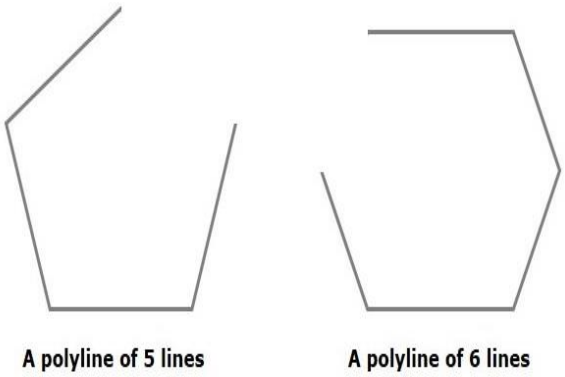
Draw Polyline

A Polyline is same as a polygon except that a polyline is not closed in the end. Or, continuous line composed of one or more line segments. In short, we can say a polygon is an open figure formed by coplanar line segments.



In JavaFX, a Polyline is represented by a class named **Polygon**. This class belongs to the package **javafx.scene.shape.**.

By instantiating this class, you can create polyline node in JavaFX. You need to pass the x, y coordinates of the points by which the polyline should be defined in the form of a double array.

You can pass the double array as a parameter of the constructor of this class as shown below −

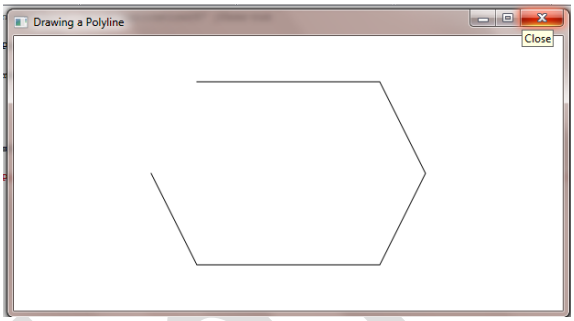
```
Polyline polyline = new Polyline(doubleArray);
```

Or, by using the **getPoints()** method as follows −

```
polyline.getPoints().addAll(new Double[]{List of XY coordinates separated by commas });
```

```
package javafxapplication28;

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.scene.shape.Polyline;

public class JavaFXApplication28 extends Application {

    @Override
    public void start(Stage stage) {
      //Creating a polyline
      Polyline polyline = new Polyline();

      //Adding coordinates to the polygon
      polyline.getPoints().addAll(new Double[]{
         200.0, 50.0,
         400.0, 50.0,
         450.0, 150.0,
         400.0, 250.0,
```

```
            200.0, 250.0,
            150.0, 150.0,
        });

        //Creating a Group object
        Group root = new Group(polyline);

        //Creating a scene object
        Scene scene = new Scene(root, 600, 300);

        //Setting title to the Stage
        stage.setTitle("Drawing a Polyline");

        //Adding scene to the stage
        stage.setScene(scene);

        //Displaying the contents of the stage
        stage.show();
    }
}
```



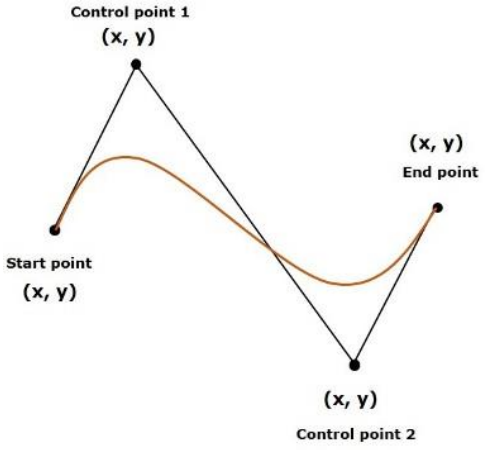
Draw CubicCurve

A CubicCurve is described by a third-degree polynomial function of two variables, and can be written in the following form −

$$a_1x^3 + a_2x^2y + a_3xy^2 + a_4y^3 + a_5x^2 + a_6xy + a_7y^2 + a_8x + a_9y + a_{10} = 0.$$

These Bezier curves are generally used in computer graphics. They are parametric curves which appear reasonably smooth at all scales. These curves are drawn based on points on the XY plane.

A cubic curve is a Bezier parametric curve in the XY plane is a curve of degree 3. It is drawn using four points − Start Point, End Point, Control Point and Control Point2 as shown in the following diagram.



In JavaFX, a CubicCurve is represented by a class named CubicCurve. This class belongs to the package javafx.scene.shape.
By instantiating this class, you can create a CubicCurve node in JavaFX.
This class has 8 properties of the double datatype namely −

- startX − The x coordinate of the starting point of the curve.
- startY − The y coordinate of the starting point of the curve.
- controlX1 − The x coordinate of the first control point of the curve.
- controlY1 − The y coordinate of the first control point of the curve.
- controlX2 − The x coordinate of the second control point of the curve.

- controlY2 − The y coordinate of the second control point of the curve.
- endX − The x coordinate of the end point of the curve.
- endY − The y coordinate of the end point of the curve.

To draw a cubic curve, you need to pass values to these properties, either by passing them to the constructor of this class, in the same order, at the time of instantiation, as shown below −

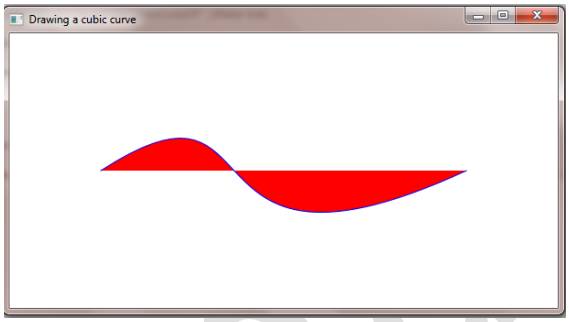
```
CubicCurve cubiccurve = new CubicCurve(
  startX, startY, controlX1, controlY1, controlX2, controlY2, endX, endY);
```

Or, by using their respective setter methods as follows −

```
setStartX(value);
setStartY(value);
setControlX1(value);
setControlY1(value);
setControlX2(value);
setControlY2(value);
setEndX(value);
setEndY(value);
```

```
package javafxapplication29;

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.scene.paint.Color;
import javafx.scene.shape.CubicCurve;

public class JavaFXApplication29 extends Application {

   @Override
   public void start(Stage stage) {

    //Drawing a cubic curve
    CubicCurve cubicCurve = new CubicCurve();

    //Setting properties to cubic curve
    cubicCurve.setStartX(100.0);
    cubicCurve.setStartY(150.0);
    cubicCurve.setControlX1(300.0);
    cubicCurve.setControlY1(20.0);
    cubicCurve.setControlX2(175.0);
    cubicCurve.setControlY2(300.0);
    cubicCurve.setEndX(500.0);
    cubicCurve.setEndY(150.0);

    cubicCurve.setStroke(Color.BLUE);
    cubicCurve.setFill(Color.RED);
```

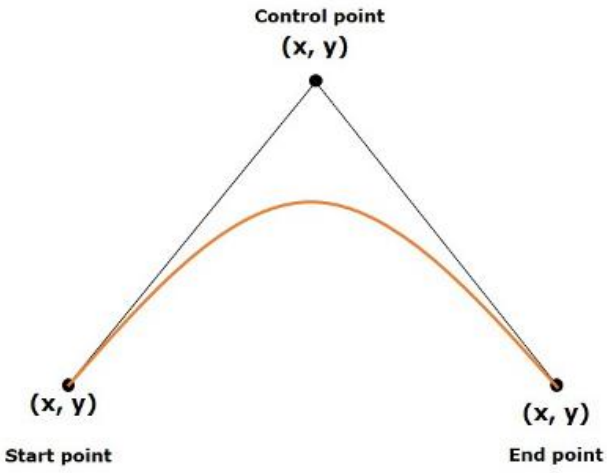
```
        //Creating a Group object
        Group root = new Group(cubicCurve);

        //Creating a scene object
        Scene scene = new Scene(root, 600, 300);

        //Setting title to the Stage
        stage.setTitle("Drawing a cubic curve");

        //Adding scene to the stage
        stage.setScene(scene);

        //Displaying the contents of the stage
        stage.show();
      }
      public static void main(String args[]){
        launch(args);
      }
}
```



Draw quadratic curve

Mathematically a quadratic curve is one that is described by a quadratic function like − $y = ax^2 + bx + c$. In computer graphics Bezier curves are used. These are parametric curves which appear reasonably smooth at all scales. These Bezier curves are drawn based on points on an XY plane.

A quadratic curve is a Bezier parametric curve in the XY plane which is a curve of degree 2. It is drawn using three points: **start point, end point** and **control point** as shown in the following diagram



In JavaFX, a QuadCurve is represented by a class named **QuadCurve**. This class belongs to the package **javafx.scene.shape**.

By instantiating this class, you can create a QuadCurve node in JavaFX.

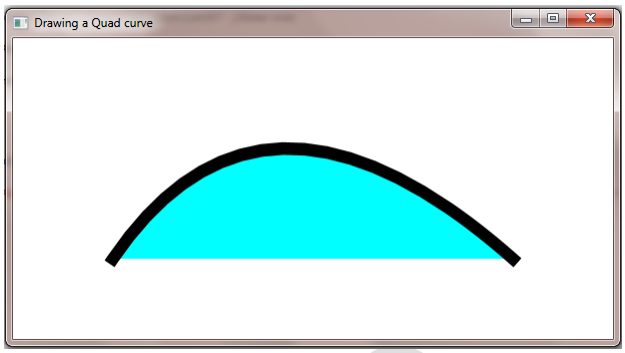
This class has 6 properties of the double datatype namely –

startX − The x coordinate of the starting point of the curve.
startY − The y coordinate of the starting point of the curve.
controlX − The x coordinate of the control point of the curve.
controlY − The y coordinate of the control point of the curve.

endX − The x coordinate of the end point of the curve.
endY − The y coordinate of the end point of the curve.
To draw a QuadCurve, you need to pass values to these properties. This can be done either by passing them to the constructor of this class, in the same order, at the time of instantiation, as follows −
QuadCurve quadcurve = new QuadCurve(startX, startY, controlX, controlY, endX, endY);
Or, by using their respective setter methods as follow −
setStartX(value);
setStartY(value);
setControlX(value);
setControlY(value);
setEndX(value);
setEndY(value);

```
package javafxapplication30;

import javafx.application.Application;
import javafx.stage.Stage;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.scene.paint.Color;
import javafx.scene.shape.QuadCurve;

public class JavaFXApplication30 extends Application {

   @Override
   public void start(Stage stage) {
 //Creating a QuadCurve
    QuadCurve quadCurve = new QuadCurve();

    //Adding properties to the Quad Curve
    quadCurve.setStartX(100.0);
    quadCurve.setStartY(220.0);
    quadCurve.setEndX(500.0);
    quadCurve.setEndY(220.0);
    quadCurve.setControlX(250.0);
    quadCurve.setControlY(0.0);
    quadCurve.setFill(Color.AQUA);
    quadCurve.setStrokeWidth(12);
    quadCurve.setStroke(Color.BLACK);

    //Creating a Group object
    Group root = new Group(quadCurve);

    //Creating a scene object
```

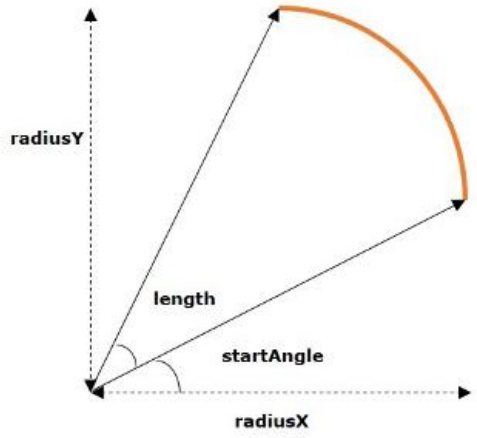
```
        Scene scene = new Scene(root, 600, 300);

        //Setting title to the Stage
        stage.setTitle("Drawing a Quad curve");

        //Adding scene to the stage
        stage.setScene(scene);

        //Displaying the contents of the stage
        stage.show();
    }
}
```



Draw Arc

An arc is part of a curve. It is described by the following properties −
length − The distance along the arc.
angle − The angle the curve makes at the centre of the circle.
radiusX − The width of the full Ellipse of which the current arc is a part of.
radiusY − The height of the full Ellipse of which the current arc is a part of.

In JavaFX, an arc is represented by a class named **Arc**.
This class belongs to the package **javafx.scene.shape**.
By instantiating this class, you can create an arc node in JavaFX.
This class has a few properties of the double datatype namely −
centerX − The x coordinate of the center of the arc.
centerY − The y coordinate of the center of the arc.
radiusX − The width of the full ellipse of which the current arc is a part of.
radiusY − The height of the full ellipse of which the current arc is a part of.
startAngle − The starting angle of the arc in degrees.
length − The angular extent of the arc in degrees.



To draw an arc, you need to pass values to these properties, either by passing them to the constructor of this class, in the same order, at the time of instantiation, as shown below −

```
Circle circle = new Circle(centerX, centerY, radiusX, radiusY);
```

Or, by using their respective setter methods as follows −

```
setCenterX(value);
setCenterY(value);
setRadiusX(value);
setRadiusY(value);
```