

1.4 (optional) Graphics Supplement: Outline

- 1. A Sample JavaFX Application
- 2. Size and Position of Figures
- 3. Drawing Ovals and Circles
- 4. Drawing Arcs



The logo graphic consists of a black crosshair centered over a yellow square, a red square, and a blue square. The text "JavaFX" is positioned to the right of the crosshair.

JavaFX

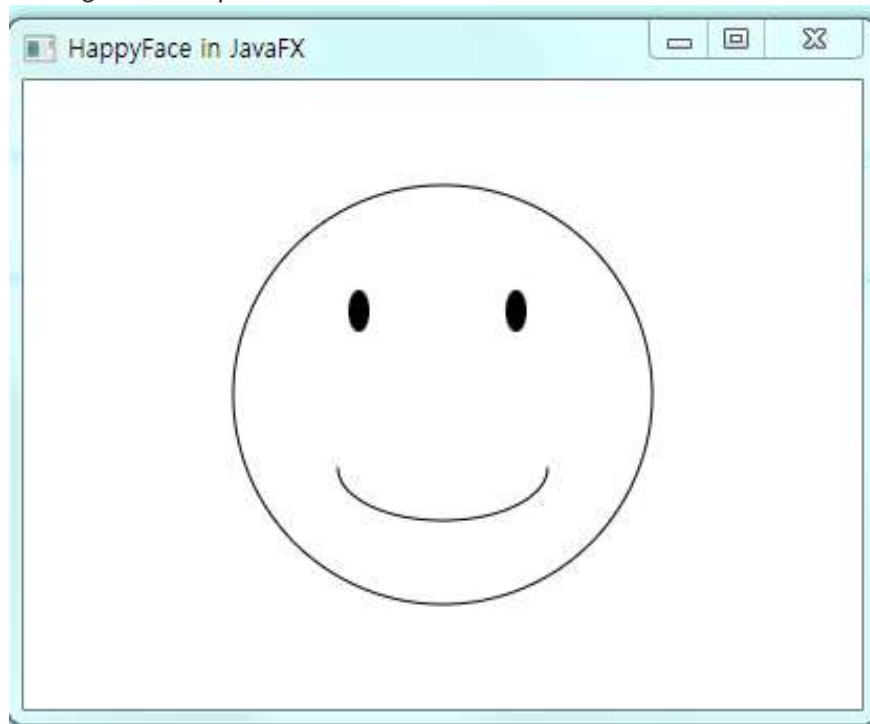
- JavaFX is a set of **APIs** that allow Java programmers to create **graphics and media applications**
 - 2D, 3D games
 - Visual effects
 - Touch-enabled applications
 - **Successor** to AWT and Swing for making **graphical applications**
- In JavaFX you add **scenes** to the **stage**. A **canvas** is an object that you can draw shapes on.



A Sample JavaFX Application

- View [sample program](#) Listing 1.2
 - **class HappyFace**

Program Output



Sample
screen
output



// Listing 1.2

```
import javafx.application.Application;
import javafx.scene.canvas.Canvas;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.stage.Stage;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.shape.ArcType;

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

    @Override
    public void start(Stage primaryStage) throws Exception
    {
        Group root = new Group();
        Scene scene = new Scene(root);

        Canvas canvas = new Canvas(400, 300);
        GraphicsContext gc = canvas.getGraphicsContext2D();
        gc.strokeOval(100, 50, 200, 200);
        gc.fillOval(155, 100, 10, 20);
        gc.fillOval(230, 100, 10, 20);
        gc.strokeArc(150, 160, 100, 50, 180, 180, ArcType.OPEN);

        root.getChildren().add(canvas);
        primaryStage.setTitle("HappyFace in JavaFX");
        primaryStage.setScene(scene);
        primaryStage.show();
    }
}
```





이클립스에서 JavaFX 설치하기

- 1) 이클립스를 키고 Help -> eclipse Market place에 접속
- 2) 검색에서 fx로 검색을 하면
e(fx)clipse 플러그인이 나옴 (이클립스 마켓플레이스는 이클립스에서 사용할 다양한 플러그인을 받아서 설치하는 곳)



- 3) Install 버튼을 눌러서 설치
- 4) 두 설치가 완료된 후에는 이클립스를 반드시 재시작해주어야 합니다.
- (반드시 최신버전의 이클립스를 사용해야함. 구버전의 이클립스를 사용할 경우 설치가 안될 수 있음)


Eclipse Marketplace

Eclipse Marketplace

Select solutions to install. Press Install Now to proceed with installation.
Press the "more info" link to learn more about a solution.

Search Recent Popular Favorites Installed Research@Eclipse

Find: fx All Markets All Categories Go




e(fx)clipse 3.6.0

e(fx)clipse is a set of plugins who make developing JavaFX 2 application with your favorite IDE an excellent experience. It provides wizards, specialized CSS and... [more info](#)

by [BestSolution.at](#), EPL
[javaafx](#)

★ 306 Installs: **273K** (6,828 last month) [Install](#)



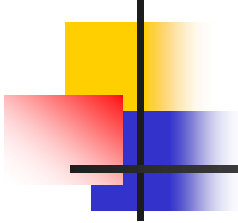
Jubula Automated Functional Testing Feature 7.0

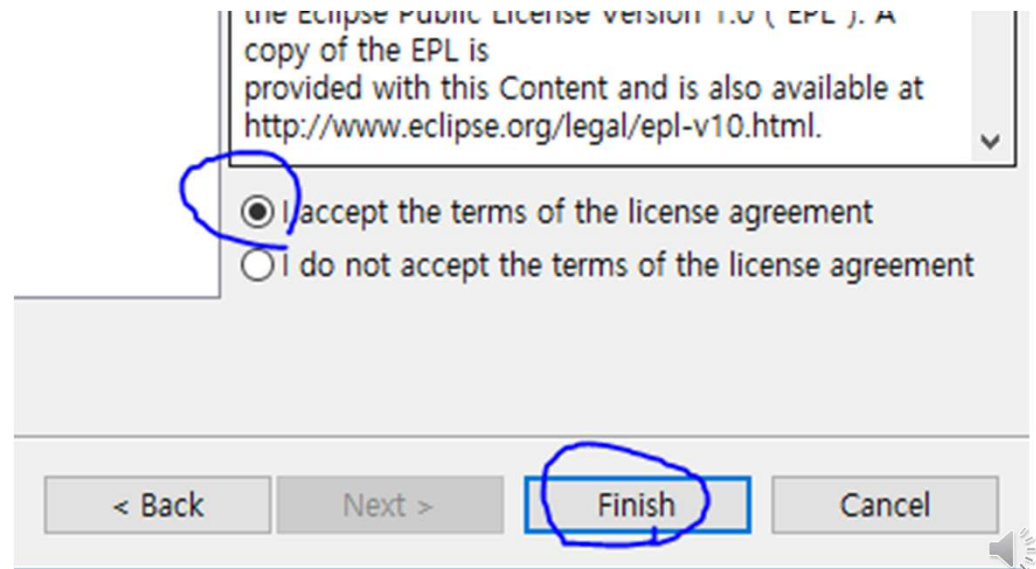
Jubula provides automated functional GUI testing. It is aimed at teams who want their automated tests to be written by test experts from the user perspective,... [more info](#)

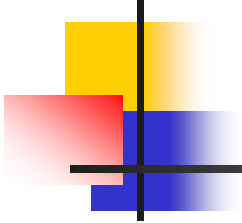
by [BREDEX GmbH](#), EPL
[SWT UI Testing](#) [Swing UI Testing](#) [runtime testing](#) [Jubula](#)

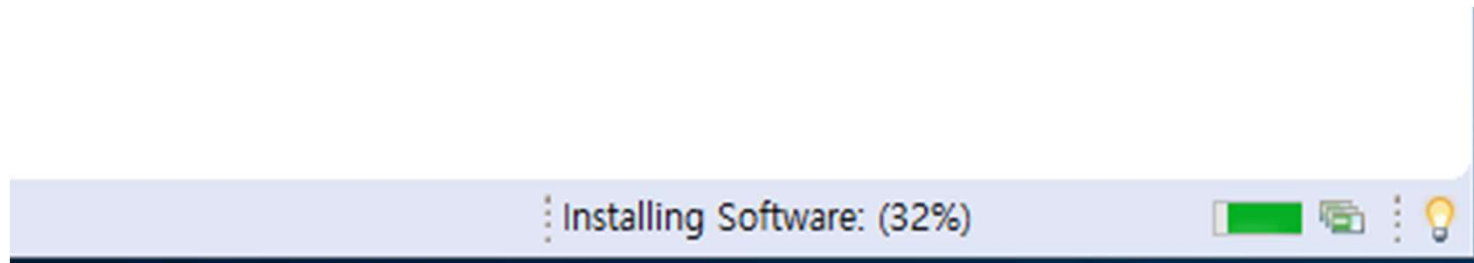
★ 13 Installs: **6.14K** (47 last month) [Install](#)

EMF Forms 1.21.0

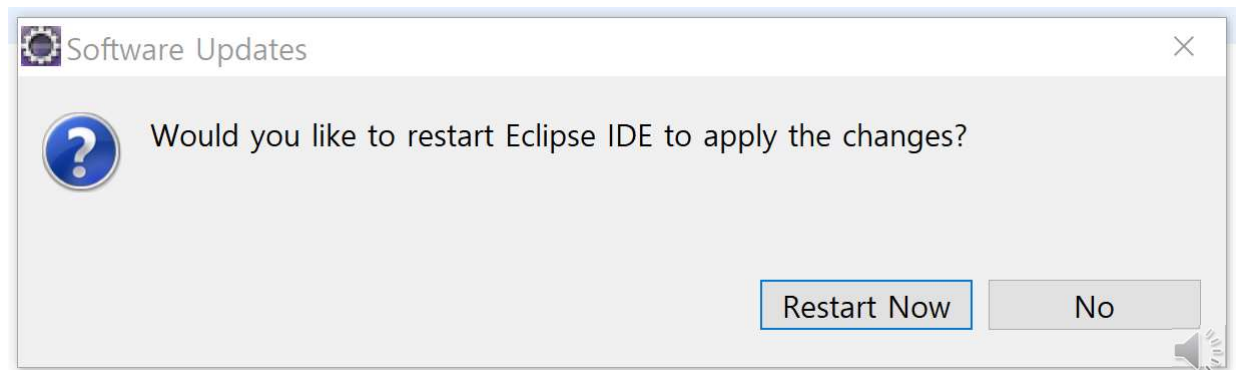
- 
- 설치 중간에 나오는 라이선스 관련 물음
은 I accept the terms of the license
agreement 를 선택후 finish를 눌러줌

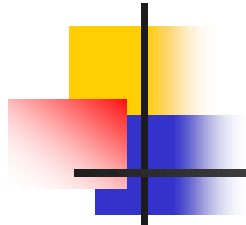


- 
- 하단에 초록색 게이지가 올라가면서 설치되고 있는 것을 확인할 수 있음



- Restart





- <https://docs.oracle.com/javase/8/javafx/api/>

The screenshot shows a web browser window with the URL `docs.oracle.com/javase/8/javafx/api/`. The page title is "Overview (JavaFX 8)". The left sidebar contains a navigation menu with "All Classes" and "Packages". Under "Packages", the following packages are listed: `javafx.animation`, `javafx.application`, `javafx.beans`, `javafx.beans.binding`, and `javafx.beans.property`. The main content area has a top navigation bar with tabs: "OVERVIEW" (selected), "PACKAGE", "CLASS", "USE", "TREE", "DEPRECATED", "INDEX", and "HELP". Below this is a sub-navigation bar with "PREV", "NEXT", "FRAMES", and "NO FRAMES". The main content area displays a table titled "Packages" with two columns: "Package" and "Description".

Package	Description
<code>javafx.animation</code>	Provides the set of classes for ease of use transition
<code>javafx.application</code>	Provides the application life-cycle classes.
<code>javafx.beans</code>	The package <code>javafx.beans</code> contains the interfaces



A Sample JavaFX Application

- The **start** method
 - the starting method for a JavaFX application, not the *main* method
- The first four lines of the **start** method
 - set up a canvas on a scene for you to draw simple graphics



// Listing 1.2

```
import javafx.application.Application;
import javafx.scene.canvas.Canvas;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.stage.Stage;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.shape.ArcType;
```

```
public class HappyFace extends Application
```

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

```
@Override
```

```
public void start(Stage primaryStage) throws Exception
```

```
{
    Group root = new Group();
    Scene scene = new Scene(root);
```

```
    Canvas canvas = new Canvas(400, 300);
    GraphicsContext gc = canvas.getGraphicsContext2D();
    gc.strokeOval(100, 50, 200, 200);
    gc.fillOval(155, 100, 10, 20);
    gc.fillOval(230, 100, 10, 20);
    gc.strokeArc(150, 160, 100, 50, 180, 180, ArcType.OPEN);
```

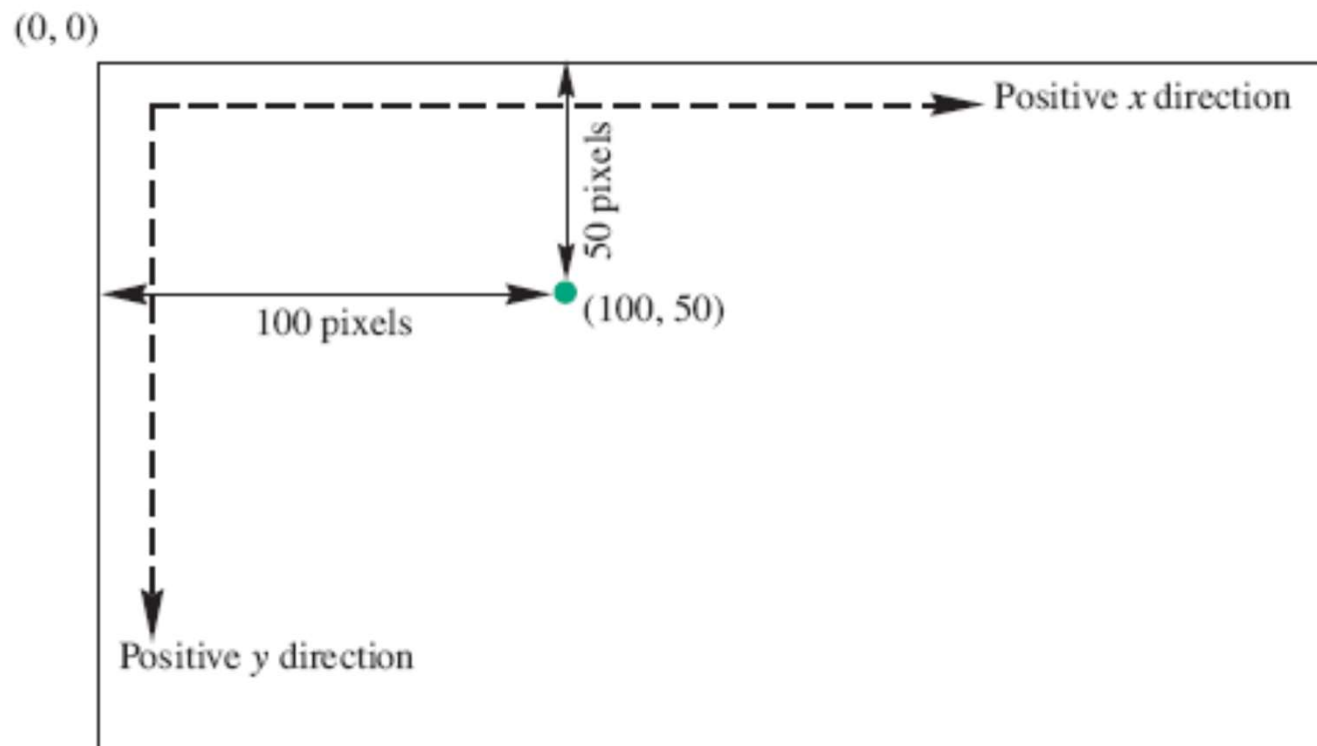
```
    root.getChildren().add(canvas);
    primaryStage.setTitle("HappyFace in JavaFX");
    primaryStage.setScene(scene);
    primaryStage.show();
```

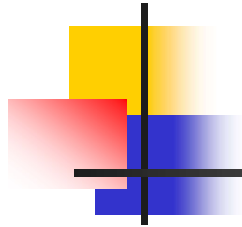
```
    }
}
```



Screen Coordinate System

- Figure 1.6





Screen Coordinate System

- The x-coordinate is the number of pixels from the left.
- The y-coordinate is the number of pixels from the top (not from the bottom).





Drawing Ovals and Circles

- The `drawOval` method

- draws only the outline of the oval.

```
gc.drawOval(100, 50, 90, 50);
```

- The `fillOval` method

- draws a filled-in oval.

```
gc.fillOval(100, 50, 90, 50);
```





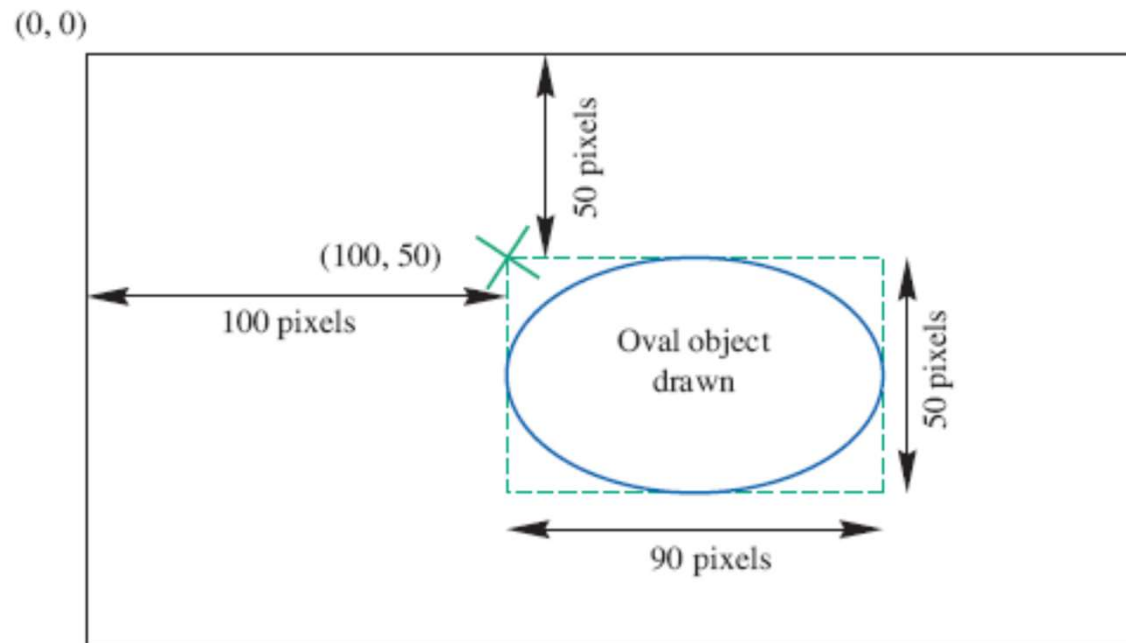
Drawing Ovals and Circles

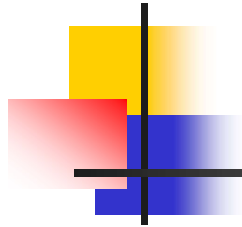
- The **strokeOval** and **fillOval** methods take four arguments.
 - The first two arguments indicate the upper-left corner of an invisible rectangle around the oval.
 - The last two arguments indicate the width and height of the oval.
- A circle is just an oval whose height is the same as its width.




Drawing Ovals and Circles

- Figure 1.7 The Oval Drawn by `gc.strokeOval(100, 50, 90, 50)`





Size and Positions of Figures

- Sizes and positions in a JavaFX program are given in 
- Think of the display surface for the applet as being a two-dimensional grid of individual pixels.





Drawing Arcs

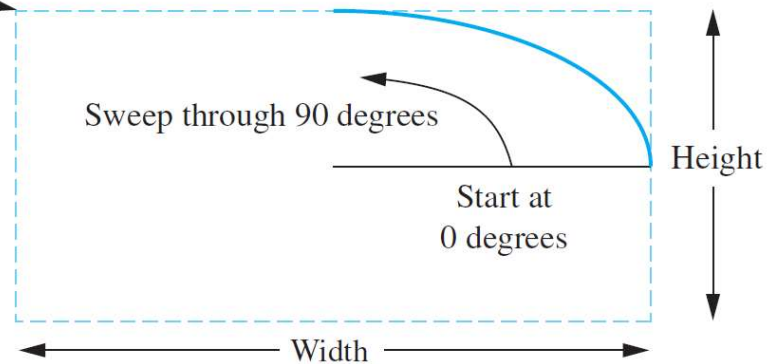
- The `drawArc` method draws an arc.
`drawArc(100, 50, 200, 200, 180, 180, ArcType.OPEN);`
- The `drawArc` method takes **seven arguments**.
 - The first four arguments are the same as the four arguments needed by the `drawRect` method.
 - The next two arguments indicate where the arc starts, and the number of degrees through which it sweeps.
 - 0 degrees is horizontal and to the right.
 - The last argument indicates if the arc is **open or closed**
 - closure type (Round, Chord, Open)



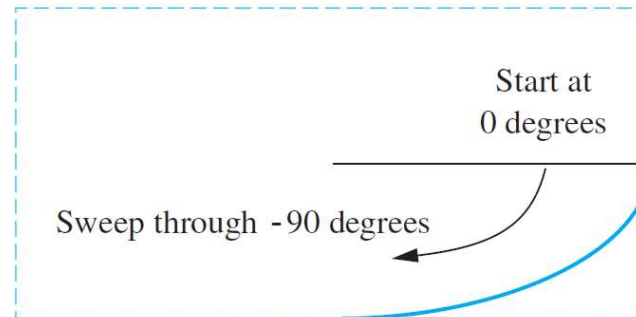
Specifying an Arc

FIGURE 1.8 Specifying an Arc

- (a) `gc.strokeArc(x, y, width, height, 0, 90, ArcType.OPEN);`
(x, y) →

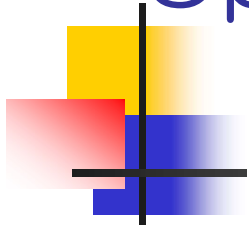


- (b) `gc.strokeArc(x, y, width, height, 0, -90, ArcType.OPEN);`

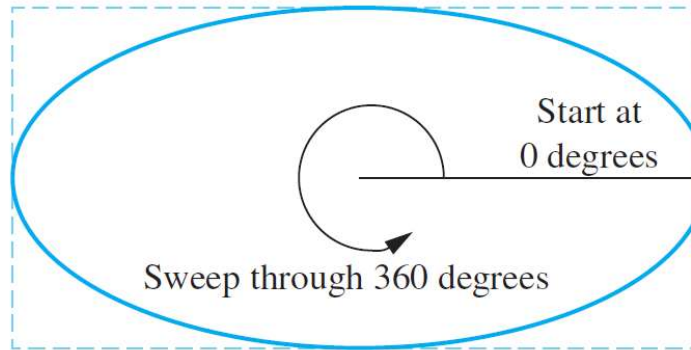


Chord

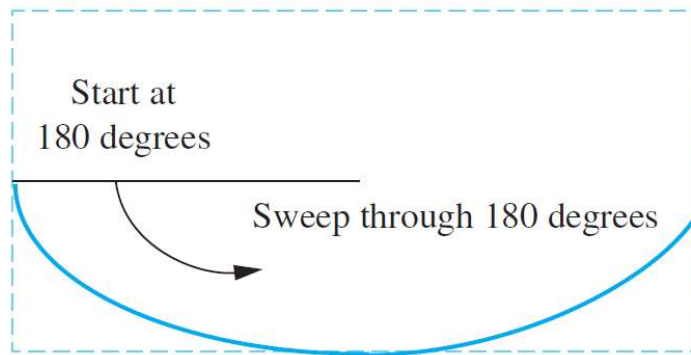
Specifying an Arc



(c) `gc.strokeArc(x, y, width, height, 0, 360, ArcType.OPEN);`



(d) `gc.strokeArc(x, y, width, height, 180, 180, ArcType.OPEN);`



// Listing 1.2

```
import javafx.application.Application;
import javafx.scene.canvas.Canvas;
import javafx.scene.Scene;
import javafx.scene.Group;
import javafx.stage.Stage;
import javafx.scene.canvas.GraphicsContext;
import javafx.scene.shape.ArcType;
```

```
public class HappyFace extends Application
```

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

```
@Override
```

```
public void start(Stage primaryStage) throws Exception
{
```

```
    Group root = new Group();
    Scene scene = new Scene(root);
```

```
    Canvas canvas = new Canvas(400, 300);
    GraphicsContext gc = canvas.getGraphicsContext2D();
```

```
    gc.strokeOval(100, 50, 200, 200);
```

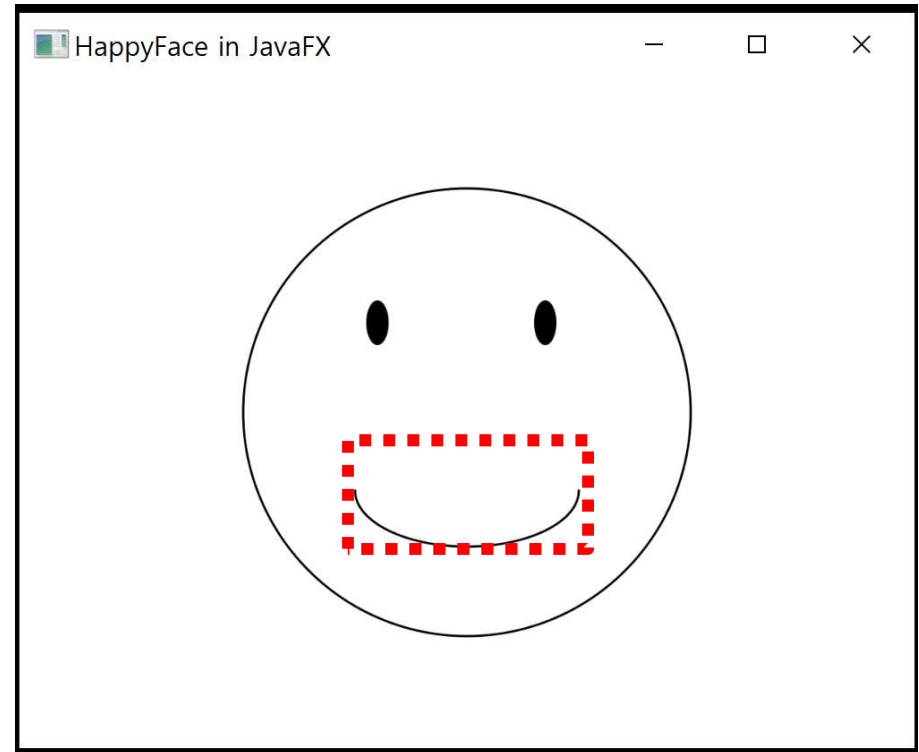
```
    gc.fillOval(155, 100, 10, 20);
```

```
    gc.fillOval(230, 100, 10, 20);
```

```
    gc.strokeArc(150, 160, 100, 50, 180, 180, ArcType.OPEN);
```

```
    root.getChildren().add(canvas);
    primaryStage.setTitle("HappyFace in JavaFX");
    primaryStage.setScene(scene);
    primaryStage.show();
```

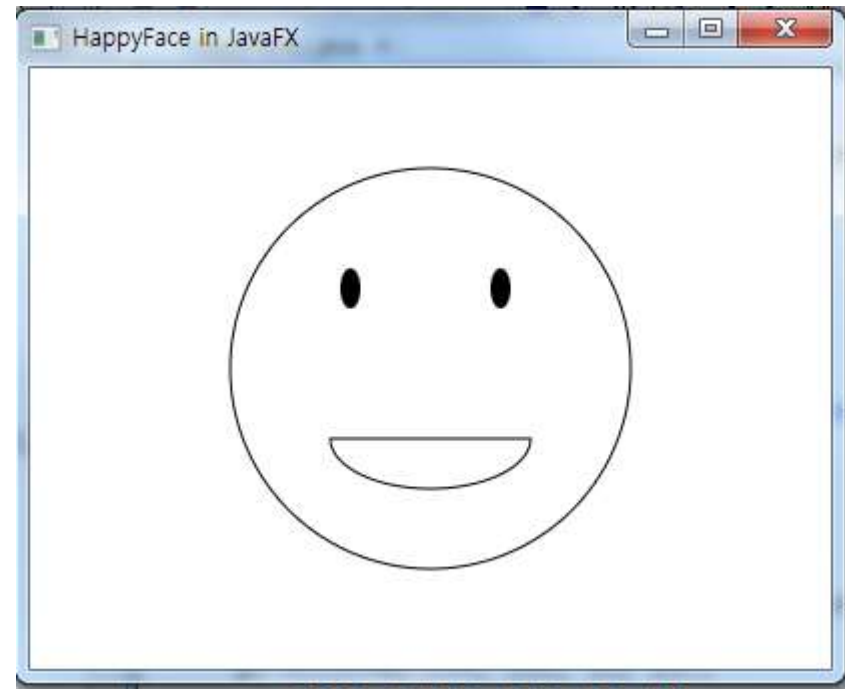
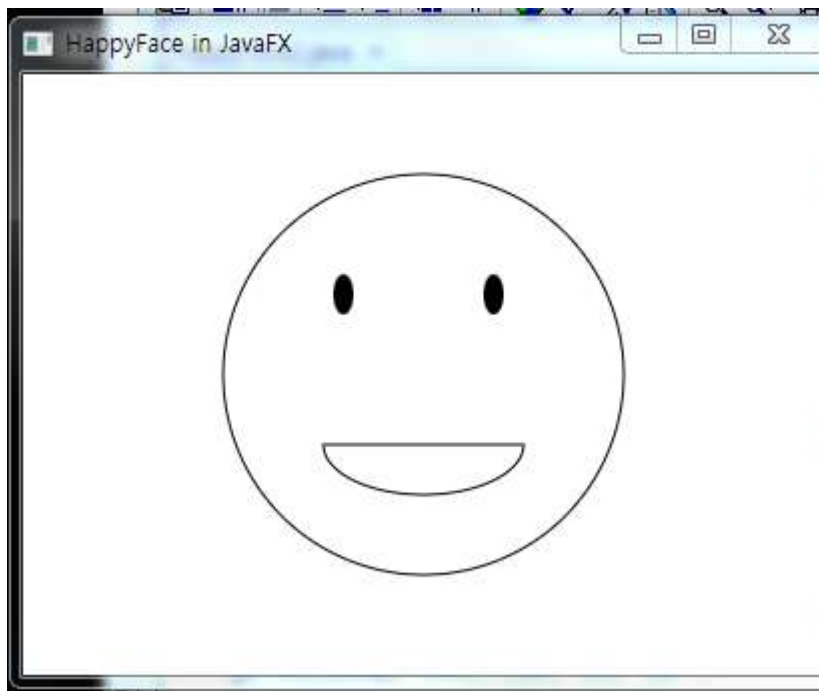
```
    }
}
```

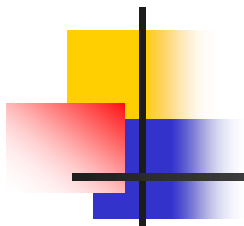




closure type (Round, Chord, Open)

- `ArcType.ROUND` `ArcType.CHORD`





בב
ע

