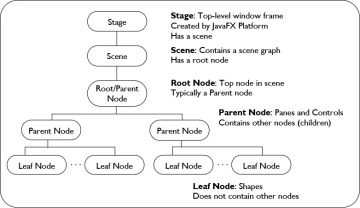
**Parts of a JavaFX Program**



**Stage** – The location where graphic elements will be displayed (usually a window).

**Scene** – A container for the graphic elements, called nodes, which will be displayed (inside the window).

A scene graph must contain at least one node (the root node). When a scene is assigned to a stage, all the scene contents (defined by the scene graph) are displayed on the stage.

**Node** – A graphic element that can be added to the scene. These are organized in a tree-like structure.

**Root Node** – The top node in the scene (often a layout).

**Leaf Node** – Element that cannot contain any other node. (ie. Shape, Canvas, Button, Menu, Control, Chart, etc.)

### Shape subclasses:

Arc, Circle, CubicCurve, Ellipse, Line, Path, Polygon, Polyline, QuadCurve, Rectangle, SVGPath, Text

### Canvas methods:

**getHeight()** – returns a double value of the property height

**getWidth()**– returns a double value of the property width

**setHeight(**double value**)**

**setWidth(**double value**)**

[**getGraphicsContext2D**](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/Canvas.html#getGraphicsContext2D--)()returns the GraphicsContext associated with this Canvas.

#### GraphicsContext methods (selected):

[fillRect()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#fillRect-double-double-double-double-), [fillRoundRect()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#fillRoundRect-double-double-double-double-double-double-), [fillOval()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#fillOval-double-double-double-double-), [fillArc()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#fillArc-double-double-double-double-double-double-javafx.scene.shape.ArcType-), [strokeLine()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokeLine-double-double-double-double-), [strokeRect()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokeRect-double-double-double-double-), [strokeRoundRect()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokeRoundRect-double-double-double-double-double-double-), [strokeOval()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokeOval-double-double-double-double-), [strokeArc()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokeArc-double-double-double-double-double-double-javafx.scene.shape.ArcType-), [clearRect()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#clearRect-double-double-double-double-), [fillPolygon()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#fillPolygon-double:A-double:A-int-), [strokePolygon()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokePolygon-double:A-double:A-int-), [strokePolyline()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokePolyline-double:A-double:A-int-), [fillText()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#fillText-java.lang.String-double-double-), [strokeText()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#strokeText-java.lang.String-double-double-), [drawImage()](https://docs.oracle.com/javase/8/javafx/api/javafx/scene/canvas/GraphicsContext.html#drawImage-javafx.scene.image.Image-double-double-)

**Programming a JavaFX Application**

## JavaFX

## Application

The entry point for JavaFX applications is the Application class. The **Application** class is an abstract class, which means it has one or more abstract methods.

public abstract class Application {...

To create your own JavaFX application your class must **extend** the Application class,

**public class MyGame \_\_extends\_\_\_ Application { ...**

and implement the abstract method, **public void** [**start**](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html#start-javafx.stage.Stage-)([**Stage**](https://docs.oracle.com/javase/8/javafx/api/javafx/stage/Stage.html)**primaryStage**).

**public** **void** start(Stage primaryStage){

//decorate stage

//make nodes to show on stage

//create layout node (root) and add your nodes to it

//make a scene from your layout

//put scene on stage

//show the stage!

}

To start your application you need to call the launch() method, usually this is done from your main. It must not be called more than once or an exception will be thrown.

## Starting the program

**public** **static void** main(String[] args) {

## launch( )

launch(args)

}

## Application

## Life-Cycle

The JavaFX runtime does the following, in order, whenever an application is launched:

1. Constructs an instance of the specified \_\_\_**Application**\_\_\_\_ class

## start( )

1. Calls the [\_init()\_\_](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html#init--) method
2. Calls the [\_\_start(Stage stage)\_\_\_\_](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html#start-javafx.stage.Stage-) method
3. Waits for the application to finish, which happens when either of the following occur:
   * the application calls [\_\_Platform.exit()\_](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Platform.html#exit--)
   * the last window has been closed and the implicitExit attribute on Platform is true
4. Calls the [\_\_stop()\_\_\_\_](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html#stop--) method

*Note that the*start*method is abstract and must be overridden. The*init*and*stop*methods have concrete implementations that do nothing.*

Calling [Platform.exit()](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Platform.html" \l "exit--) is the preferred way to explicitly terminate a JavaFX Application.

Directly calling [System.exit(int)](https://docs.oracle.com/javase/8/docs/api/java/lang/System.html?is-external=true" \l "exit-int-" \o "class or interface in java.lang) is an acceptable alternative, but doesn't allow the Application [stop()](https://docs.oracle.com/javase/8/javafx/api/javafx/application/Application.html#stop--) method to run.