# CREATING A JAVAFX APPLICATION

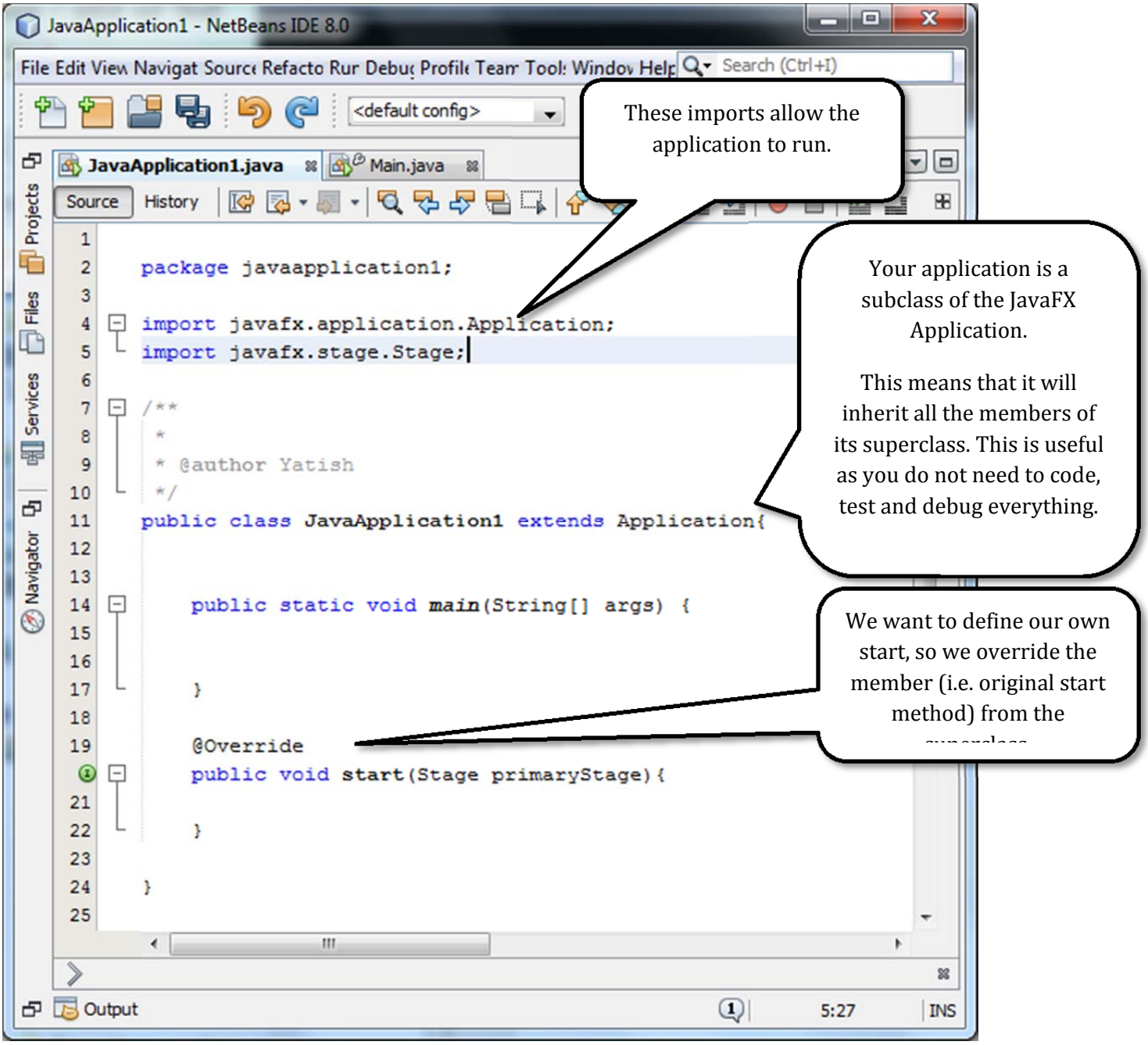
JavaFX is a set of graphics and media packages provided by Oracle. It allows developers to create applications that make use of a GUI. JAvaFX has superseded Swing.

## SETTING UP THE BASIC APPLICATION STRUCTURE

Although you can generate the basic structure of a JavaFX project using a wizard, it is more useful to create an application from scratch.

Create a standard Java application in your IDE.

Add in the code annotated below.




From yatishparmar.com

## SETTING UP THE WINDOW

This is our new start method.

```
21      @Override
        public void start(Stage primaryStage){
23          StackPane stack = new StackPane();
24
25          Scene myScene = new Scene(stack, 800, 400);
26
27          primaryStage.setTitle("Hello World");
28          primaryStage.setScene(myScene);
29          primaryStage.show();
30
31      }
```

In order to remove errors you will have to import `javafx.scene.Scene` and `javafx.stage.Stage`.

### *JAVAFX LAYOUTS*

```
23          StackPane stack = new StackPane();
```

Every stage will use a root node to manage its content. Here we will use a StackPane to organise our content. There are eight built-in layouts than can be used instead of Group in JavaFX:

1) BorderPane *used to create a window divided into to, bottom, left, right and centre areas.*
2) HBox *used to line up content horizontally.*
3) VBox *used to line up content vertically.*
4) StackPane *used to stack content items on top of each other.*
5) GridPane *used to create a tabular layout.*
6) FlowPane *used to create a layout that can flow either horizontally or vertically.*
7) TilePane *is similar to FlowPane, but all nodes are the same dimensions.*
8) AnchorPane *used to create a layout where nodes are anchored to the sides or centre of the layout.*

### *THE SCENE*

```
25          Scene myScene = new Scene(stack, 800, 400);
```

The Scene class is the container for all content in the scene on screen. There are different constructors available. You must specify the root node. This can be a group, or one of the eight layouts discussed above. The 800 and 400 refer to the width (x) and height (y) of the scene.

Try running the application now, nothing happens. That's because the main method needs to be told to launch the application.

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

    launch(args);

}
```

This passes the command-line arguments when the application is run to the JavaFX launch method.

If you run the application now, you should see a blank window called Hello World.

From [yatishparmar.com](yatishparmar.com)

## ADDING CONTENT

JavaApplication1 - NetBeans IDE 8.0

```
package javaapplication1;

import javafx.application.Application;
import javafx.scene.Scene;
import javafx.scene.control.Label;
import javafx.scene.layout.StackPane;
import javafx.stage.Stage;

/**
 *
 * @author Yatish
 */
public class JavaApplication1 extends Application{


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

    @Override
    public void start(Stage primaryStage){
        StackPane stack = new StackPane();

        Scene myScene = new Scene(stack, 800, 400);

        Label lblText = new Label();
        lblText.setText("Hello World");
        stack.getChildren().add(lblText);

        primaryStage.setTitle("Hello World");
        primaryStage.setScene(myScene);
        primaryStage.show();

    }

}
```

Import for the **Label** control.

New code to create a label that says Hello World and then add it to the StackPane.

Run your application to see what happens.