



# JavaFX

**tutorialspoint**

SIMPLY EASY LEARNING

[www.tutorialspoint.com](http://www.tutorialspoint.com)



<https://www.facebook.com/tutorialspointindia>



<https://twitter.com/tutorialspoint>

## About the Tutorial

---

JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc.

To develop **GUI Applications** using Java programming language, the programmers rely on libraries such as **Advanced Windowing Toolkit** and **Swings**. After the advent of JavaFX, these Java programmers can now develop GUI applications effectively with rich content.

In this tutorial, we will discuss all the necessary elements of JavaFX that you can use to develop effective Rich Internet Applications.

## Audience

---

This tutorial has been prepared for beginners who want to develop Rich Internet Applications using JavaFX.

## Prerequisites

---

For this tutorial, it is assumed that the readers have a prior knowledge of Java programming language.

## Copyright & Disclaimer

---

© Copyright 2016 by Tutorials Point (I) Pvt. Ltd.

All the content and graphics published in this e-book are the property of Tutorials Point (I) Pvt. Ltd. The user of this e-book is prohibited to reuse, retain, copy, distribute or republish any contents or a part of contents of this e-book in any manner without written consent of the publisher.

We strive to update the contents of our website and tutorials as timely and as precisely as possible, however, the contents may contain inaccuracies or errors. Tutorials Point (I) Pvt. Ltd. provides no guarantee regarding the accuracy, timeliness or completeness of our website or its contents including this tutorial. If you discover any errors on our website or in this tutorial, please notify us at [contact@tutorialspoint.com](mailto:contact@tutorialspoint.com)

## Table of Contents

---

About the Tutorial.....	i
Audience.....	i
Prerequisites.....	i
Copyright & Disclaimer .....	i
Table of Contents.....	ii
<b>1. JAVA FX – OVERVIEW.....</b>	<b>1</b>
What is JavaFX? .....	2
Need for JavaFX .....	2
Features of JavaFX .....	2
History of JavaFX.....	3
<b>2. JAVA FX – ENVIRONMENT .....</b>	<b>4</b>
Installing Java8.....	4
Setting up the Path for Windows .....	10
Setting NetBeans Environment of JavaFX.....	11
Installing JavaFX in Eclipse .....	28
<b>3. JAVA FX – ARCHITECTURE.....</b>	<b>39</b>
Scene Graph.....	40
Prism.....	41
GWT (Glass Windowing Toolkit).....	41
Quantum Toolkit.....	41
WebView .....	41
Media Engine .....	42

4.	JAVAFX – APPLICATION.....	43
	JavaFX Application Structure.....	43
	Creating a JavaFX Application .....	45
	Lifecycle of JavaFX Application.....	48
	Example 1 – Creating an Empty Window (Stage).....	49
	Example 2 – Drawing a Straight Line .....	52
	Example 3 – Displaying Text .....	56
5.	JAVAFX – 2D SHAPES .....	62
	2D Shape.....	62
	Creating a 2D Shape .....	62
	2D Shapes – Line .....	66
	2D Shapes – Rectangle .....	70
	Rounded Rectangle.....	75
	2D Shapes – Circle.....	78
	2D Shapes – Ellipse.....	83
	2D Shapes – Polygon .....	88
	2D Shapes – Polyline .....	94
	2D Shapes – QuadCurve .....	98
	2D Shapes – CubicCurve.....	104
	2D Shapes – Arc .....	110
	2D Shapes – SVGPath .....	116
	Drawing More Shapes Through Path Class .....	121
	Example – Drawing a Complex Path .....	124
	PathElement – Line .....	129
	PathElement – Horizontal Line .....	134
	PathElement – Vertical Line .....	139

PathElement – Quadratic Curve .....	145
PathElement – Cubic Curve .....	151
PathElement – Arc .....	156
Properties of 2D Objects .....	162
Operations on 2D Objects .....	167
6. JAVAFX – TEXT .....	177
Creating a Text Node.....	177
Position and Font of the Text .....	179
Stroke and Color .....	182
Applying Decorations to Text .....	185
7. JAVAFX – EFFECTS.....	189
Applying Effects to a Node .....	189
Color Adjust Effect .....	193
Color Input Effect .....	196
Image Input Effect.....	199
Blend Effect.....	202
Bloom Effect .....	208
Glow Effect .....	211
Box Blur Effect .....	213
Gaussian Blur Effect .....	216
Motion Blur Effect.....	219
Reflection Effect.....	222
Sepia Tone Effect .....	225
Shadow Effect .....	228
Drop Shadow Effect .....	231
Inner Shadow Effect.....	235

Lighting Effect (Default Source).....	239
Lighting Effect (Distant Source) .....	242
Lighting Effect (Spot Light as Source) .....	246
8. JAVA FX – TRANSFORMATIONS.....	255
Rotation.....	255
Scaling.....	258
Translation.....	262
Shearing.....	265
Multiple Transformations .....	268
Transformations on 3D Objects.....	271
9. JAVA FX – ANIMATIONS.....	274
Fade Transition .....	274
Fill Transition .....	277
Rotate Transition .....	279
Scale Transition.....	282
Stroke Transition.....	284
Translate Transition .....	286
Sequential Transition .....	289
Parallel Transition .....	293
Pause Transition .....	297
PathTransition .....	300
10. JAVA FX – COLORS.....	307
Applying Color to the Nodes .....	307
Applying Image Pattern to the Nodes .....	310
Applying Linear Gradient Pattern.....	313

Applying Radial Gradient Pattern.....	316
11. JAVA FX – IMAGES .....	320
Loading an Image .....	320
Multiple Views of an Image .....	323
Writing Pixels.....	325
12. JAVA FX – 3D SHAPES .....	329
Creating a 3D Shape .....	329
3D Shapes – Box.....	331
3D Shapes – Cylinder.....	335
3D Shapes – Sphere.....	340
Properties of 3D Objects .....	345
13. JAVA FX – EVENT HANDLING .....	358
Types of Events .....	358
Events in JavaFX.....	358
Event Handling.....	359
Phases of Event Handling in JavaFX.....	359
Adding and Removing Event Filter .....	361
Event Handling Example .....	361
Adding and Removing Event Handlers .....	364
Using Convenience Methods for Event Handling.....	369
14. JAVA FX – CONTROLS (UI CONTROLS).....	375
15. JAVA FX – CHARTS .....	386
Creating a Chart .....	386
Charts – Pie Chart.....	390
Charts – Line Chart .....	397

Charts – Area Chart .....	404
Charts – Bar Chart .....	410
Charts – Bubble Chart .....	417
Charts – Scatter Chart .....	424
Charts – StackedArea Chart.....	430
Charts – Stacked Bar Chart.....	439
16. JAVA FX – LAYOUT PANES (CONTAINERS) .....	448
HBox .....	451
VBox .....	453
BorderPane.....	456
StackPane .....	458
FlowPane .....	461
AnchorPane .....	464
TextFlow .....	467
TilePane .....	470
GridPane.....	473
17. JAVA FX – CSS .....	477
CSS in JavaFX.....	477



# 1. JAVAFX – OVERVIEW

**Rich Internet Applications** are those web applications which provide similar features and experience as that of desktop applications. They offer a better visual experience when compared to the normal web applications to the users. These applications are delivered as browser plug-ins or as a virtual machine and are used to transform traditional static applications into more enhanced, fluid, animated and engaging applications.

Unlike traditional desktop applications, RIA's don't require to have any additional software to run. As an alternative, you should install software such as ActiveX, Java, Flash, depending on the Application.

In an RIA, the graphical presentation is handled on the client side, as it has a plugin that provides support for rich graphics. In a nutshell, data manipulation in an RIA is carried out on the server side, while related object manipulation is carried out on the client side.

We have three main technologies using which we can develop an RIA. These include the following:

- Adobe Flash
- Microsoft Silverlight
- JavaFX

## Adobe Flash

This software platform is developed by Adobe Systems and is used in creating Rich Internet Applications. Along with these, you can also build other Applications such as Vector, Animation, Browser Games, Desktop Applications, Mobile Applications and Games, etc.

This is the most commonly used platform for developing and executing RIA's with a desktop browser penetration rate of 96%.

## Microsoft Silverlight

Just like Adobe flash, Microsoft Silverlight is also a software application framework for developing as well as executing Rich Internet Applications. Initially this framework was used for streaming media. The present versions support multimedia, graphics, and animation as well.

This platform is rarely used with a desktop browser penetration rate of 66%.

## JavaFX

JavaFX is a Java library using which you can develop Rich Internet Applications. By using Java technology, these applications have a browser penetration rate of 76%.

## What is JavaFX?

---

JavaFX is a Java library used to build Rich Internet Applications. The applications written using this library can run consistently across multiple platforms. The applications developed using JavaFX can run on various devices such as Desktop Computers, Mobile Phones, TVs, Tablets, etc.

To develop **GUI Applications** using Java programming language, the programmers rely on libraries such as **Advanced Windowing Toolkit** and **Swings**. After the advent of JavaFX, these Java programmers can now develop GUI applications effectively with rich content.

## Need for JavaFX

---

To develop **Client Side Applications** with rich features, the programmers used to depend on various libraries to add features such as Media, UI controls, Web, 2D and 3D, etc. JavaFX includes all these features in a single library. In addition to these, the developers can also access the existing features of a Java library such as **Swings**.

JavaFX provides a rich set of graphics and media API's and it leverages the modern **Graphical Processing Unit** through hardware accelerated graphics. JavaFX also provides interfaces using which developers can combine graphics animation and UI control.

One can use JavaFX with JVM based technologies such as Java, Groovy and JRuby. If developers opt for JavaFX, there is no need to learn additional technologies, as prior knowledge of any of the above-mentioned technologies will be good enough to develop RIA's using JavaFX.

## Features of JavaFX

---

Following are some of the important features of JavaFX –

- **Written in Java:** The JavaFX library is written in Java and is available for the languages that can be executed on a JVM, which include – **Java, Groovy** and **JRuby**. These JavaFX applications are also platform independent.
- **FXML:** JavaFX features a language known as FXML, which is a HTML like declarative markup language. The sole purpose of this language is to define a user Interface.
- **Scene Builder:** JavaFX provides an application named Scene Builder. On integrating this application in IDE's such as Eclipse and NetBeans, the users can access a drag and drop design interface, which is used to develop FXML applications (just like Swing Drag & Drop and DreamWeaver Applications).

- **Swing Interoperability:** In a JavaFX application, you can embed Swing content using the **Swing Node** class. Similarly, you can update the existing Swing applications with JavaFX features like embedded web content and rich graphics media.
- **Built-in UI controls:** JavaFX library caters UI controls using which we can develop a full-featured application.
- **CSS like Styling:** JavaFX provides a CSS like styling. By using this, you can improve the design of your application with a simple knowledge of CSS.
- **Canvas and Printing API:** JavaFX provides Canvas, an immediate mode style of rendering API. Within the package **javafx.scene.canvas** it holds a set of classes for canvas, using which we can draw directly within an area of the JavaFX scene. JavaFX also provides classes for Printing purposes in the package **javafx.print**.
- **Rich set of API's:** JavaFX library provides a rich set of API's to develop GUI applications, 2D and 3D graphics, etc. This set of API's also includes capabilities of Java platform. Therefore, using this API, you can access the features of Java languages such as Generics, Annotations, Multithreading, and Lambda Expressions. The traditional Java Collections library was enhanced and concepts like observable lists and maps were included in it. Using these, the users can observe the changes in the data models.
- **Integrated Graphics library:** JavaFX provides classes for **2d** and **3d** graphics.
- **Graphics pipeline:** JavaFX supports graphics based on the Hardware-accelerated graphics pipeline known as Prism. When used with a supported Graphic Card or GPU it offers smooth graphics. In case the system does not support graphic card then prism defaults to the software rendering stack.

## History of JavaFX

---

JavaFX was originally developed by **Chris Oliver**, when he was working for a company named **See Beyond Technology Corporation**, which was later acquired by **Sun Microsystems** in the year 2005.

The following points give us more information of this project:

- Initially this project was named as F3 (**Form Follows Functions**) and it was developed with an intention to provide richer interfaces for developing GUI Applications.
- **Sun Microsystems** acquired the See Beyond company in June 2005, it adapted the F3 project as **JavaFX**.
- In the year 2007, JavaFX was announced officially at **Java One**, a world wide web conference which is held yearly.

- In the year 2008, **Net Beans** integrated with JavaFX was available. In the same year, the Java **Standard Development Kit** for JavaFX 1.0 was released.
- In the year 2009, Oracle Corporation acquired Sun Microsystems and in the same year the next version of JavaFX (1.2) was released as well.
- In the year 2010, JavaFX 1.3 came out and in the year 2011 JavaFX 2.0 was released.
- The latest version, JavaFX8, was released as an integral part of Java on 18<sup>th</sup> of March 2014.

## 2. JAVAFX – ENVIRONMENT

From Java8 onwards, the JDK (**Java Development Kit**) includes **JavaFX** library in it. Therefore, to run JavaFX applications, you simply need to install Java8 or later version in your system.

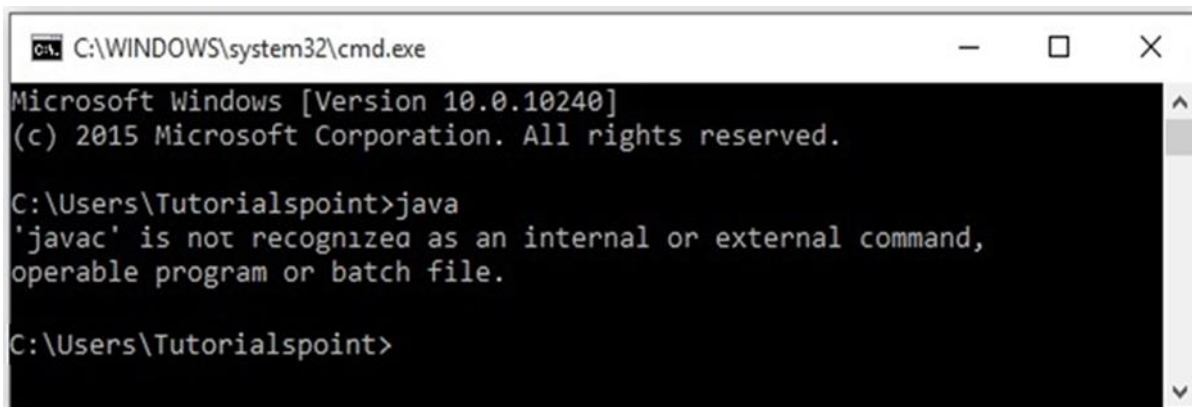
In addition to it, IDE's like Eclipse and NetBeans provide support for JavaFX. This chapter teaches you how to set the environment to run JavaFX Applications in various ways.

### Installing Java8

---

First of all, you will have to verify whether there is Java Installed in your system or not by opening the command prompt and typing the command "Java" in it.

If you haven't installed Java in your system, the command prompt displays the message shown in the following screenshot.



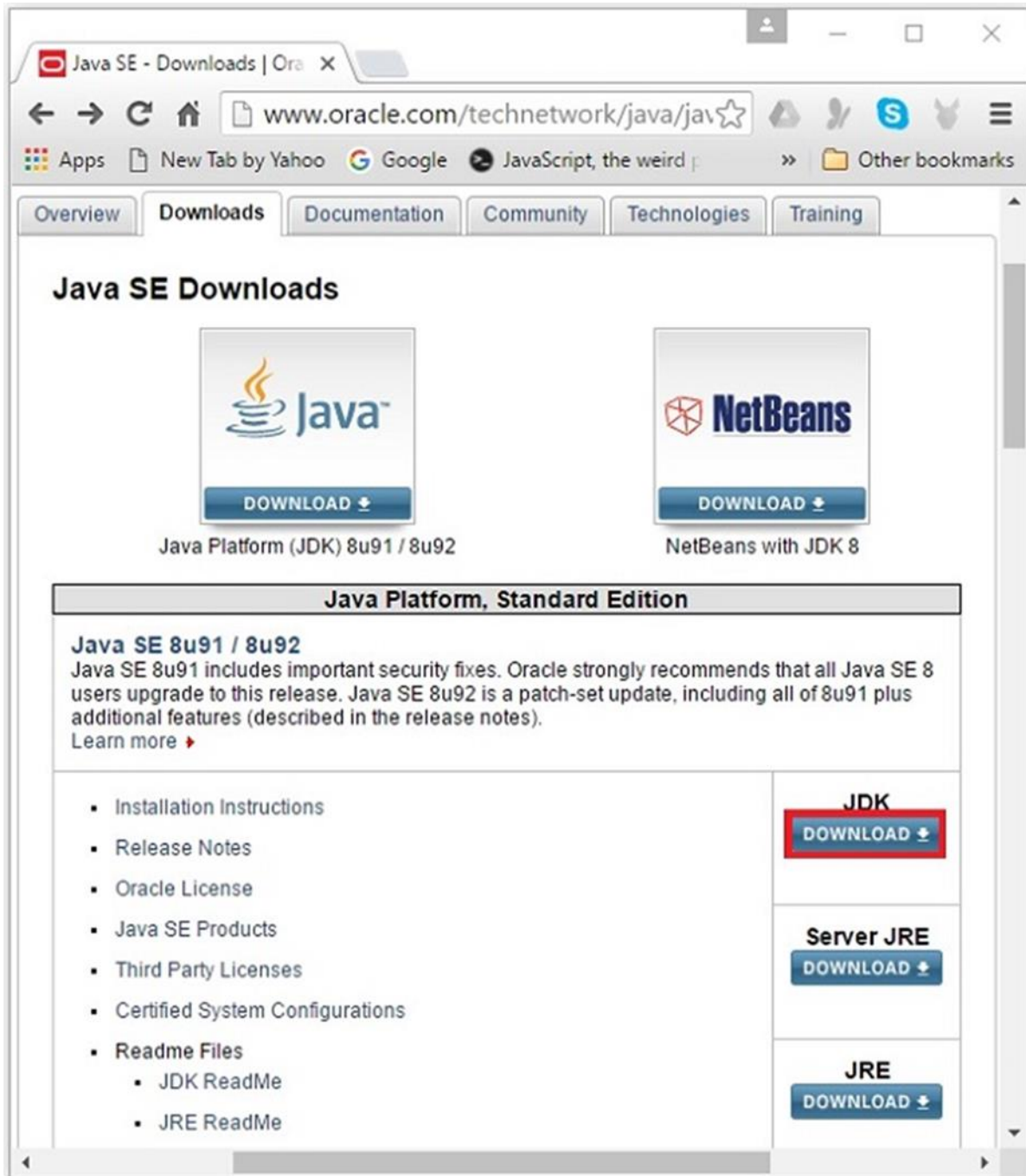
```
C:\WINDOWS\system32\cmd.exe
Microsoft Windows [Version 10.0.10240]
(c) 2015 Microsoft Corporation. All rights reserved.

C:\Users\Tutorialspoint>java
'javac' is not recognized as an internal or external command,
operable program or batch file.

C:\Users\Tutorialspoint>
```

Then install Java by following the steps given below.

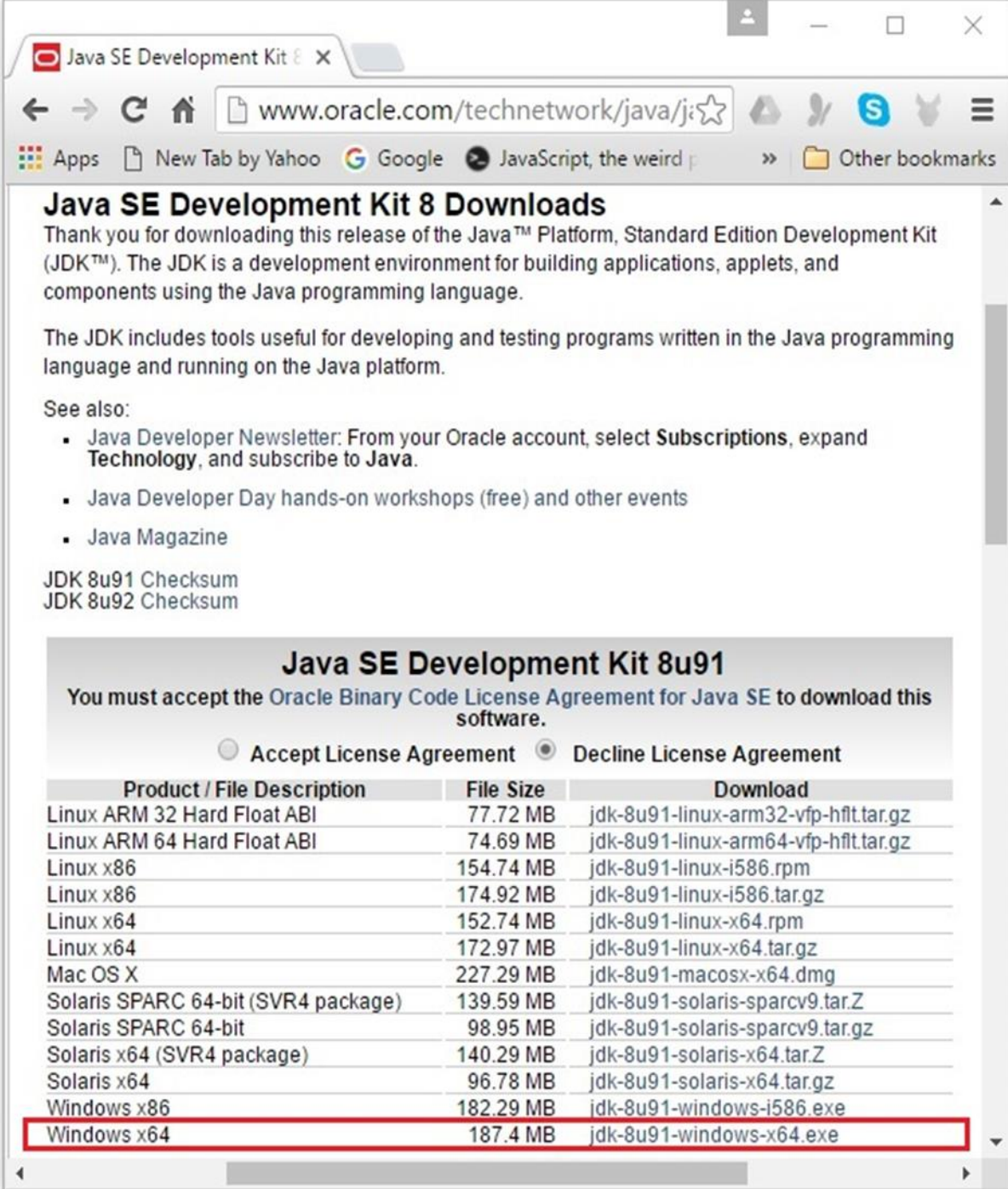
**Step 1:** Visit the [JavaSE Downloads](#) Page, click on the JDK **Download** button as highlighted in the following screenshot.



**Step 2:** On clicking the Download button, you will be redirected to the **Java SE Development Kit 8 Downloads** page. This page provides you links of JDK for various platforms.

Accept the license agreement and download the required software by clicking on its respective link.

For example, if you are working on a windows 64-bit Operating System then you need to download the JDK version highlighted in the following screenshot.



**Java SE Development Kit 8 Downloads**

Thank you for downloading this release of the Java™ Platform, Standard Edition Development Kit (JDK™). The JDK is a development environment for building applications, applets, and components using the Java programming language.

The JDK includes tools useful for developing and testing programs written in the Java programming language and running on the Java platform.

See also:

- Java Developer Newsletter: From your Oracle account, select **Subscriptions**, expand **Technology**, and subscribe to **Java**.
- Java Developer Day hands-on workshops (free) and other events
- Java Magazine

JDK 8u91 Checksum  
JDK 8u92 Checksum

**Java SE Development Kit 8u91**

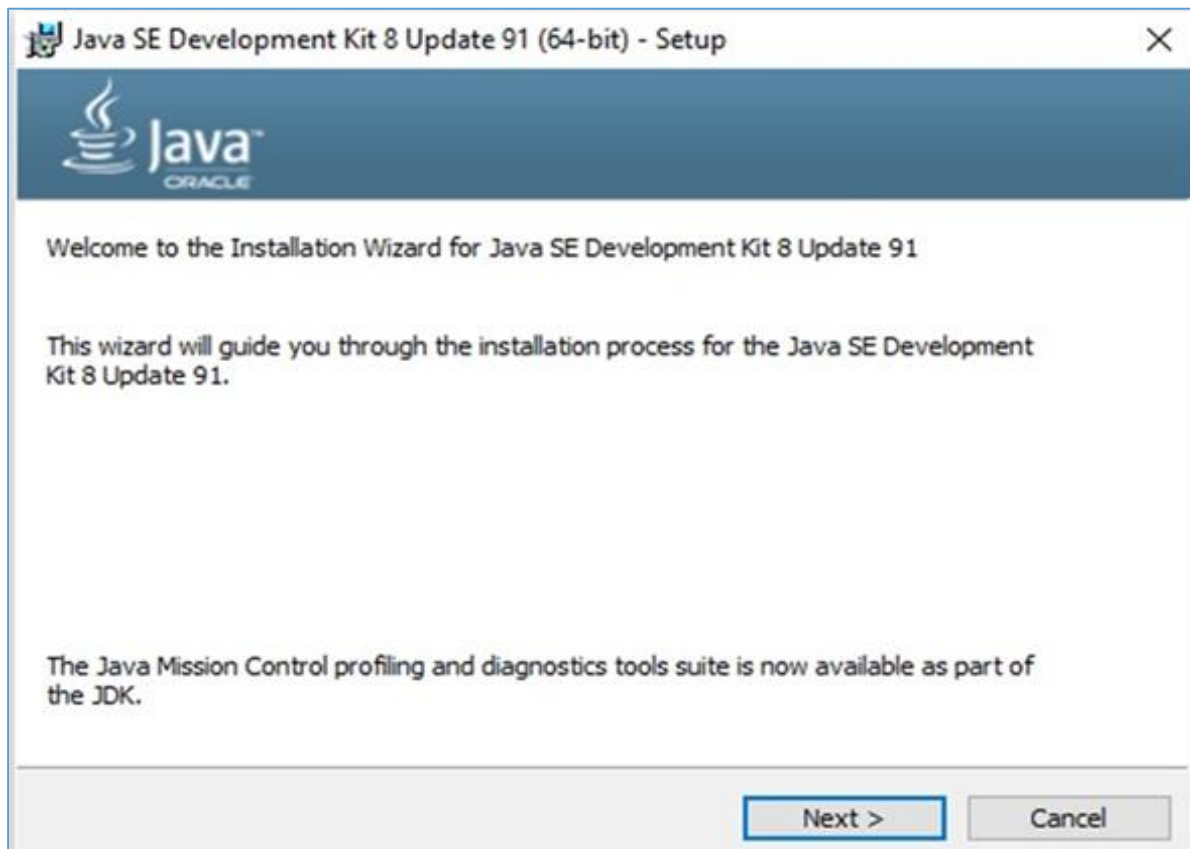
You must accept the [Oracle Binary Code License Agreement for Java SE](#) to download this software.

Accept License Agreement  Decline License Agreement

Product / File Description	File Size	Download
Linux ARM 32 Hard Float ABI	77.72 MB	jdk-8u91-linux-arm32-vfp-hflt.tar.gz
Linux ARM 64 Hard Float ABI	74.69 MB	jdk-8u91-linux-arm64-vfp-hflt.tar.gz
Linux x86	154.74 MB	jdk-8u91-linux-i586.rpm
Linux x86	174.92 MB	jdk-8u91-linux-i586.tar.gz
Linux x64	152.74 MB	jdk-8u91-linux-x64.rpm
Linux x64	172.97 MB	jdk-8u91-linux-x64.tar.gz
Mac OS X	227.29 MB	jdk-8u91-macosx-x64.dmg
Solaris SPARC 64-bit (SVR4 package)	139.59 MB	jdk-8u91-solaris-sparcv9.tar.Z
Solaris SPARC 64-bit	98.95 MB	jdk-8u91-solaris-sparcv9.tar.gz
Solaris x64 (SVR4 package)	140.29 MB	jdk-8u91-solaris-x64.tar.Z
Solaris x64	96.78 MB	jdk-8u91-solaris-x64.tar.gz
Windows x86	182.29 MB	jdk-8u91-windows-i586.exe
Windows x64	187.4 MB	jdk-8u91-windows-x64.exe

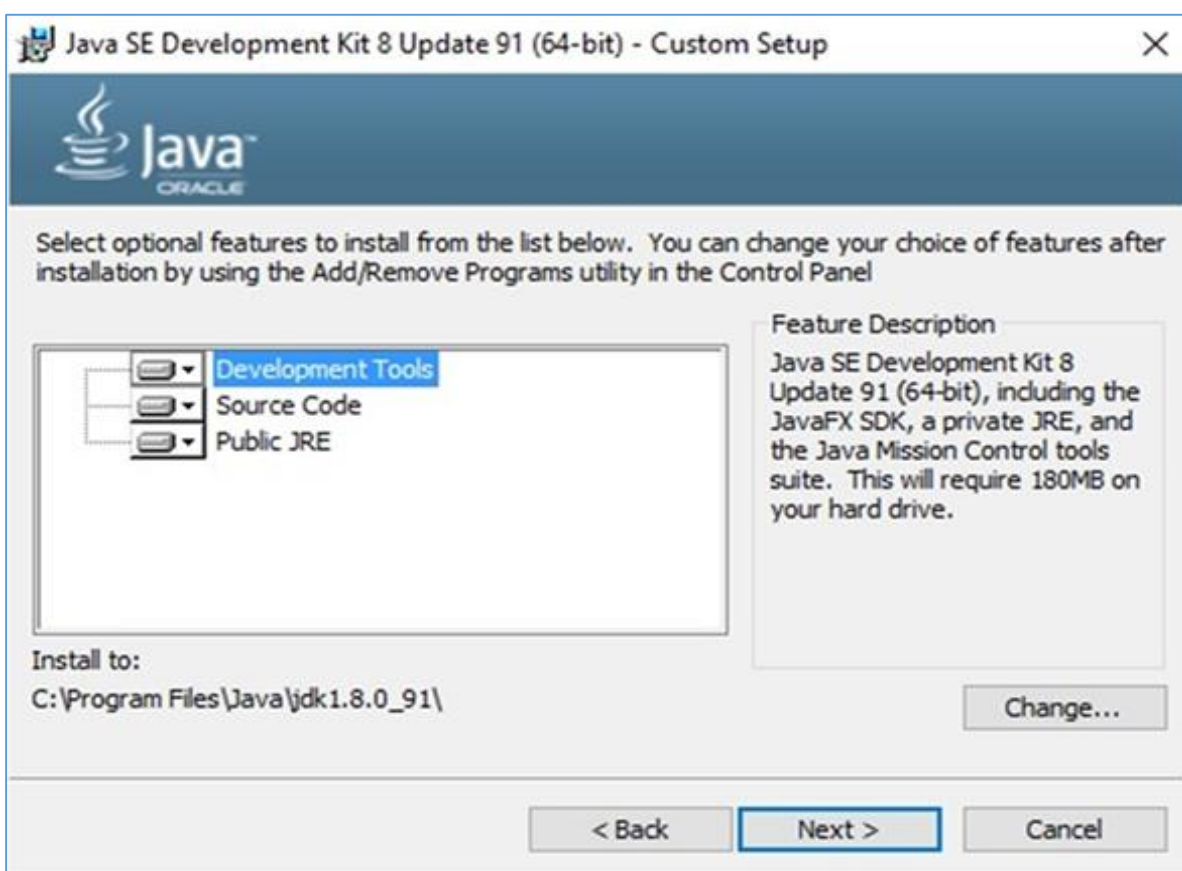
On clicking the highlighted link, the Java8 Development Kit suitable for Windows 64-bit Operating System will be downloaded onto your system.

**Step 3:** Run the downloaded binary executable file to start the installation of JDK8.

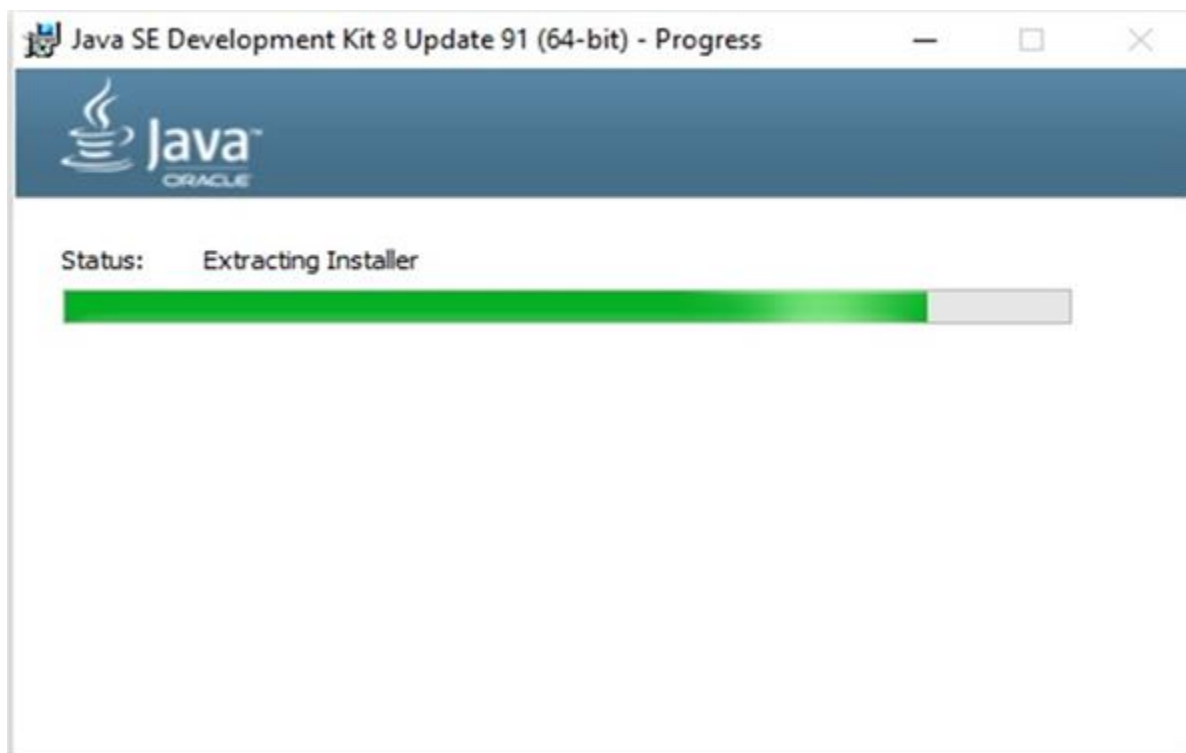




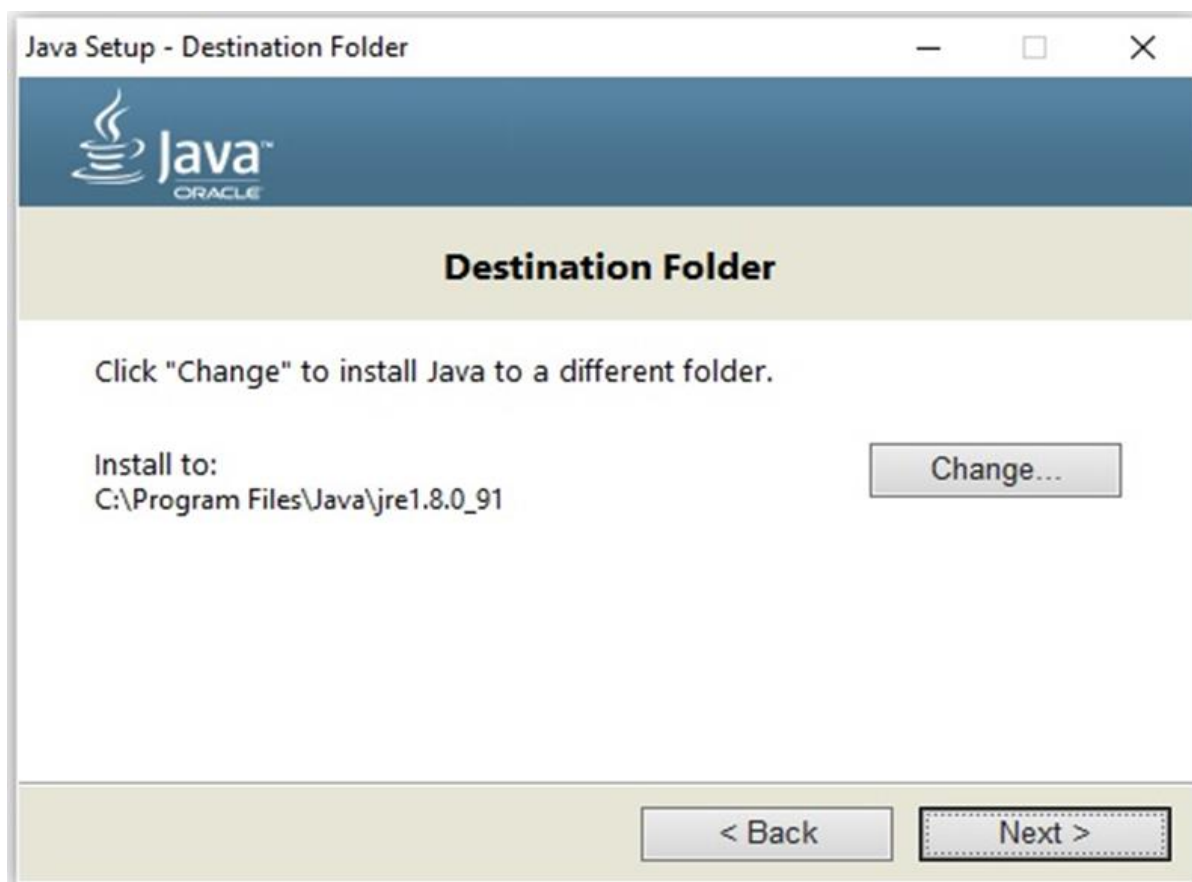
**Step 4:** Choose the Installation Directory.



**Step 5:** On selecting the destination folder and clicking Next, the JavaFX installation process starts displaying the progress bar as shown in the following screenshot.



**Step 6:** Change the installation directory if needed, else keep the default ones and proceed further.



**Step 7:** Finish the installation process by clicking the Close button as shown in the following screenshot.



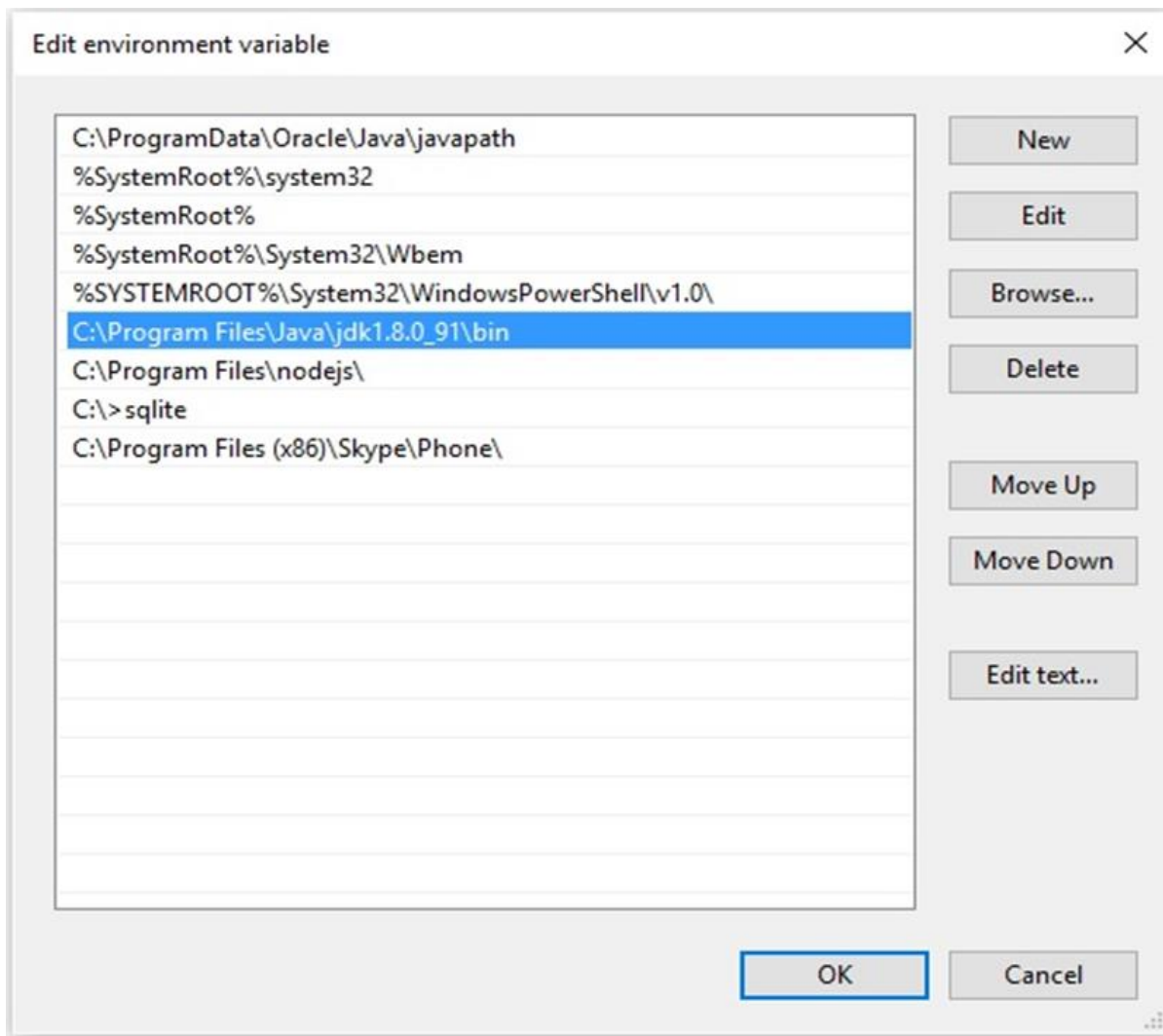
## Setting up the Path for Windows

---

After installing Java, you need to set the path variables. Assume that you have installed Java in **C:\Program Files\java\jdk1.8.0\_91** directory.

Now you can follow the steps that are given below.

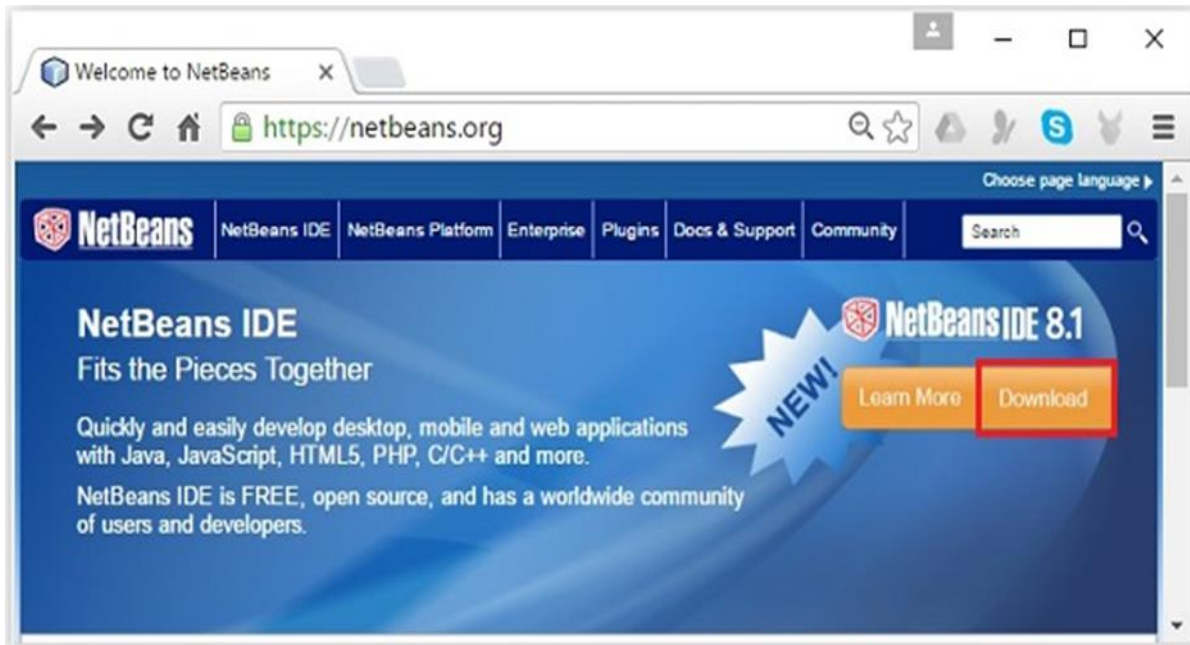
- Right-click on 'My Computer' and select 'Properties'.
- Click on the 'Environment Variables' button under the 'Advanced' tab.
- Now, alter the 'Path' variable so that it also contains the path to the Java executable. For Example, if the path is currently set to 'C:\WINDOWS\SYSTEM32', then change your path to read 'C:\WINDOWS\SYSTEM32; C:\Program Files\java\jdk1.8.0\_91\bin'.



## Setting NetBeans Environment of JavaFX

**NetBeans8** provides inbuilt support for JavaFX. On installing this, you can create a JavaFX application without any additional plugins or JAR files. To set up the NetBeans environment, you will need to follow the steps that are given below.

**Step 1:** Visit the [NetBeans website](https://netbeans.org) and click the Download button in order to download the NetBeans software.



**Step 2:** On clicking **Download**, you will get to the Downloads page of the NetBeans software, which provides NetBeans bundles for various Java applications. Download the NetBeans software for **JavaSE** as shown in the following screenshot.

The screenshot shows the NetBeans IDE 8.1 Download page. The page title is "NetBeans IDE 8.1 Download" and it includes a navigation menu with options like "NetBeans IDE", "NetBeans Platform", "Plugins", "Docs & Support", "Community", and "Partners". The page also features a search bar and a "Choose page language" dropdown.

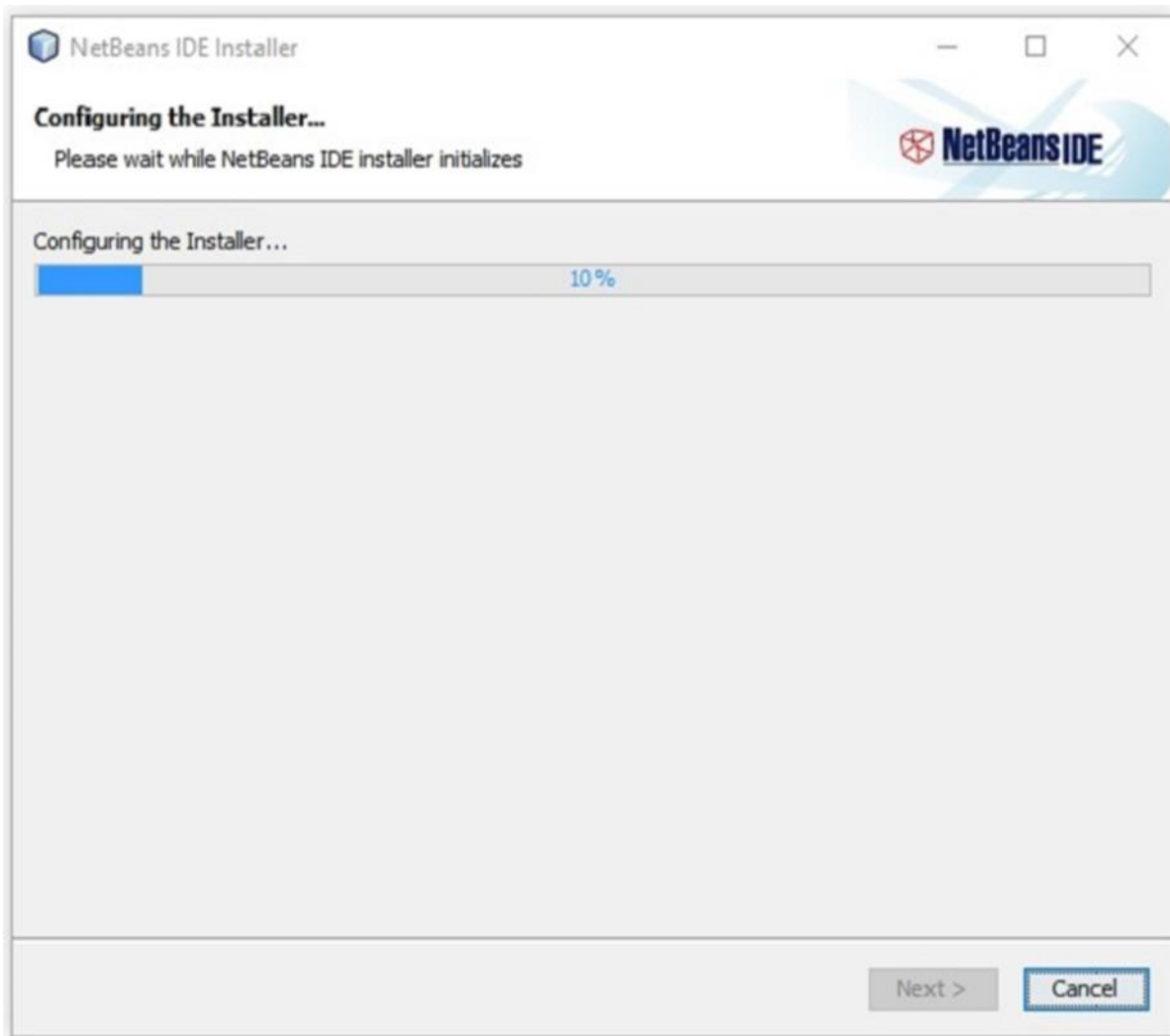
Below the navigation menu, there is a section for "NetBeans IDE 8.1 Download" with a version selector (8.0.2 | 8.1 | Development | JDK9 Branch | Archive). There is a form for "Email address (optional)" and "Subscribe to newsletters" with checkboxes for "Monthly", "Weekly", and "NetBeans can contact me at this address". The "IDE Language" is set to "English" and the "Platform" is set to "Windows".

The main content is a table titled "NetBeans IDE Download Bundles" showing supported technologies and their availability across different bundles. The table has columns for "Supported technologies", "Java SE", "Java EE", "HTML5/JavaScript", "PHP", "C/C++", and "All".

Supported technologies *	Java SE	Java EE	HTML5/JavaScript	PHP	C/C++	All
NetBeans Platform SDK	•	•				•
Java SE	•	•				•
Java FX	•	•				•
Java EE		•				•
Java ME						•
HTML5/JavaScript		•	•	•		•
PHP			•	•		•
C/C++					•	•
Groovy						•
Java Card™ 3 Connected						•
Bundled servers						
GlassFish Server Open Source Edition 4.1.1		•				•
Apache Tomcat 8.0.27		•				•

At the bottom of the table, there are several "Download" buttons. The "Download" button for the Java SE bundle is highlighted with a red box. Other buttons include "Download" for the Java EE bundle, "Download x86" for the HTML5/JavaScript bundle, "Download x86" for the PHP bundle, "Download x86" for the C/C++ bundle, "Download x64" for the HTML5/JavaScript bundle, "Download x64" for the PHP bundle, "Download x64" for the C/C++ bundle, and a final "Download" button for the "All" bundle.

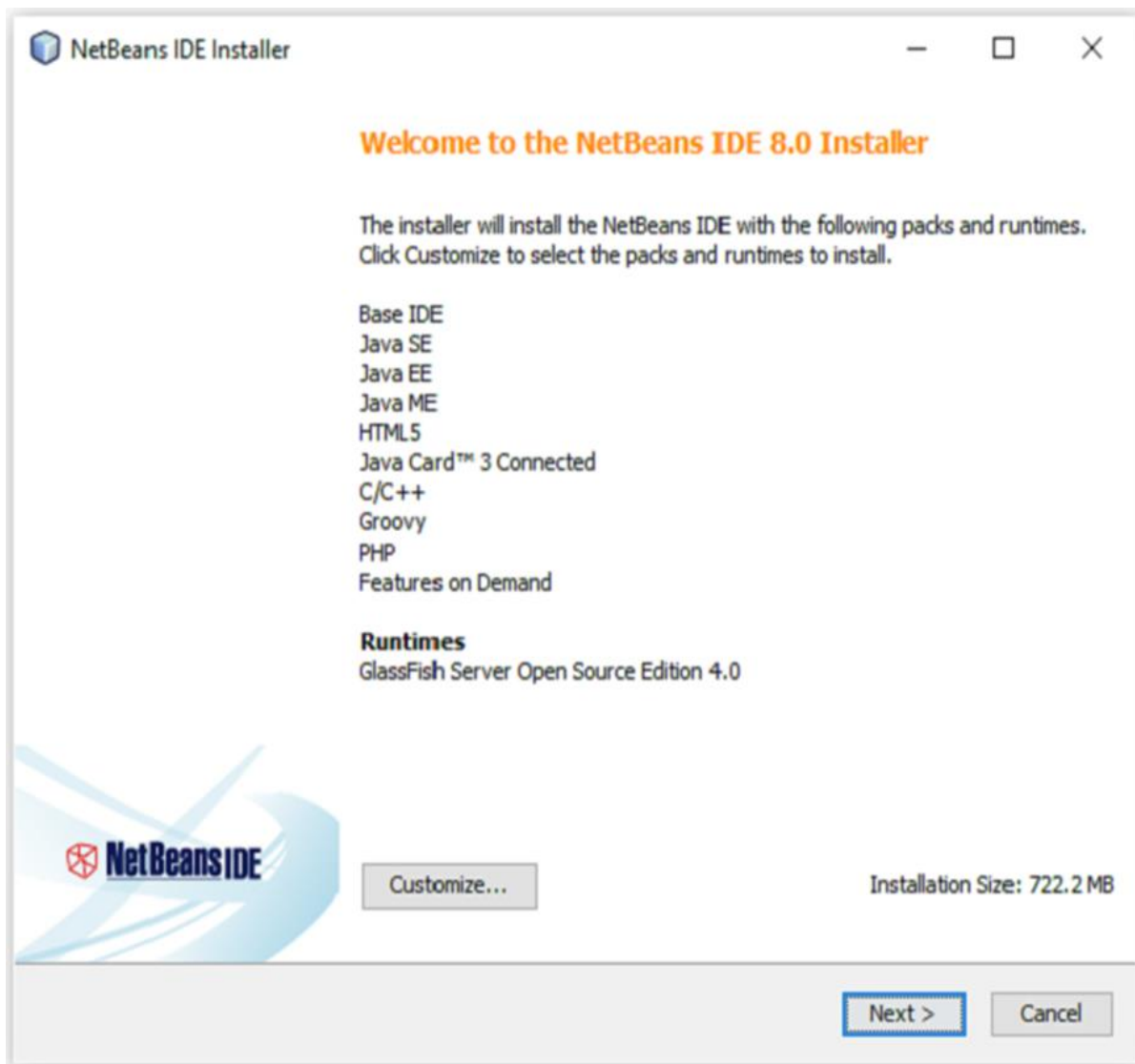
**Step 3:** On clicking this button, a file named **netbeans-8.0-windows.exe** will be downloaded onto your system. Run this file in order to install it. On running this file, a NetBeans installer will start as shown in the following screenshot.



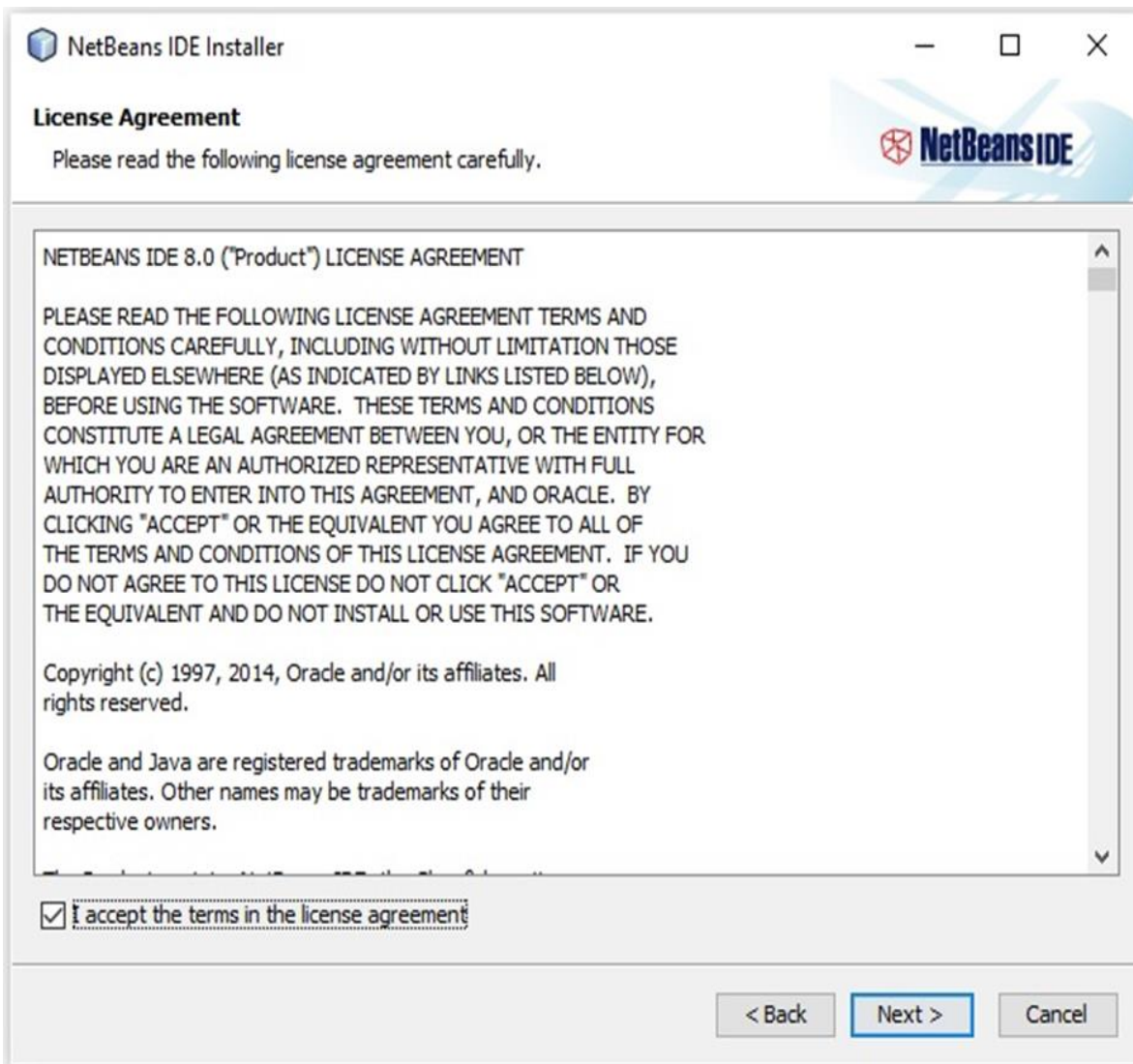
After completion of the configuration, you will see the **Welcome Page** of the installer.



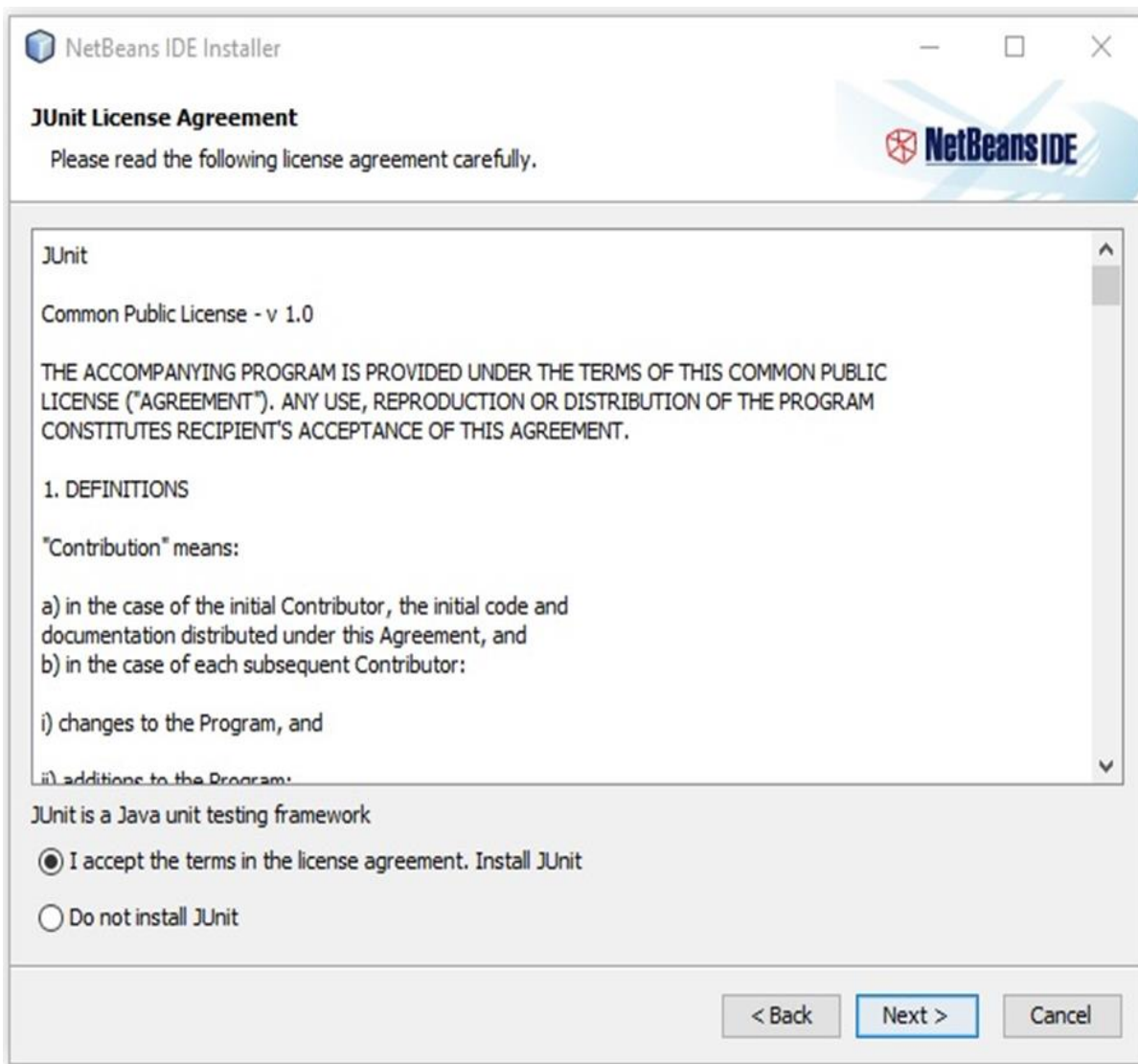
**Step 4:** Click the Next button and proceed with the installation.



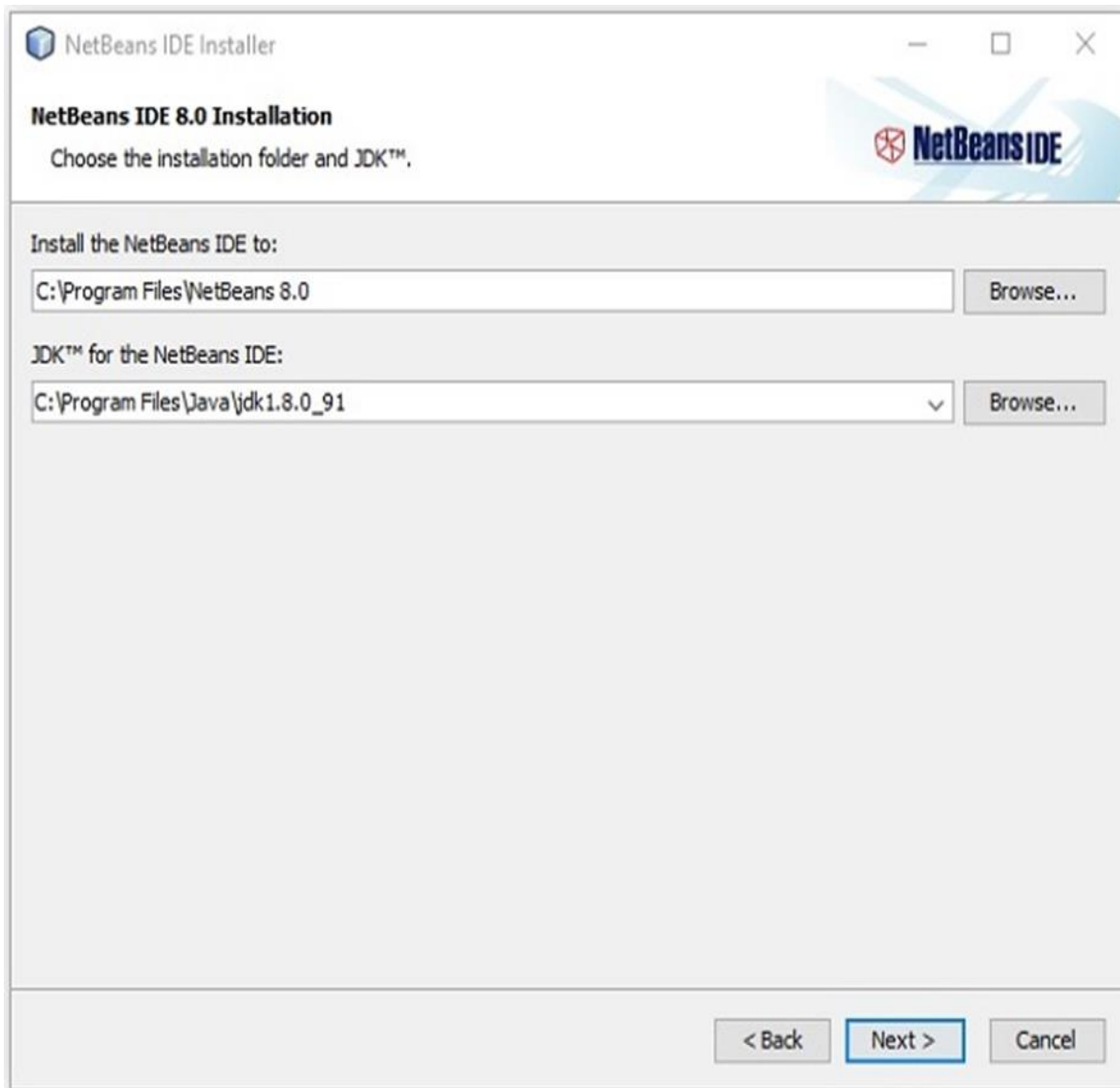
**Step 5:** The next window holds the **NETBEANS IDE 8.0 license agreement**. Read it carefully and accept the agreement by checking the checkbox at "I accept the terms in the license agreement" and then click the **Next** button.



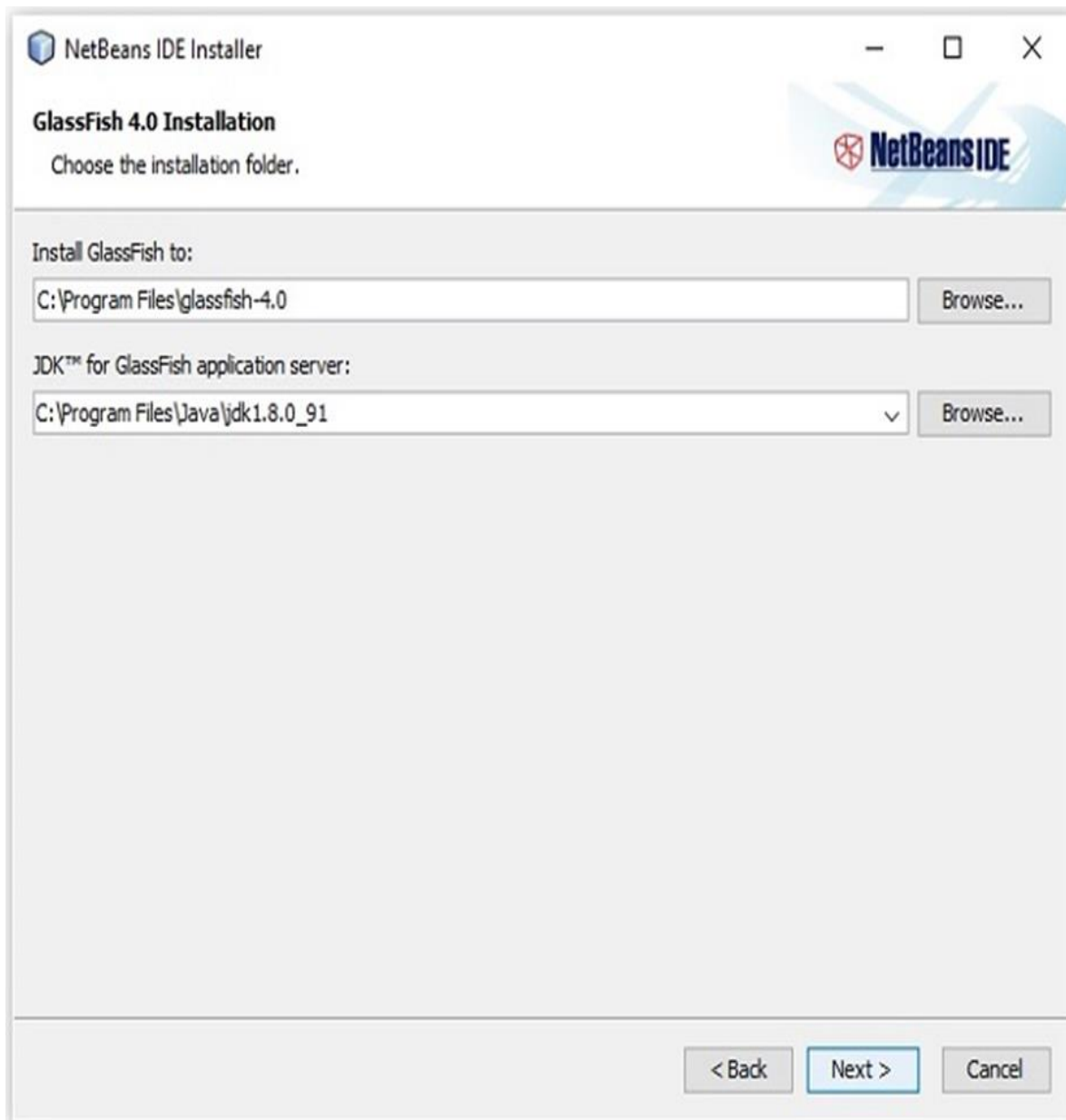
**Step 6:** In the next window, you will encounter the license agreement for **JUnit**, accept it by selecting the radio button at "I accept the terms in the license agreement, Install JUnit" and click on **Next**.



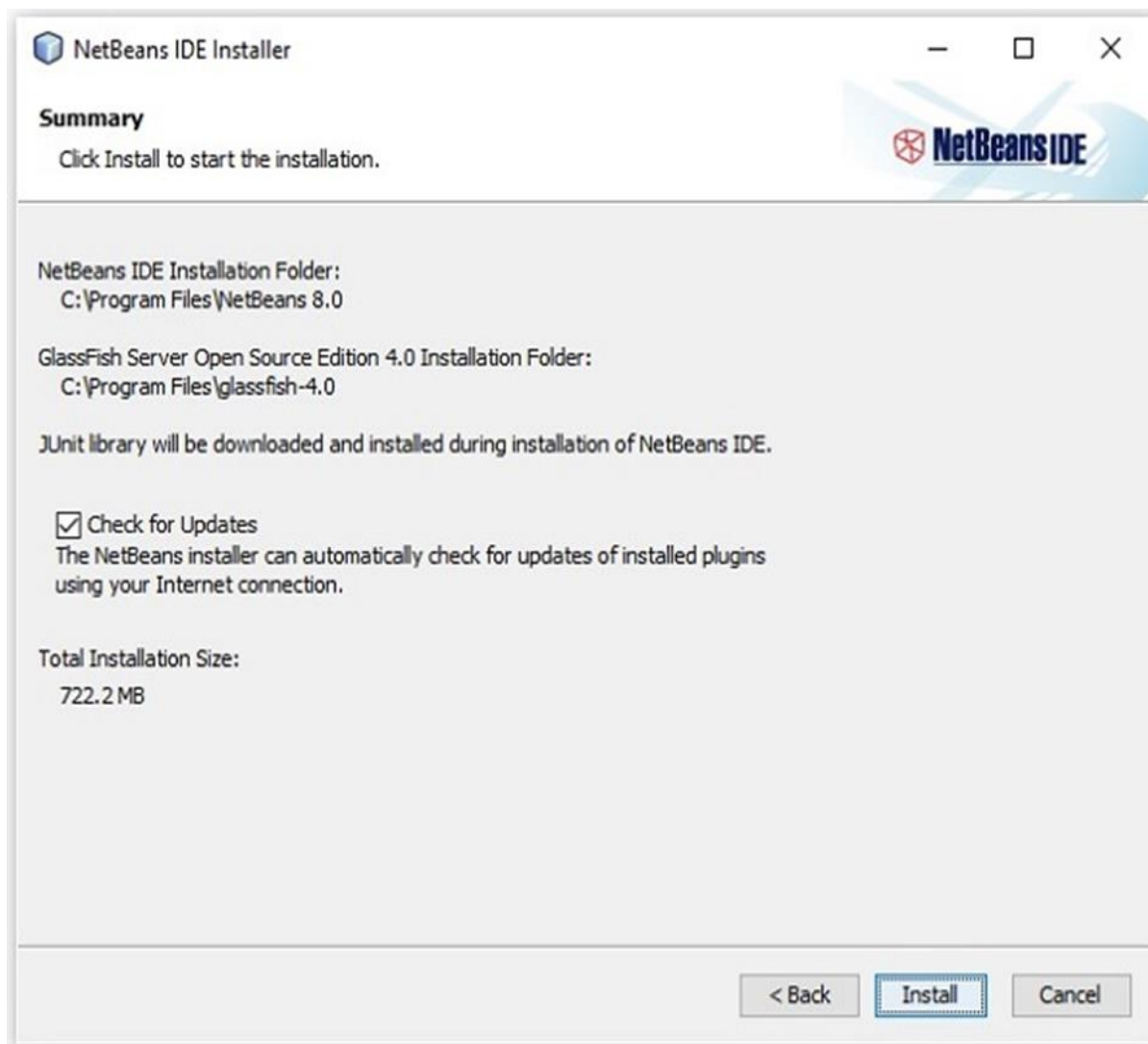
**Step 7:** Choose the destination directory where you need the Netbeans 8.0 to be installed. Furthermore, you can also browse through the directory where **Java Development Kit** is installed in your system and click on the **Next** button.



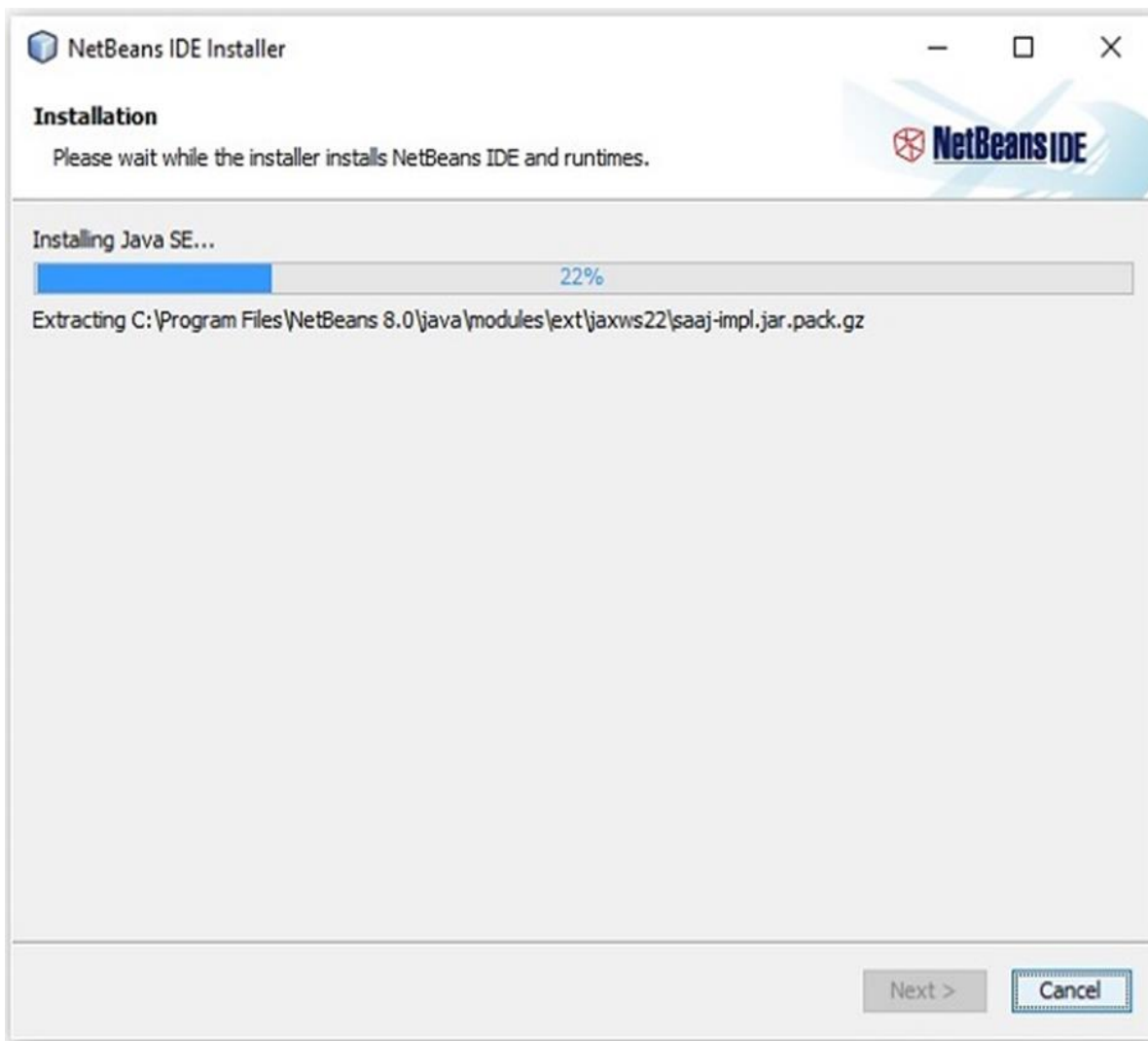
**Step 8:** Similarly, choose the destination directory for **Glassfish Server** installation. Browse through the Java Development Kit directory (now for Glassfish Reference) and then click **Next**.



**Step 9:** Check the **Check for Updates** box for automatic updates and click the **Install** button to start the installation.

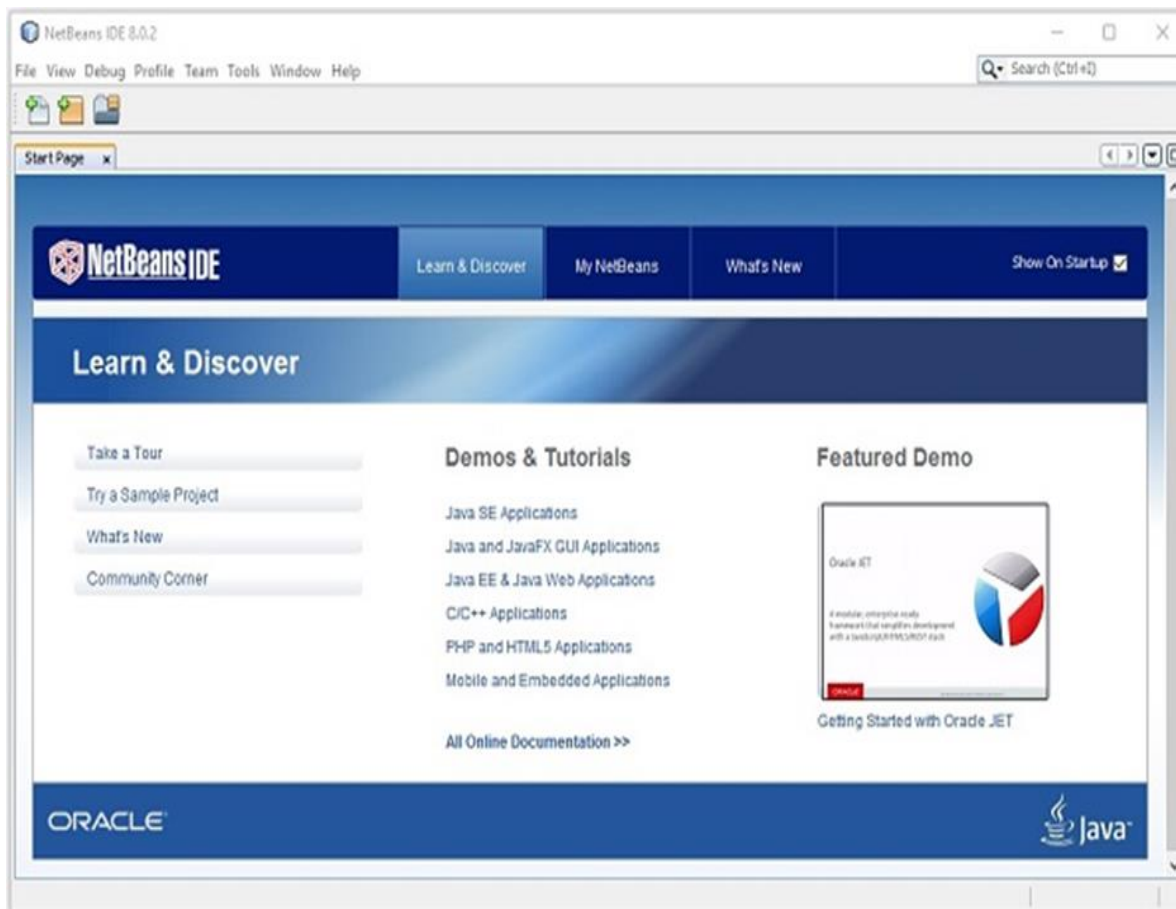


**Step 10:** This step starts the installation of NetBeans IDE 8.0 and it may take a while.



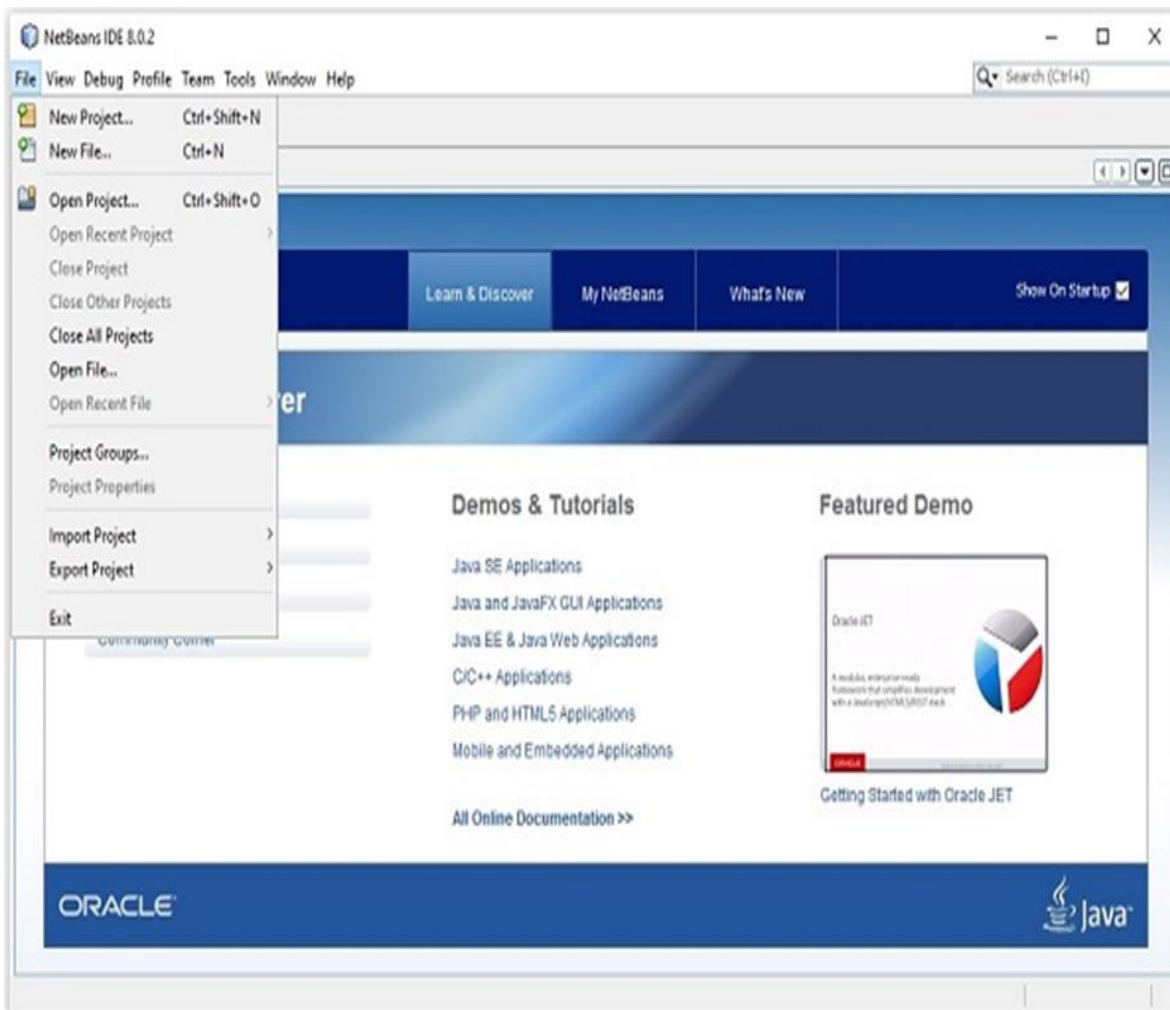
**Step 11:** Once the process is complete, click the **Finish** button to finish the installation.

**Step 12:** Once you launch the NetBeans IDE, you will see the start page as shown in the following screenshot.

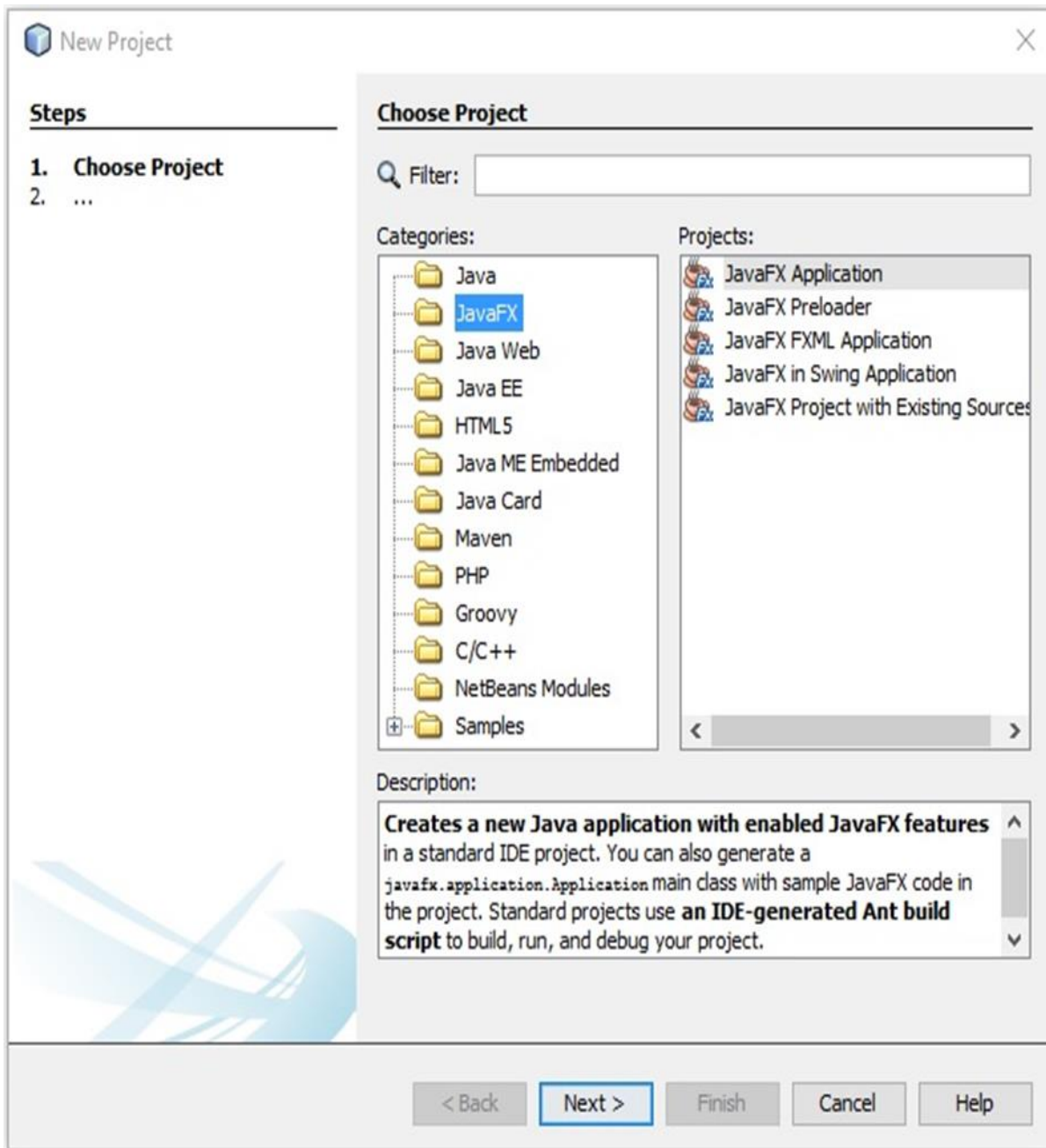




**Step 13:** In the file menu, select **New Project...** to open the New project wizard as shown in the following screenshot.



**Step 14:** In the **New Project** wizard, select **JavaFX** and click on **Next**. It starts creating a new JavaFX Application for you.



**Step 15:** Select the name of the project and location of the project in the **NewJavaFX Application** window and then click **Finish**. It creates a sample application with the given name.

**New JavaFX Application**

**Steps**

1. Choose Project
2. **Name and Location**

**Name and Location**

Project Name:

Project Location:

Project Folder:

JavaFX Platform:

Create Custom Preloader

Project Name:

Use Dedicated Folder for Storing Libraries

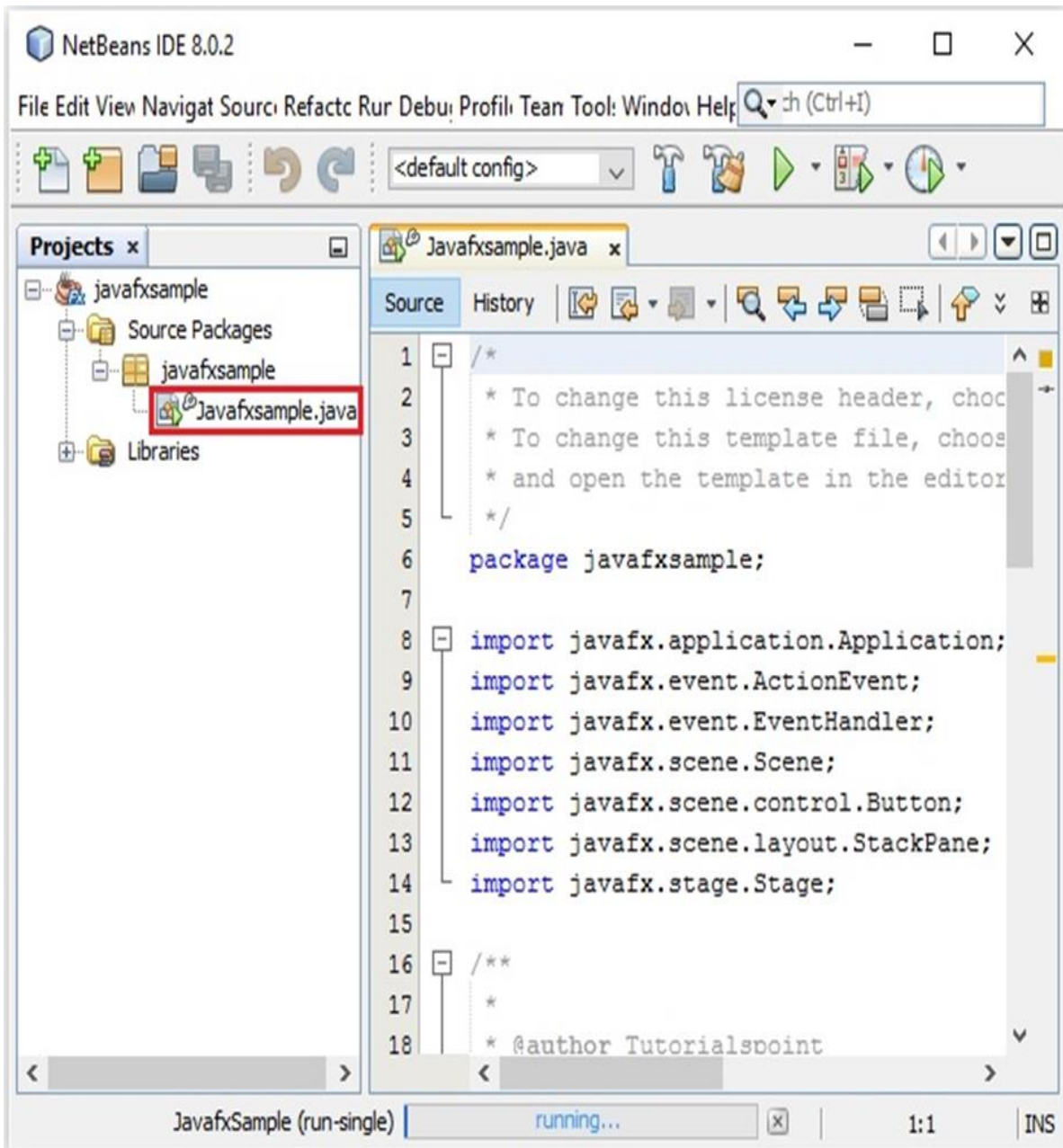
Libraries Folder:

Different users and projects can share the same compilation libraries (see Help for details).

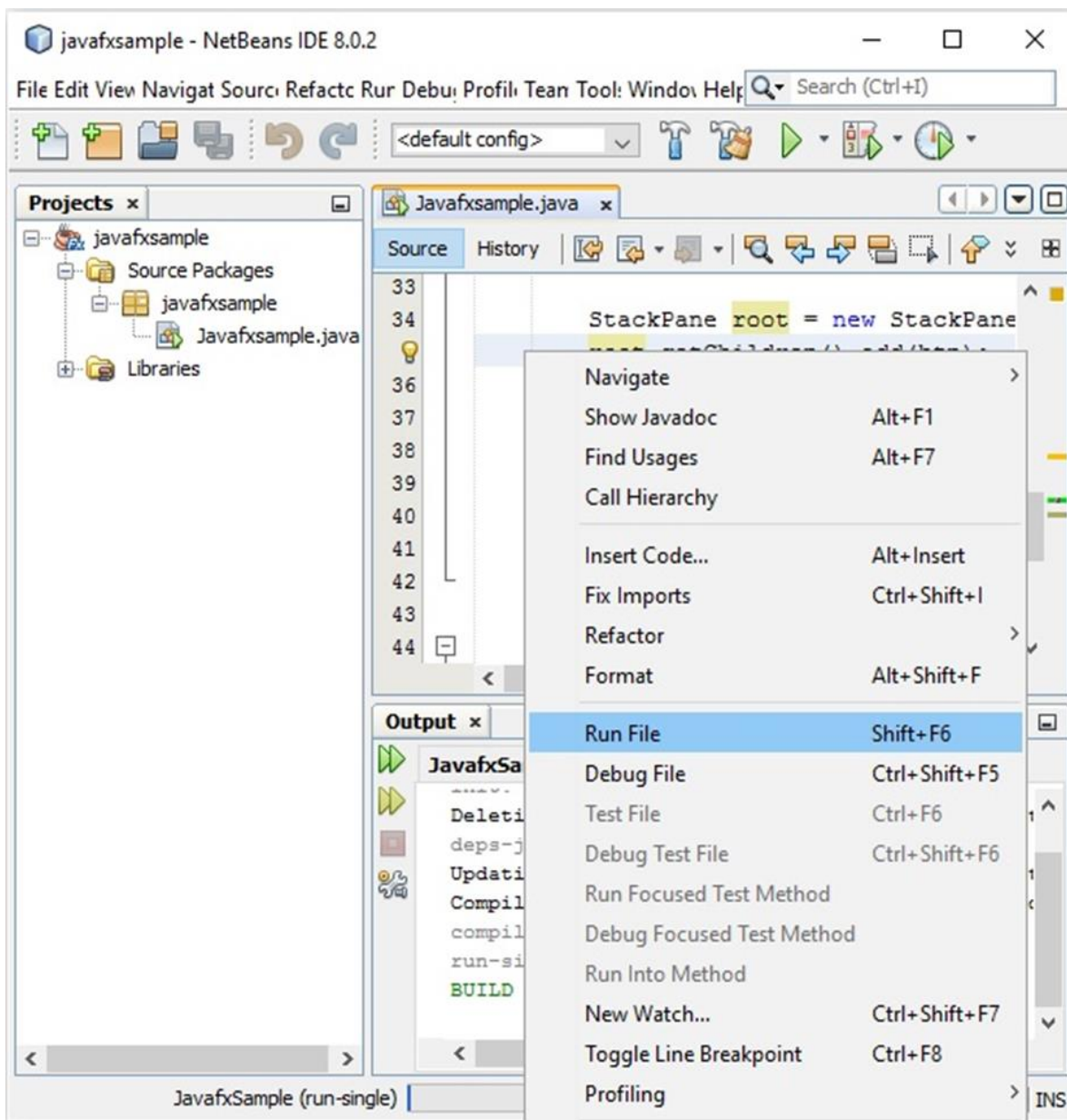
Create Application Class

< Back   Next >   **Finish**   Cancel   Help

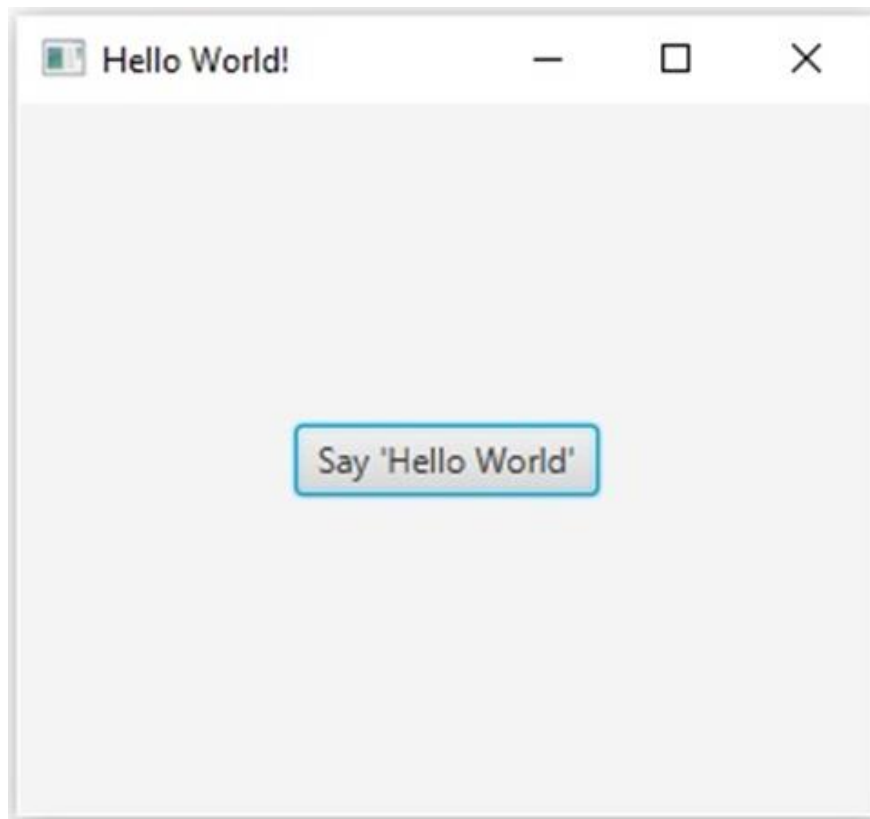
In this instance, an application with a name **javafxsample** is created. Within this application, the NetBeans IDE will generate a Java program with the name **Javafxsample.java**. As shown in the following screenshot, this program will be created inside NetBeans Source Packages → **javafxsample**.



**Step 16:** Right-click on the file and select **Run File** to run this code as shown in the following screenshot.



This automatically created program contains the code which generates a simple JavaFX window having a button with the label **Say 'Hello World'** in it. Every time you click on this button, the string **Hello World** will be displayed on the console as shown below.



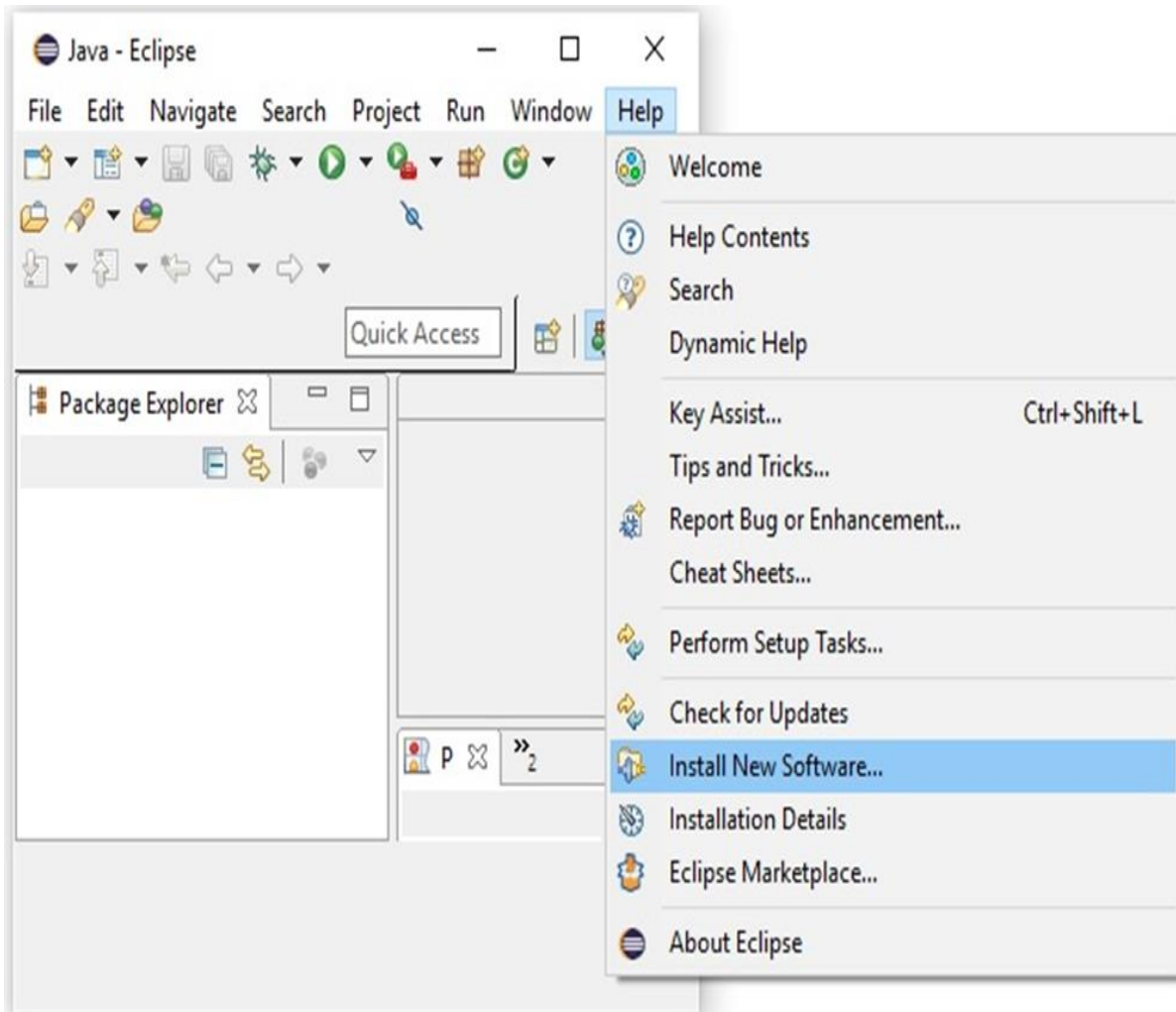
## Installing JavaFX in Eclipse

---

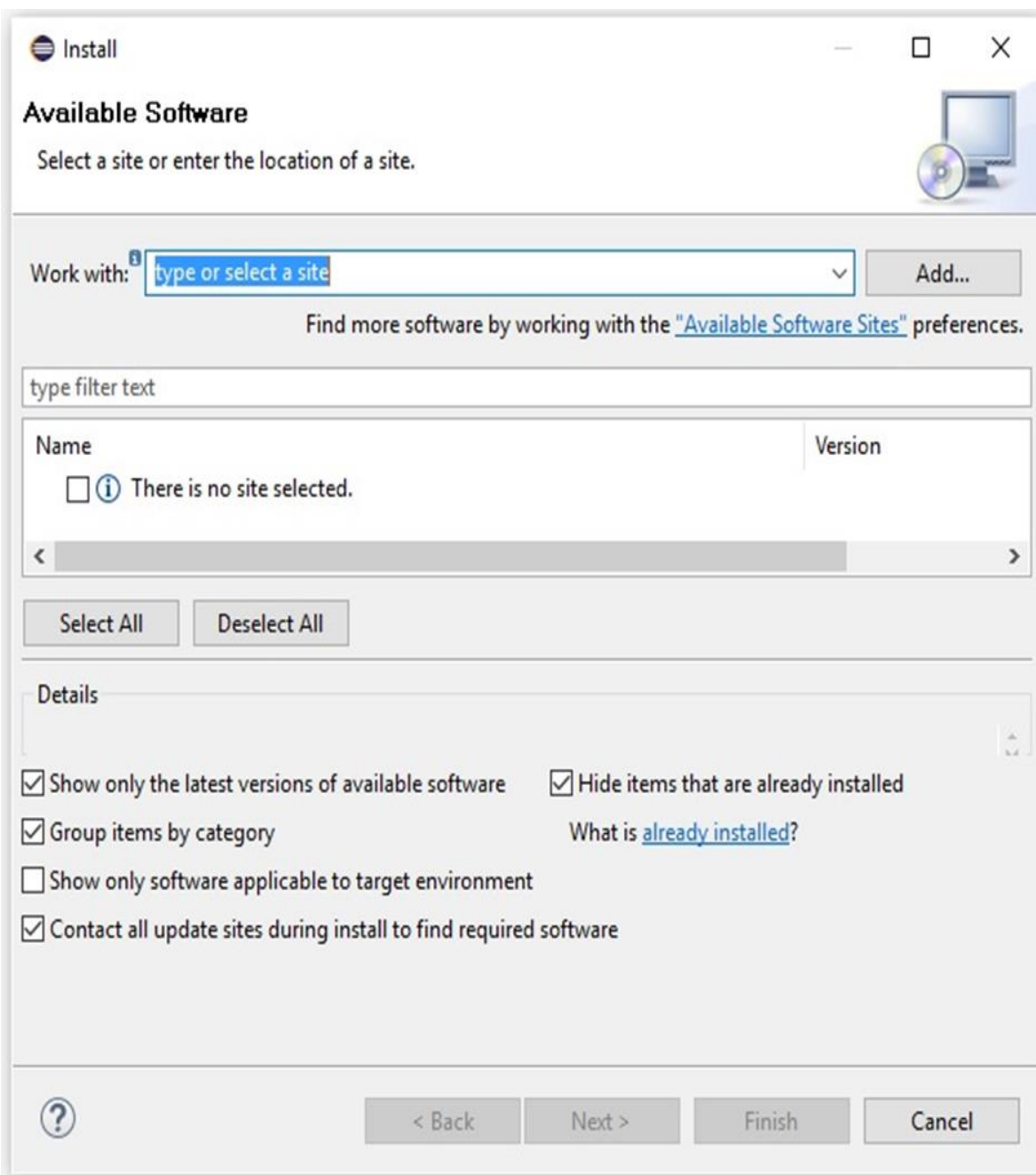
A plugin named **e(fx)clipse** is also available in JavaFX. You can use the following steps to set JavaFX in Eclipse. First of all, make sure that you have Eclipse in your system. If not, download and install Eclipse in your system.

Once Eclipse is installed, follow the steps given below to install **e(fx)clipse** in your system.

**Step 1:** Open Eclipse in the **Help** menu and select **Install New Software...** option as shown below.



Upon clicking, it will display the **Available Software** window, as shown in the following screenshot.



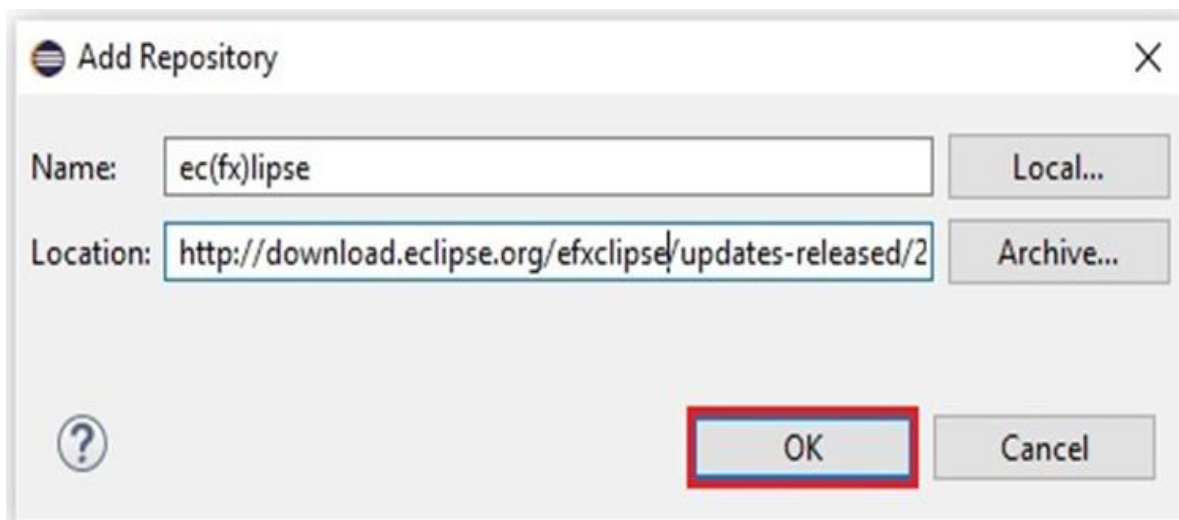
In the text box **Work with** of this window, you need to provide the link of the plugin for the required software.

**Step 2:** Click the **Add...** button. Provide the name of the plugin as **e(fx)clipse**. Next, provide the following link as the location.

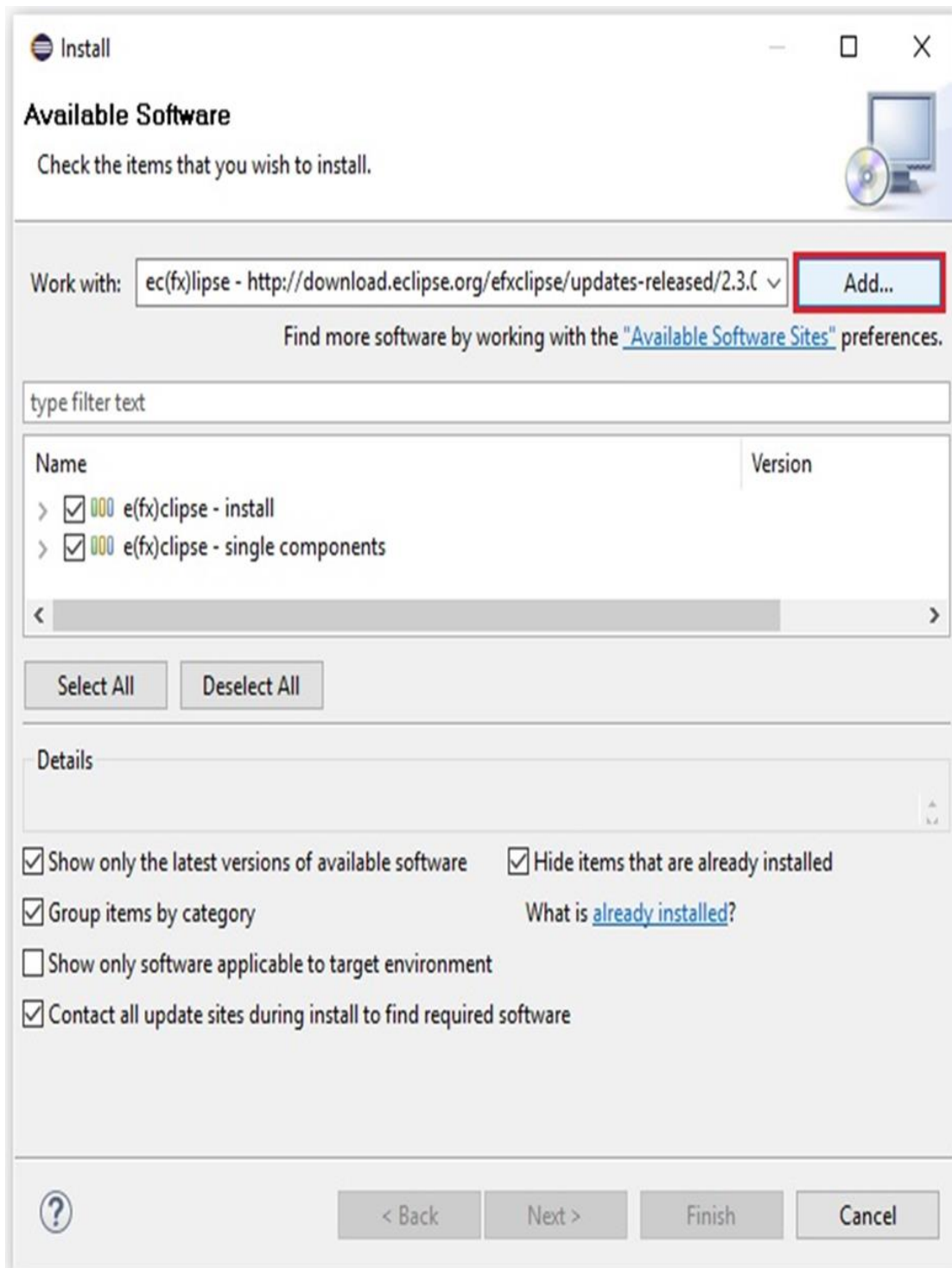
<http://download.eclipse.org/efxclipse/updates-released/2.3.0/site>



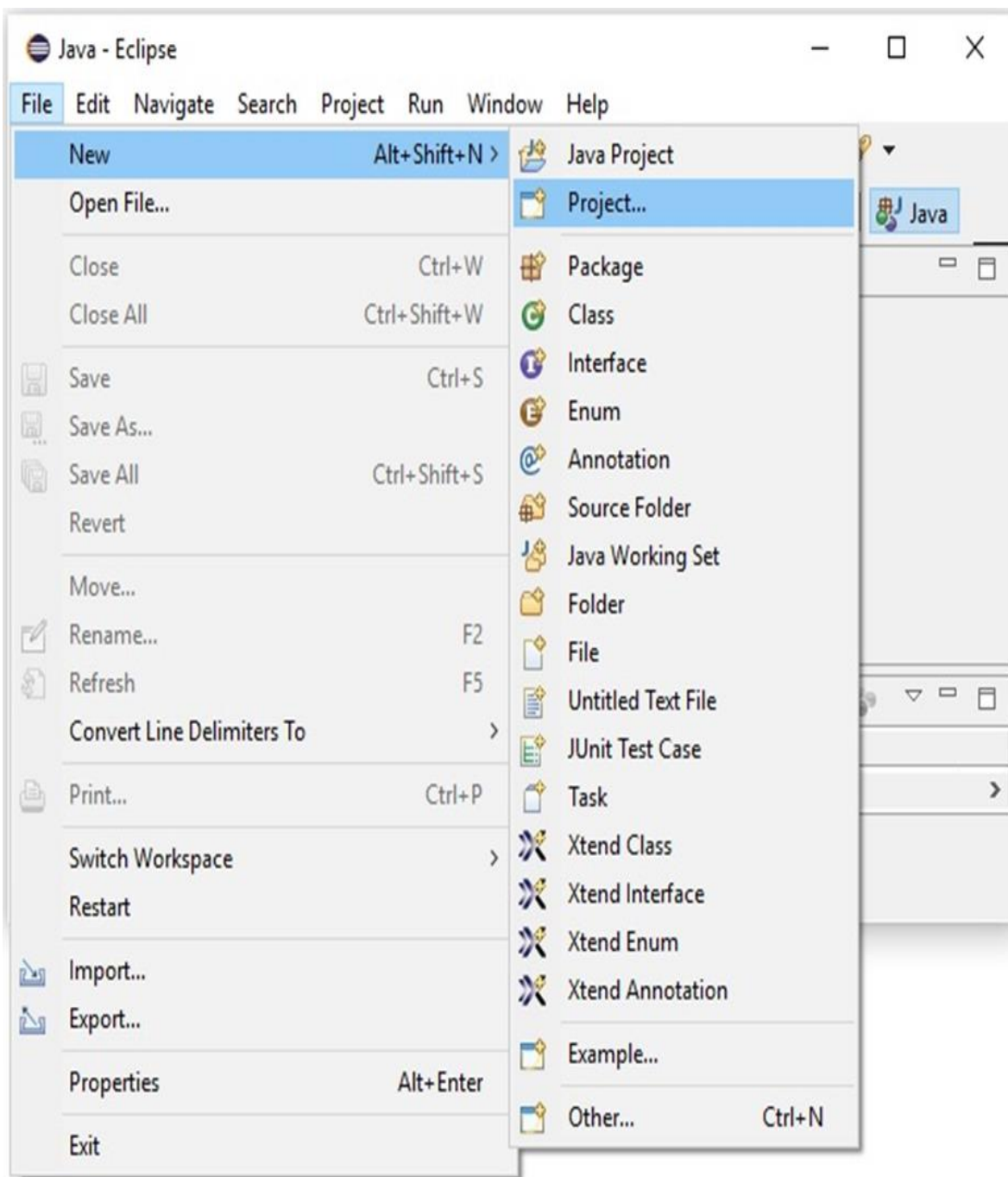
**Step 3:** After specifying the name and location of the plugin, click the **OK** button, as highlighted in the following screenshot.



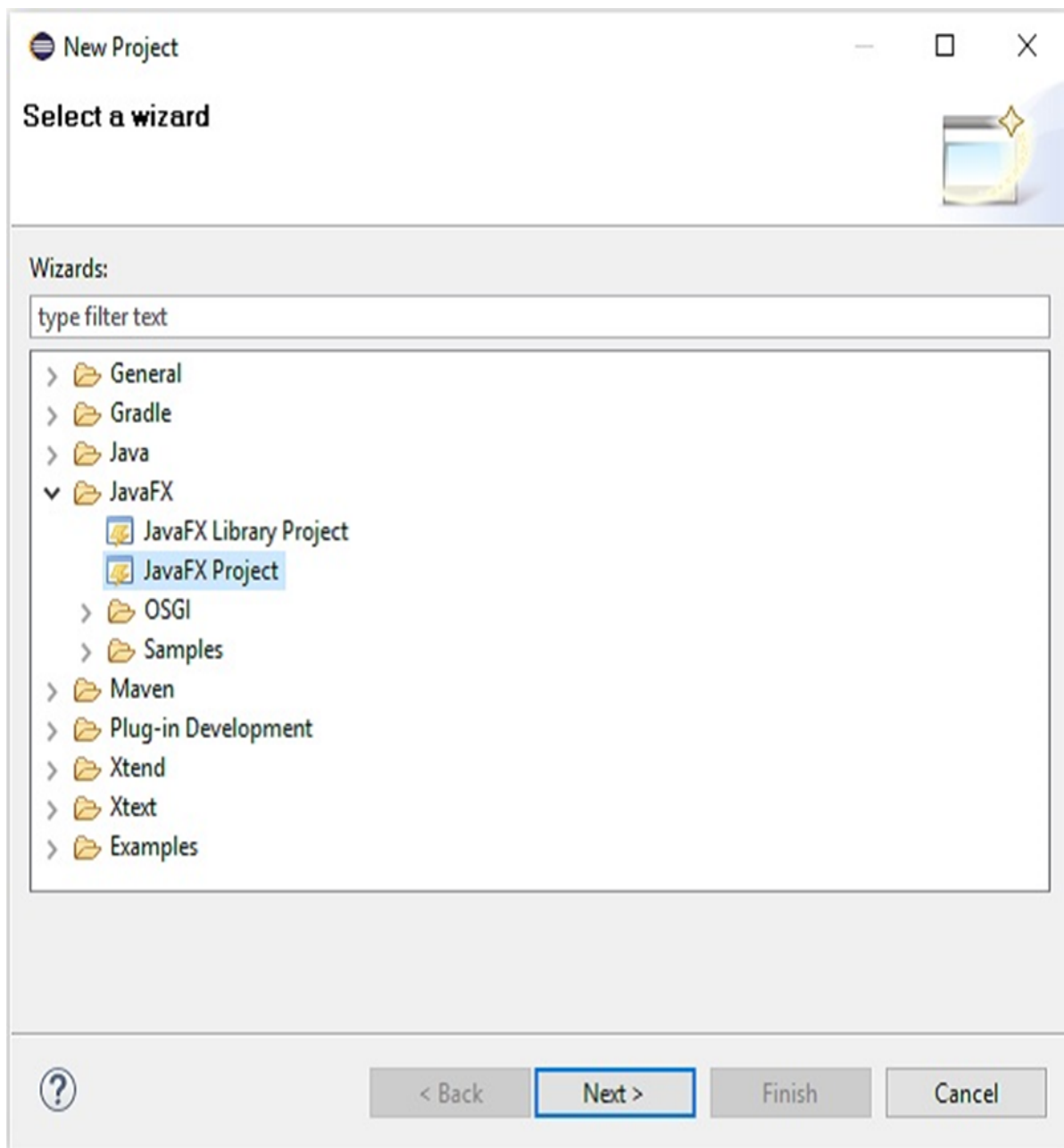
**Step 4:** Soon after you add the plugin, you will find two checkboxes for **e(fx)clipse – install** and **e(fx)clipse – single components**. Check both these checkboxes and click the **Add...** button as shown in the following screenshot.



**Step 5:** Next, open your Eclipse IDE. Click the File menu and select Project as shown in the following screenshot.



**Step 6:** Then, you will get a window where you can see a list of wizards provided by Eclipse to create a project. Expand the **JavaFX** wizard, select **JavaFX Project** and click the **Next** button as shown in the following screenshot.



**Step 7:** On clicking **Next**, a New Project Wizard opens. Here, you can type the required project name and click **Finish**.

**New Java Project**

**Create a Java Project**  
Create a Java project in the workspace or in an external location.

Project name:

Use default location  
Location:  [Browse...](#)

JRE

Use an execution environment JRE:  [Configure JREs...](#)

Use a project specific JRE:

Use default JRE (currently 'jre1.8.0\_72')

Project layout

Use project folder as root for sources and class files

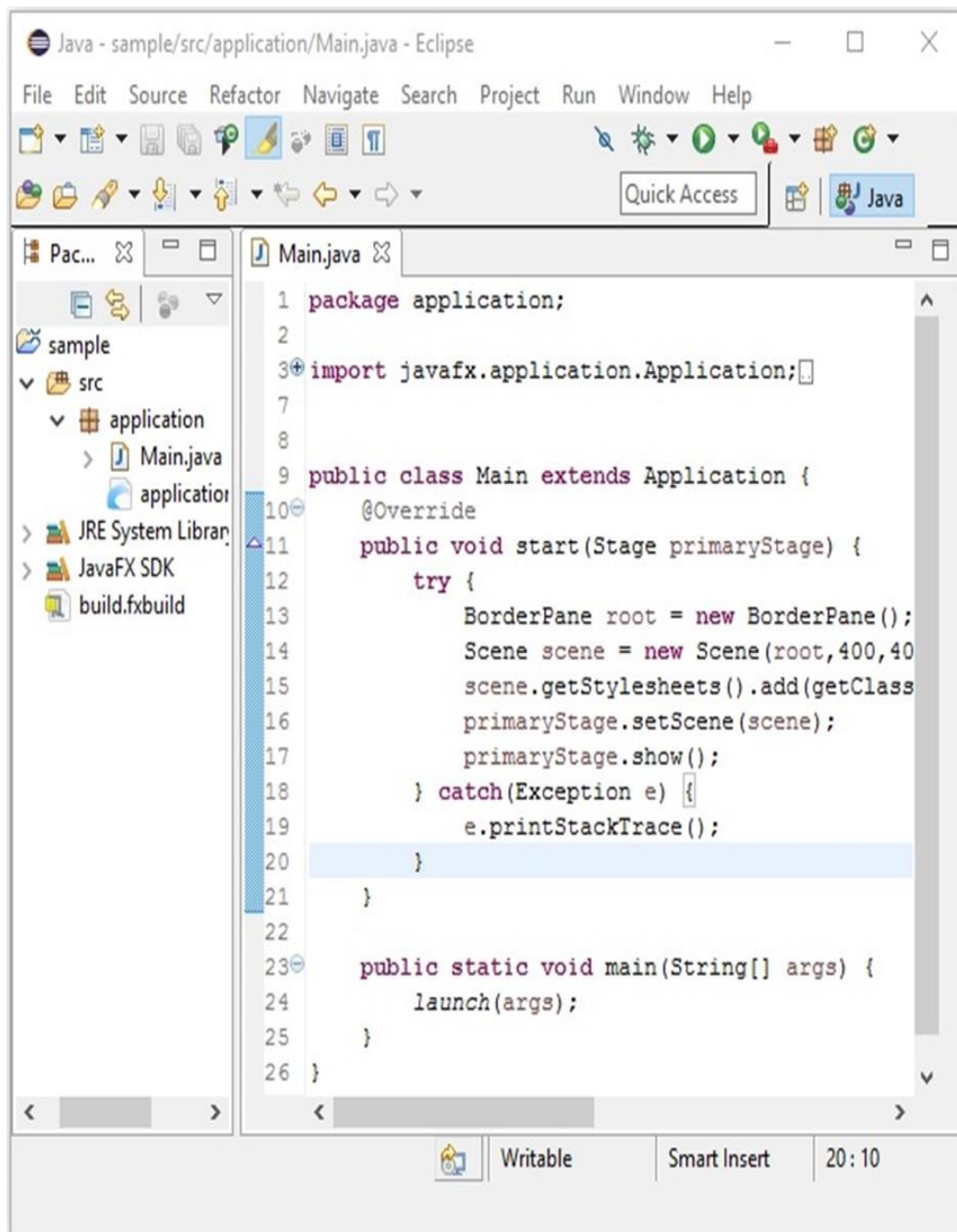
Create separate folders for sources and class files [Configure default...](#)

Working sets

Add project to working sets  
Working sets:  [Select...](#)

[?](#)

**Step 8:** On clicking Finish, an application is created with the given name (sample). In the sub-package named **application**, a program with the name **Main.java** is generated as shown below.

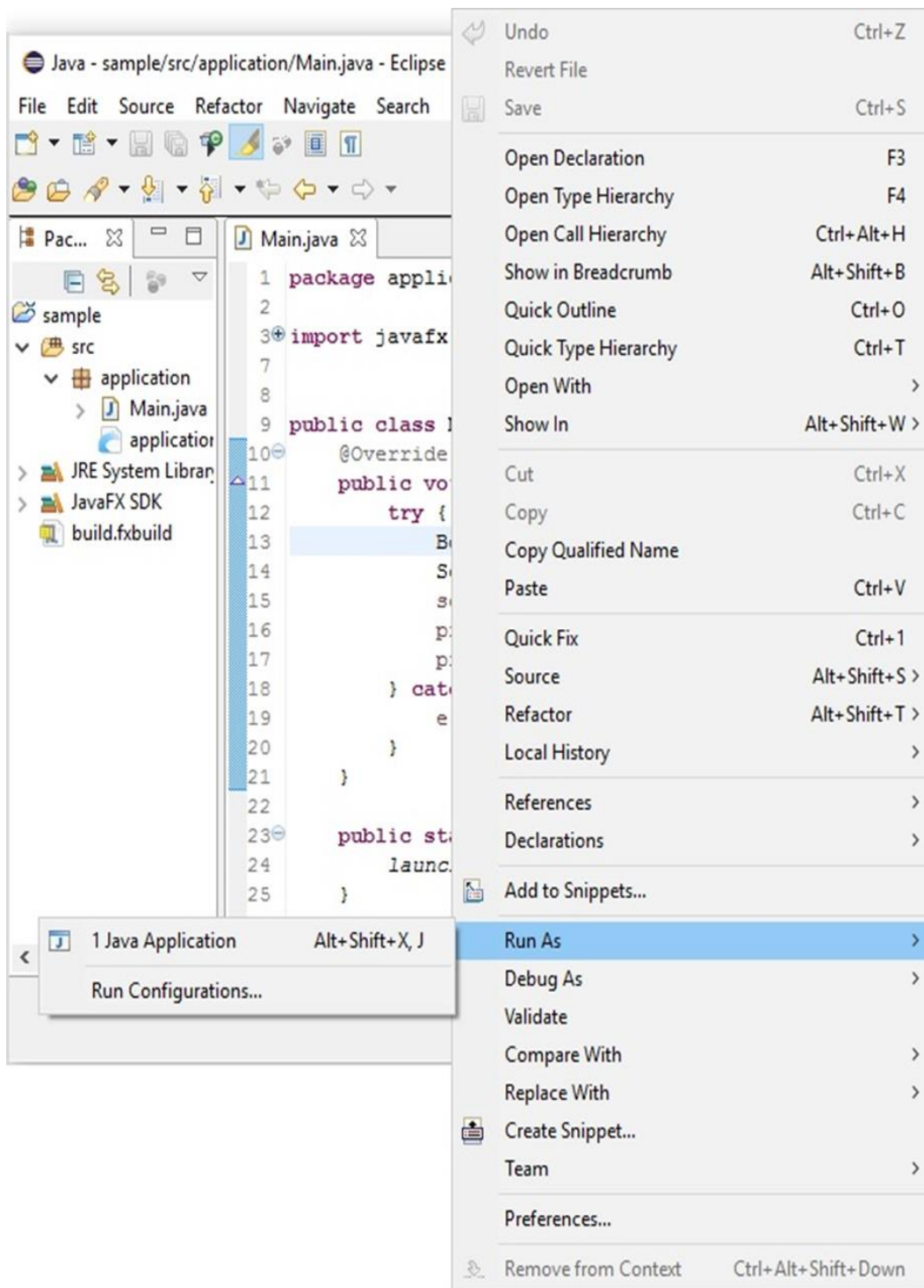


The screenshot shows the Eclipse IDE interface. The title bar reads "Java - sample/src/application/Main.java - Eclipse". The menu bar includes File, Edit, Source, Refactor, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations and development tools. The left sidebar shows the project structure: sample > src > application > Main.java. The main editor window displays the following Java code:

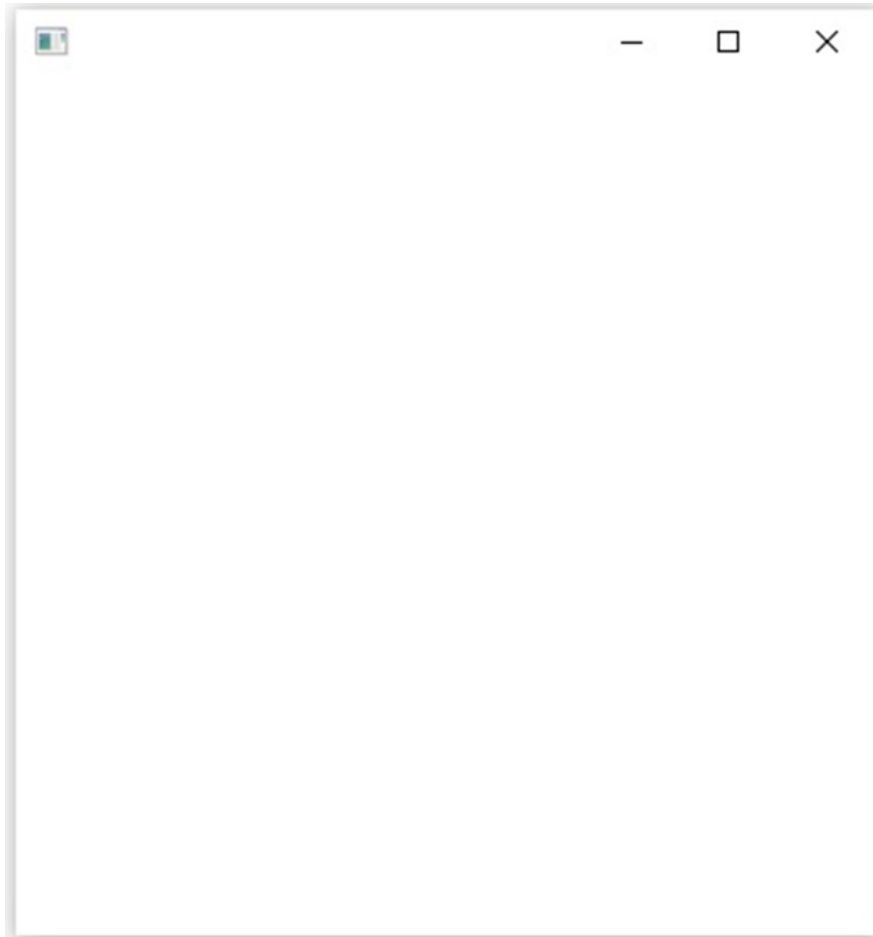
```
1 package application;
2
3 import javafx.application.Application;
4
5
6
7
8
9 public class Main extends Application {
10     @Override
11     public void start(Stage primaryStage) {
12         try {
13             BorderPane root = new BorderPane();
14             Scene scene = new Scene(root, 400, 400);
15             scene.getStylesheets().add(getClass().getResource("application.css").toExternalForm());
16             primaryStage.setScene(scene);
17             primaryStage.show();
18         } catch (Exception e) {
19             e.printStackTrace();
20         }
21     }
22
23     public static void main(String[] args) {
24         launch(args);
25     }
26 }
```

The status bar at the bottom indicates "Writable", "Smart Insert", and "20:10".

**Step 9:** This automatically generated program contains the code to generate an empty JavaFX window. Right-click on this file, select **Run As** → **Java Application** as shown in the following screenshot.



On executing this application, it gives you an empty JavaFX window as shown below.



**Note:** We will discuss more about the code in the later chapters.

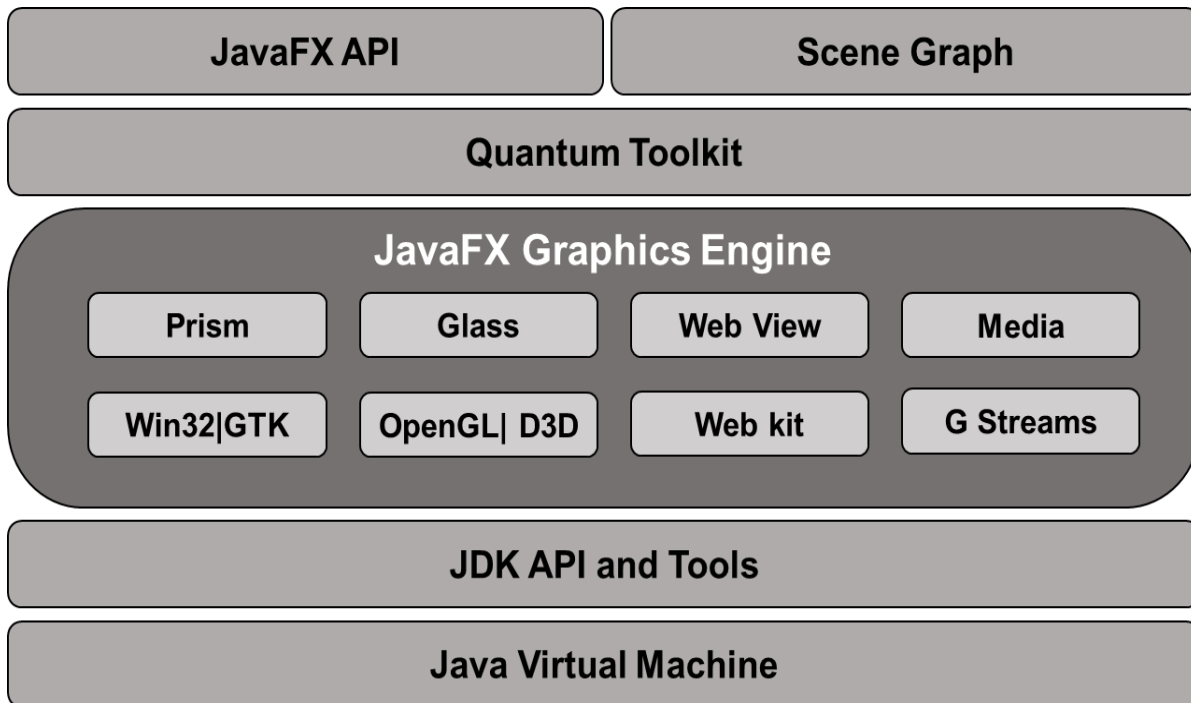


# 3. JAVAFX – ARCHITECTURE

JavaFX provides a complete API with a rich set of classes and interfaces to build GUI applications with rich graphics. The important packages of this API are –

- **javafx.animation:** Contains classes to add transition based animations such as fill, fade, rotate, scale and translation, to the JavaFX nodes.
- **javafx.application:** Contains a set of classes responsible for the JavaFX application life cycle.
- **javafx.css:** Contains classes to add CSS-like styling to JavaFX GUI applications.
- **javafx.event:** Contains classes and interfaces to deliver and handle JavaFX events.
- **javafx.geometry:** Contains classes to define 2D objects and perform operations on them.
- **javafx.stage:** This package holds the top level container classes for JavaFX application.
- **javafx.scene:** This package provides classes and interfaces to support the scene graph. In addition, it also provides sub-packages such as canvas, chart, control, effect, image, input, layout, media, paint, shape, text, transform, web, etc. There are several components that support this rich API of JavaFX.

The following illustration shows the architecture of JavaFX API. Here you can see the components that support JavaFX API.



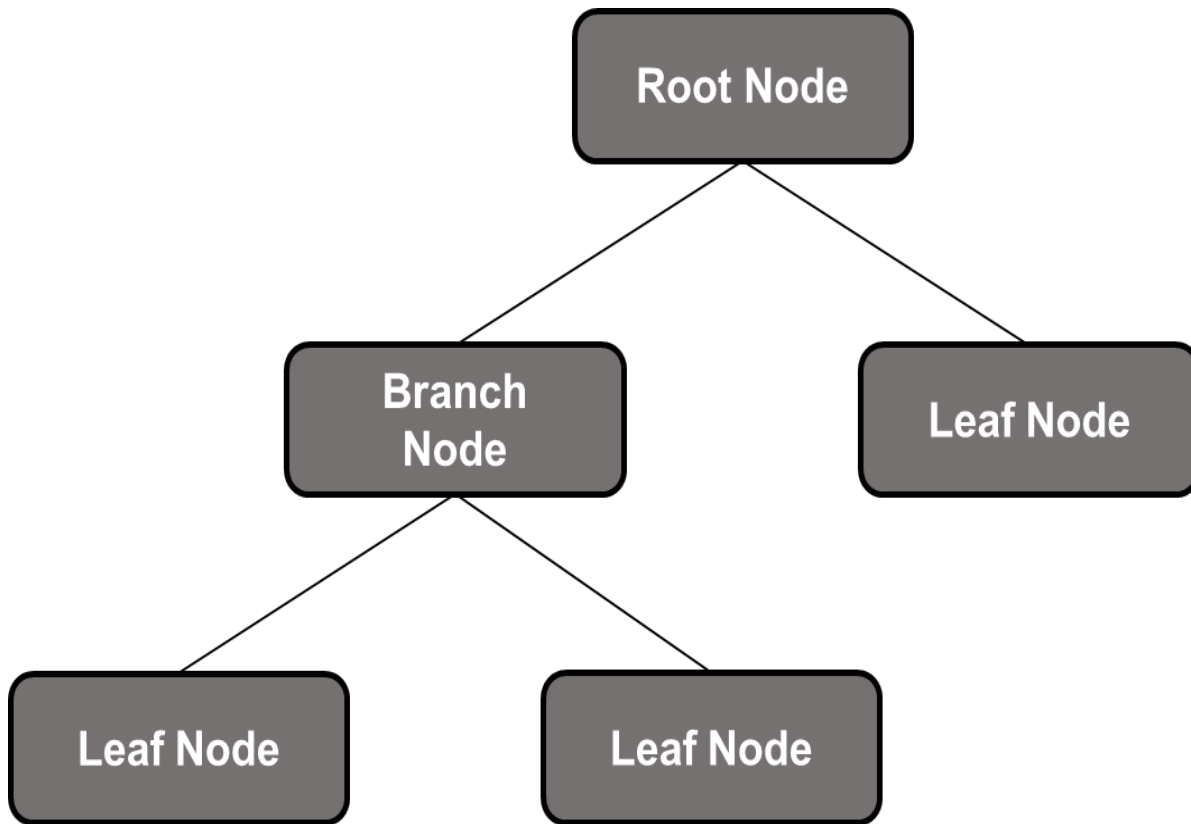
## Scene Graph

In JavaFX, the GUI Applications were coded using a Scene Graph. A Scene Graph is the starting point of the construction of the GUI Application. It holds the (GUI) application primitives that are termed as nodes.

A node is a visual/graphical object and it may include —

- **Geometrical (Graphical) objects** – (2D and 3D) such as circle, rectangle, polygon, etc.
- **UI controls** – such as Button, Checkbox, Choice box, Text Area, etc.
- **Containers** – (layout panes) such as Border Pane, Grid Pane, Flow Pane, etc.
- **Media elements** – such as audio, video and image objects.

In general, a collection of nodes makes a scene graph. All these nodes are arranged in a hierarchical order as shown below.



Each node in the scene graph has a single parent, and the node which does not contain any parents is known as the **root node**.

In the same way, every node has one or more children, and the node without children is termed as **leaf node**; a node with children is termed as a **branch node**.

A node instance can be added to a scene graph only once. The nodes of a scene graph can have Effects, Opacity, Transforms, Event Handlers, Event Handlers, Application Specific States.

## Prism

---

Prism is a **high performance hardware-accelerated graphical pipeline** that is used to render the graphics in JavaFX. It can render both 2-D and 3-D graphics.

To render graphics, a Prism uses –

- DirectX 9 on Windows XP and Vista.
- DirectX 11 on Windows 7.
- OpenGL on Mac and Linux, Embedded Systems.

In case the hardware support for graphics on the system is not sufficient, then Prism uses the software render path to process the graphics.

When used with a supported Graphic Card or GPU, it offers smoother graphics. Just in case the system does not support a graphic card, then Prism defaults to the software rendering stack (either of the above two).

## GWT (Glass Windowing Toolkit)

---

As the name suggests, GWT provides services to manage Windows, Timers, Surfaces and Event Queues. GWT connects the JavaFX Platform to the Native Operating System.

## Quantum Toolkit

---

It is an abstraction over the low-level components of Prism, Glass, Media Engine, and Web Engine. It ties Prism and GWT together and makes them available to JavaFX.

## WebView

---

Using JavaFX, you can also embed HTML content in to a scene graph. **WebView** is the component of JavaFX which is used to process this content. It uses a technology called **Web Kit**, which is an internal open-source web browser engine. This component supports different web technologies like HTML5, CSS, JavaScript, DOM and SVG.

Using WebView, you can –

- Render HTML content from local or remote URL.
- Support history and provide Back and Forward navigation.
- Reload the content.
- Apply effects to the web component.
- Edit the HTML content.
- Execute JavaScript commands.
- Handle events.

In general, using WebView, you can control web content from Java.

## Media Engine

---

The **JavaFX media engine** is based on an open-source engine known as a **Streamer**. This media engine supports the playback of video and audio content.

The JavaFX media engine provides support for audio for the following file formats:

<b>Audio</b>	<ul style="list-style-type: none"><li>• MP3</li><li>• AIFF</li><li>• WAV</li></ul>
<b>Video</b>	<ul style="list-style-type: none"><li>• FLV</li></ul>

The package **javafx.scene.media** contains the classes and interfaces to provide media functionality in JavaFX. It is provided in the form of three components, which are –

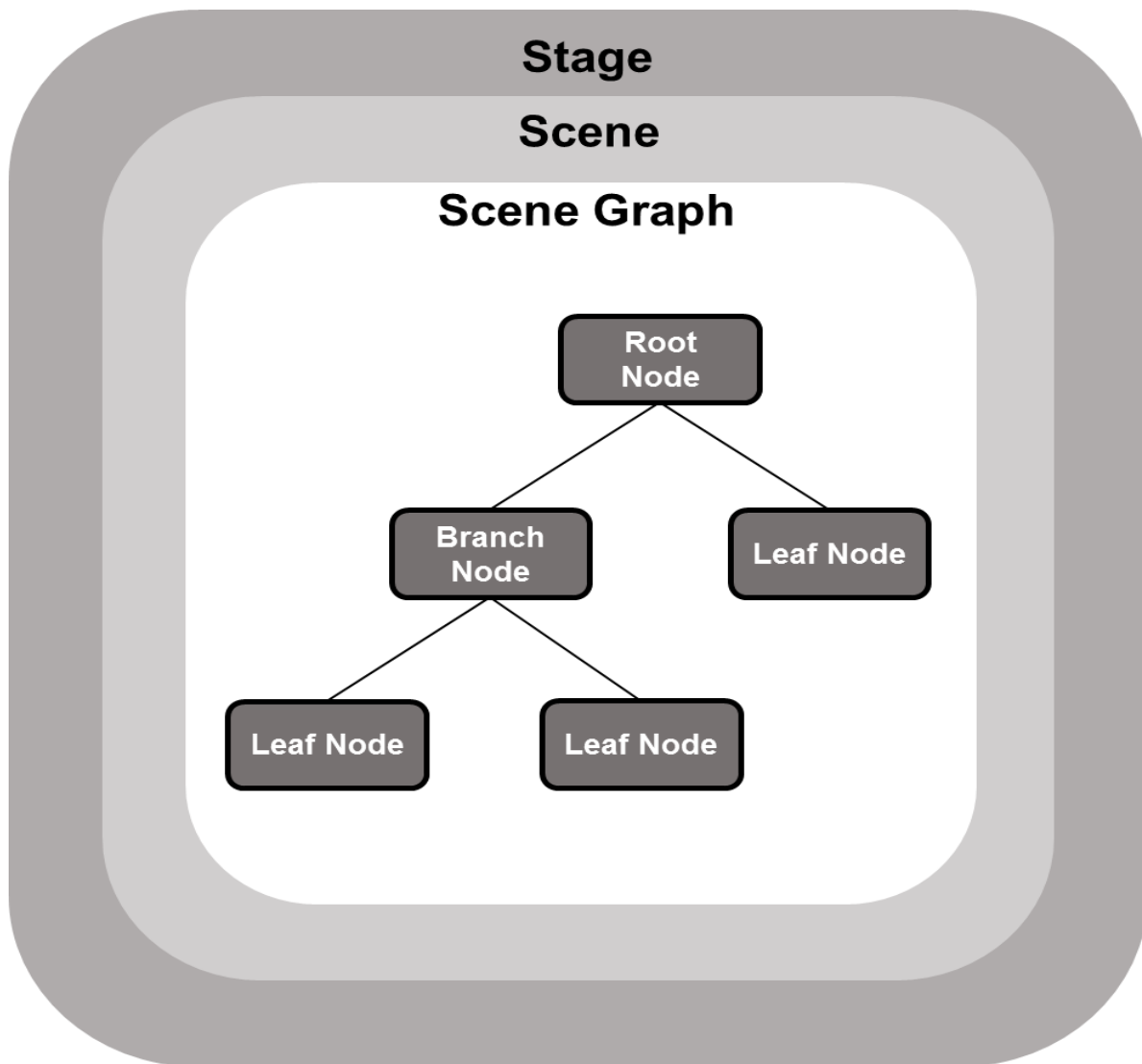
- Media Object: This represents a media file.
- Media Player: To play media content.
- Media View: To display media.

# 4. JAVAFX – APPLICATION

In this chapter, we will discuss the structure of a JavaFX application in detail and also learn to create a JavaFX application with an example.

## JavaFX Application Structure

In general, a JavaFX application will have three major components namely **Stage**, **Scene** and **Nodes** as shown in the following diagram.



## Stage

A stage (a window) contains all the objects of a JavaFX application. It is represented by **Stage** class of the package **javafx.stage**. The primary stage is created by the platform itself. The created stage object is passed as an argument to the **start()** method of the **Application** class (explained in the next section).

A stage has two parameters determining its position namely **Width** and **Height**. It is divided as Content Area and Decorations (Title Bar and Borders).

There are five types of stages available –

- Decorated
- Undecorated
- Transparent
- Unified
- Utility

You have to call the **show()** method to display the contents of a stage.

## Scene

A scene represents the physical contents of a JavaFX application. It contains all the contents of a scene graph. The class **Scene** of the package **javafx.scene** represents the scene object. At an instance, the scene object is added to only one stage.

You can create a scene by instantiating the Scene Class. You can opt for the size of the scene by passing its dimensions (height and width) along with the **root node** to its constructor.

## Scene Graph and Nodes

A **scene graph** is a tree-like data structure (hierarchical) representing the contents of a scene. In contrast, a **node** is a visual/graphical object of a scene graph.

A node may include –

- Geometrical (Graphical) objects (2D and 3D) such as – Circle, Rectangle, Polygon, etc.
- UI Controls such as – Button, Checkbox, Choice Box, Text Area, etc.
- Containers (Layout Panes) such as Border Pane, Grid Pane, Flow Pane, etc.
- Media elements such as Audio, Video and Image Objects.

The **Node** Class of the package **javafx.scene** represents a node in JavaFX, this class is the super class of all the nodes.

As discussed earlier a node is of three types –

- **Root Node:** The first Scene Graph is known as the Root node.
- **Branch Node/Parent Node:** The node with child nodes are known as branch/parent nodes. The abstract class named **Parent** of the package **javafx.scene** is the base class of all the parent nodes, and those parent nodes will be of the following types –
  - **Group:** A group node is a collective node that contains a list of children nodes. Whenever the group node is rendered, all its child nodes are rendered in order. Any transformation, effect state applied on the group will be applied to all the child nodes.
  - **Region:** It is the base class of all the JavaFX Node based UI Controls, such as Chart, Pane and Control.
  - **WebView:** This node manages the web engine and displays its contents.
- **Leaf Node:** The node without child nodes is known as the leaf node. For example, Rectangle, Ellipse, Box, ImageView, MediaView are examples of leaf nodes.

It is mandatory to pass the root node to the scene graph. If the Group is passed as root, all the nodes will be clipped to the scene and any alteration in the size of the scene will not affect the layout of the scene.

## Creating a JavaFX Application

---

To create a JavaFX application, you need to instantiate the Application class and implement its abstract method **start()**. In this method, we will write the code for the JavaFX Application.

### Application Class

The **Application** class of the package **javafx.application** is the entry point of the application in JavaFX. To create a JavaFX application, you need to inherit this class and implement its abstract method **start()**. In this method, you need to write the entire code for the JavaFX graphics.

In the **main** method, you have to launch the application using the **launch()** method. This method internally calls the **start()** method of the Application class as shown in the following program.

```
public class JavafxSample extends Application {

    @Override
    public void start(Stage primaryStage) throws Exception {
        /*
         Code for JavaFX application.
        */
    }
}
```



```
    (Stage, scene, scene graph)
    */
    }
    public static void main(String args[]){
        launch(args);
    }
}
```

End of ebook preview  
If you liked what you saw...  
Buy it from our store @ <https://store.tutorialspoint.com>