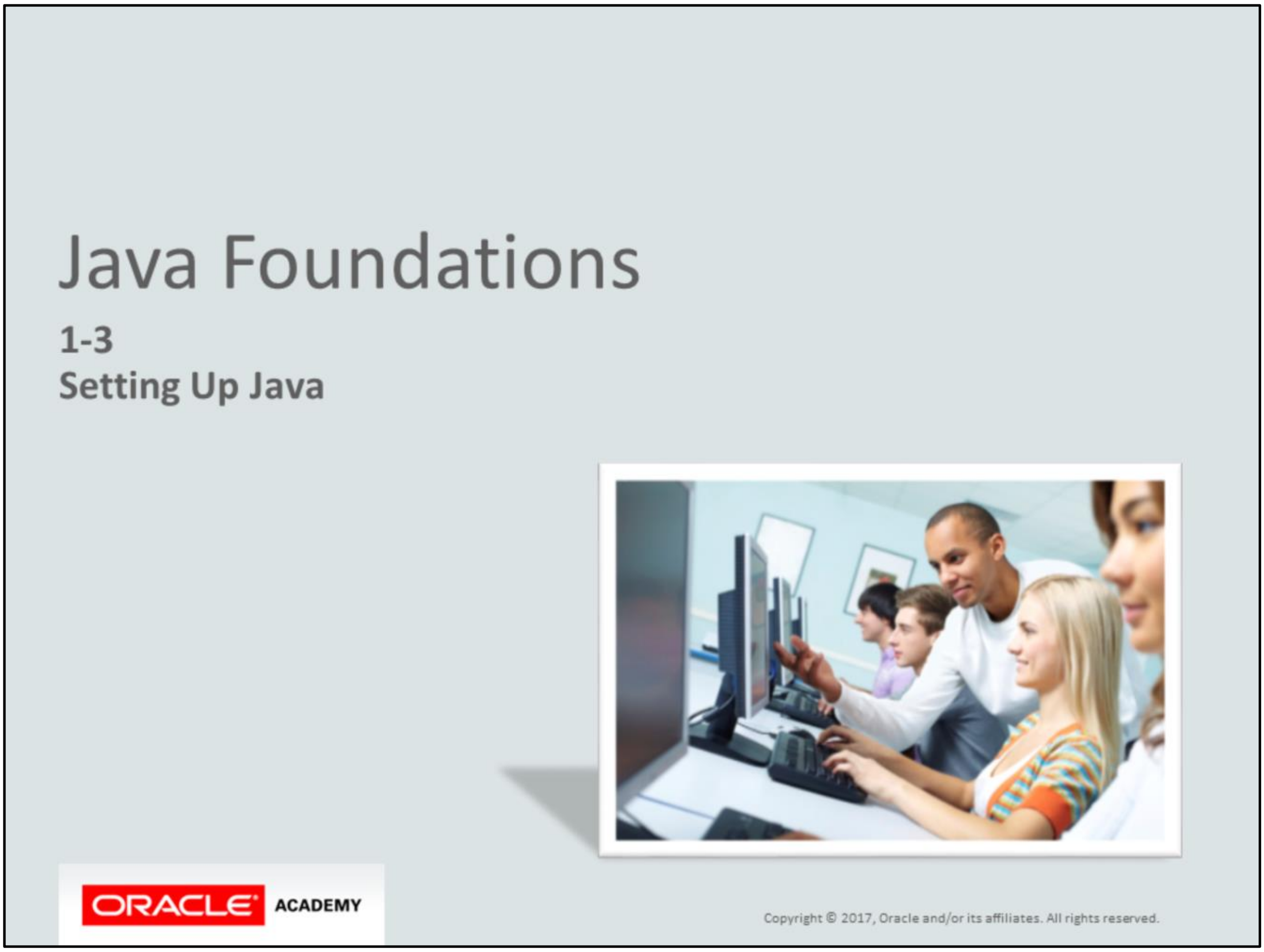
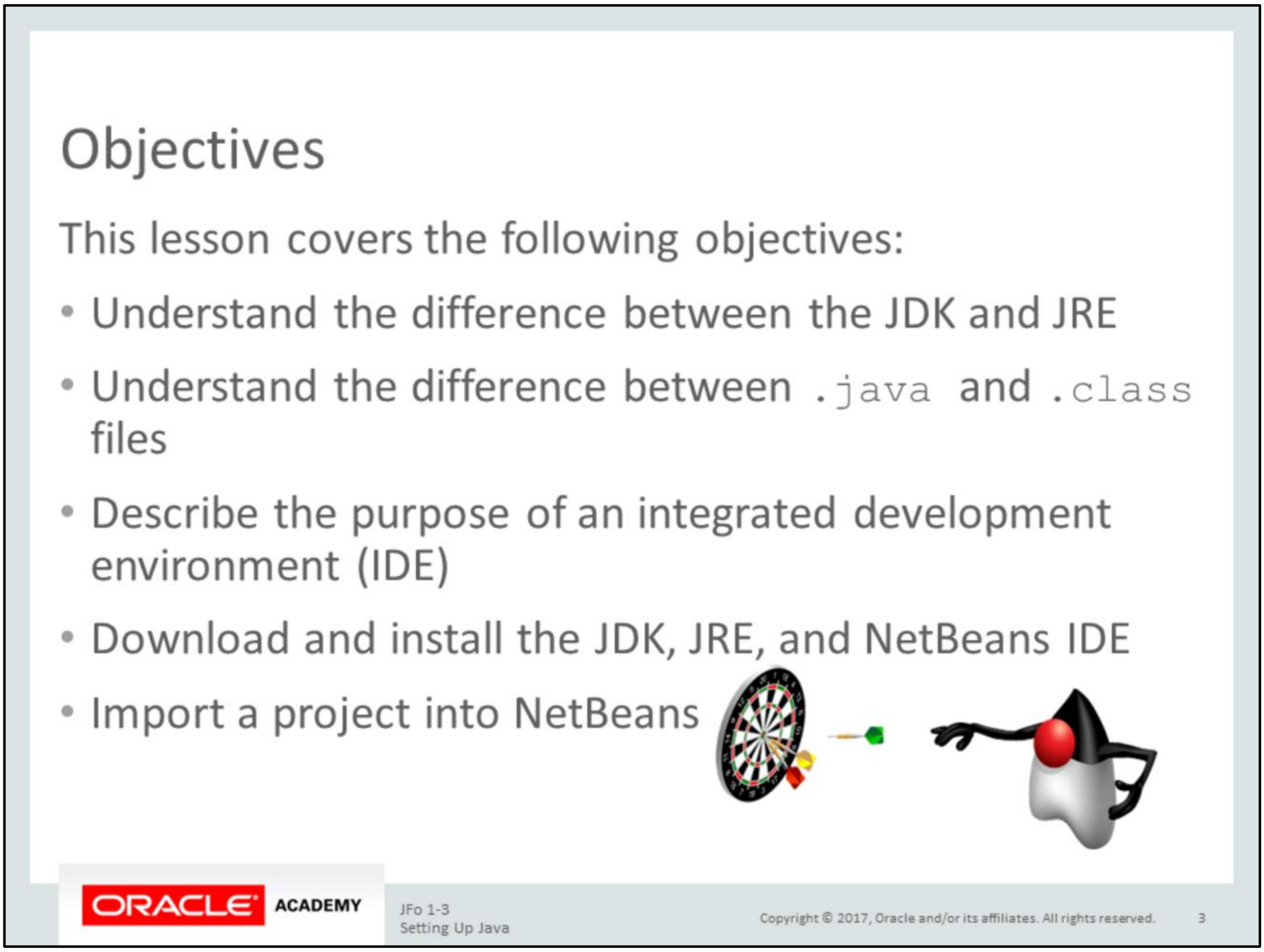
ORACLE® ACADEMY



Java Foundations

1-3
Setting Up Java

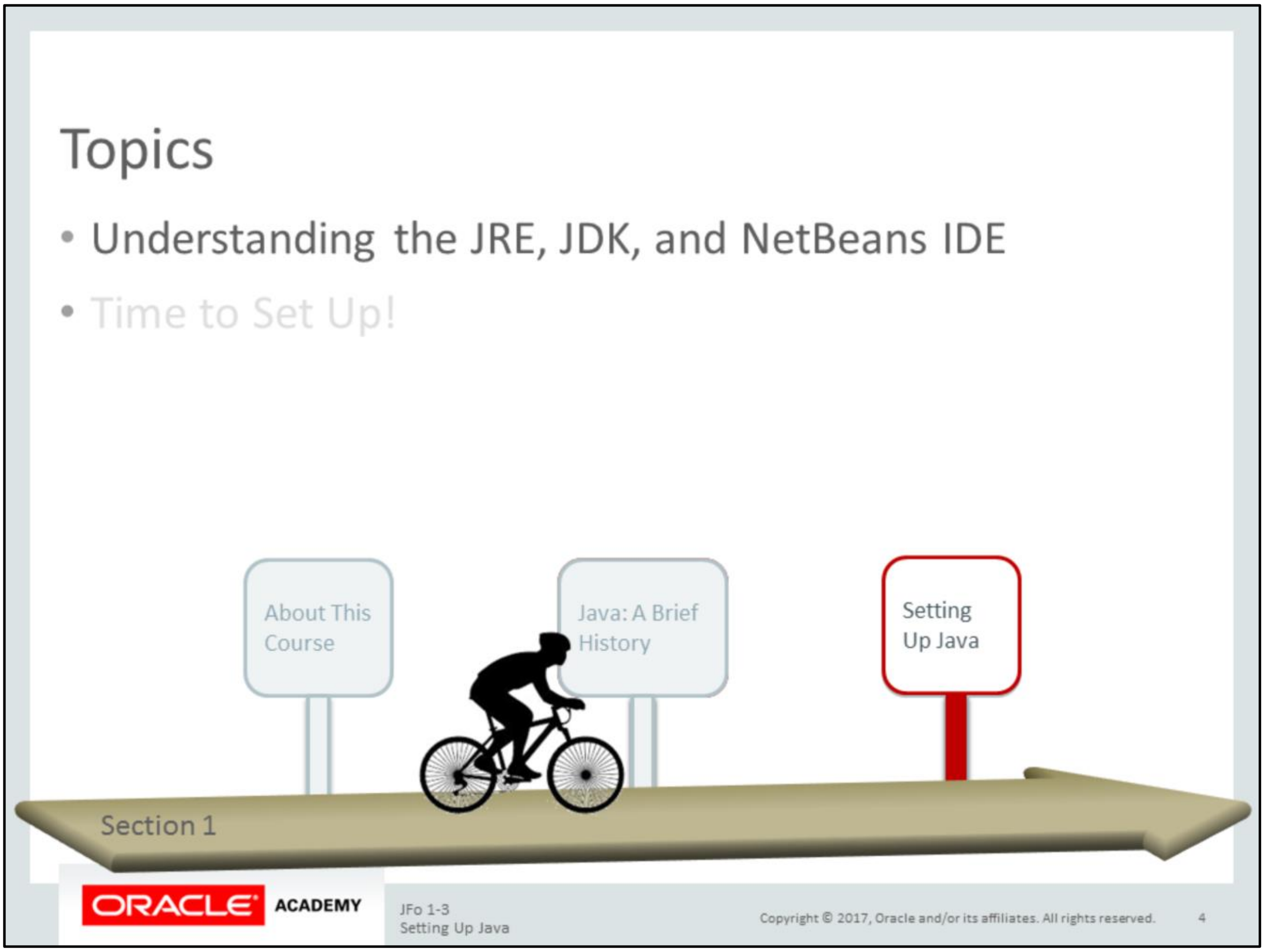
ORACLE ACADEMY



Objectives

This lesson covers the following objectives:

- Understand the difference between the JDK and JRE
- Understand the difference between `.java` and `.class` files
- Describe the purpose of an integrated development environment (IDE)
- Download and install the JDK, JRE, and NetBeans IDE
- Import a project into NetBeans



Topics

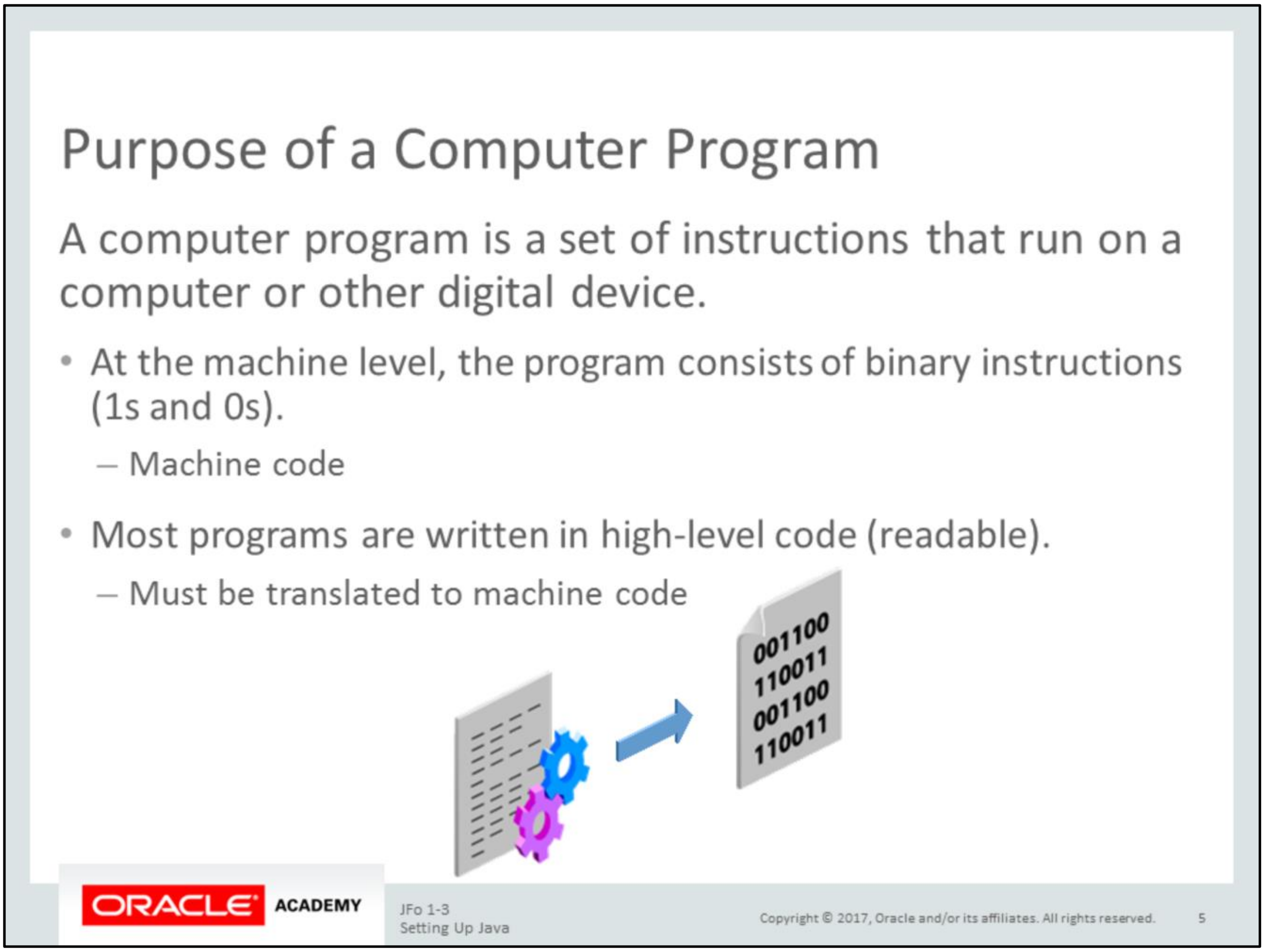
- Understanding the JRE, JDK, and NetBeans IDE
- Time to Set Up!

About This Course

Java: A Brief History

Setting Up Java

Section 1

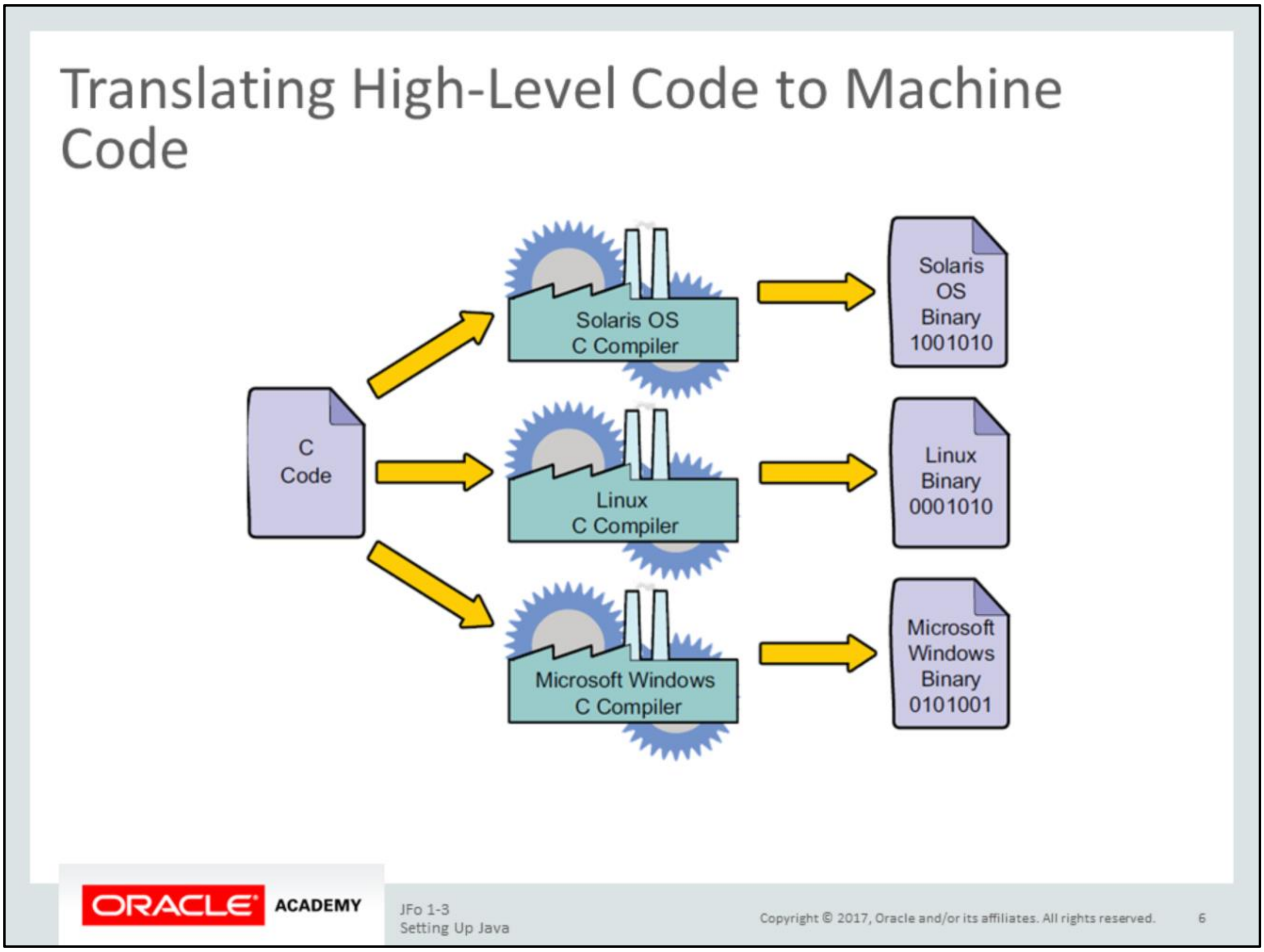


Purpose of a Computer Program

A computer program is a set of instructions that run on a computer or other digital device.

- At the machine level, the program consists of binary instructions (1s and 0s).
  - Machine code
- Most programs are written in high-level code (readable).
  - Must be translated to machine code

001100
110011
001100
110011



Translating High-Level Code to Machine Code

ORACLE ACADEMY

JFo 1-3
Setting Up Java

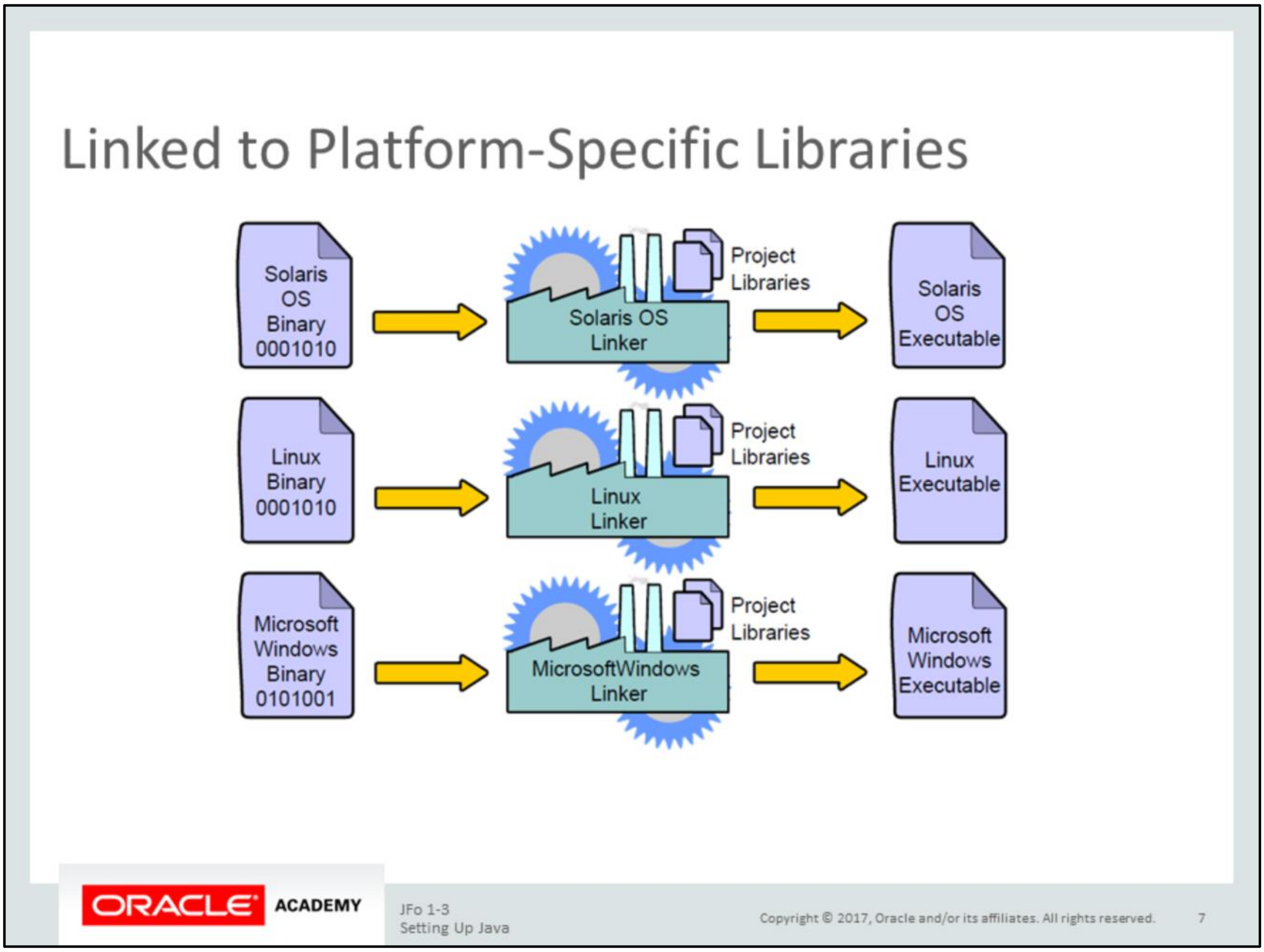
6

Programs written in most languages usually require numerous modifications to run on more than one type of computing platform (a combination of a CPU and an operating system). That’s because most languages require you to write code specific to the underlying platform.

Popular programming languages like C and C++ require programmers to compile and link their programs, resulting in an executable program that’s unique to a platform. A compiler is an application that converts a program that you write into a CPU-specific code called *machine code*. These platform-specific files (binary files) are often combined with other files, such as libraries of prewritten code. And a linker creates a platform-dependent program, called an *executable*, that an end user can execute.

Unlike C and C++, the Java programming language is platform-independent.

The image illustrates how a compiler creates a binary file.



Linked to Platform-Specific Libraries

Solaris OS Binary 0001010

Solaris OS Linker

Project Libraries

Solaris OS Executable

Linux Binary 0001010

Linux Linker

Project Libraries

Linux Executable

Microsoft Windows Binary 0101001

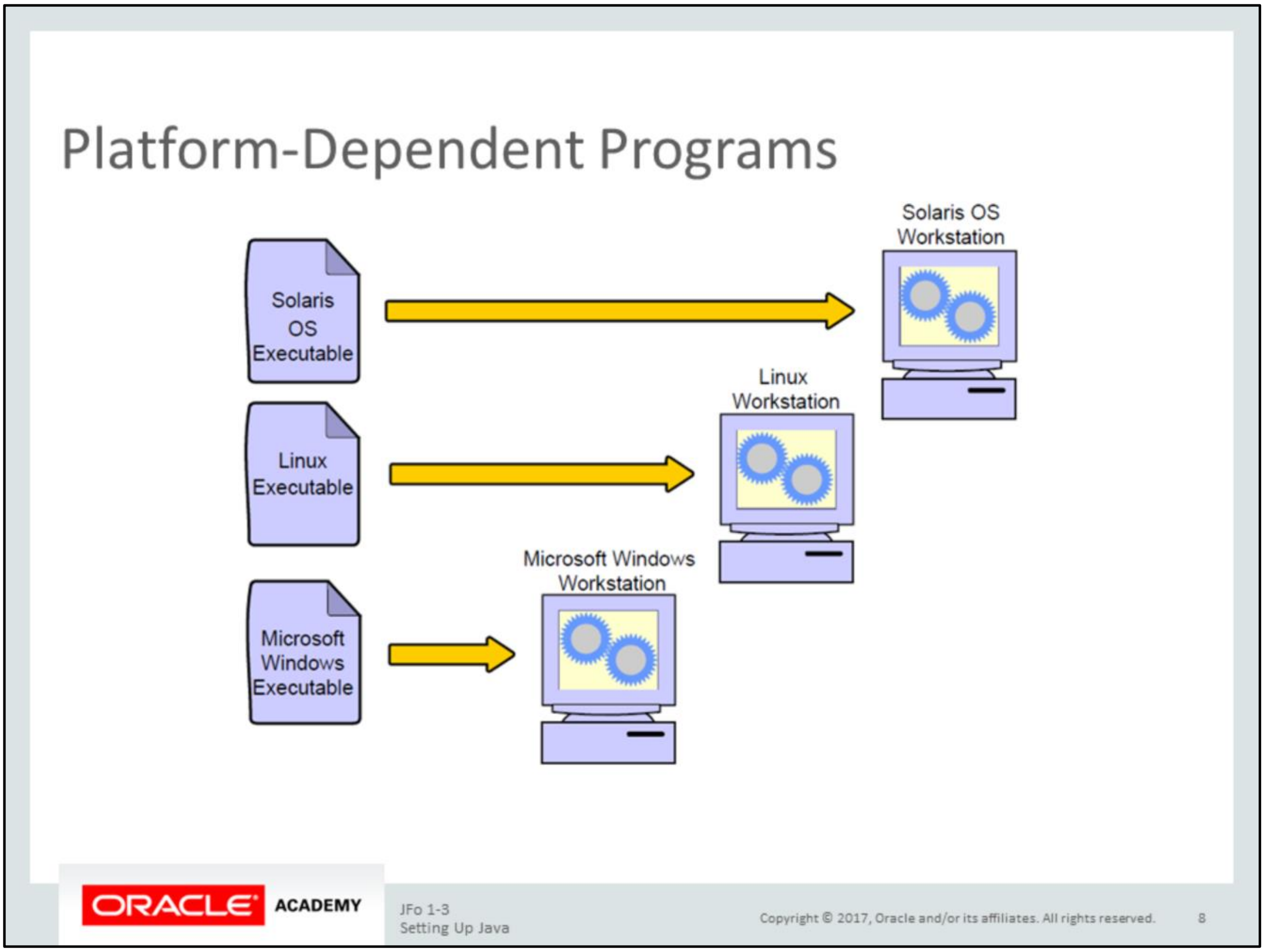
MicrosoftWindows Linker

Project Libraries

Microsoft Windows Executable

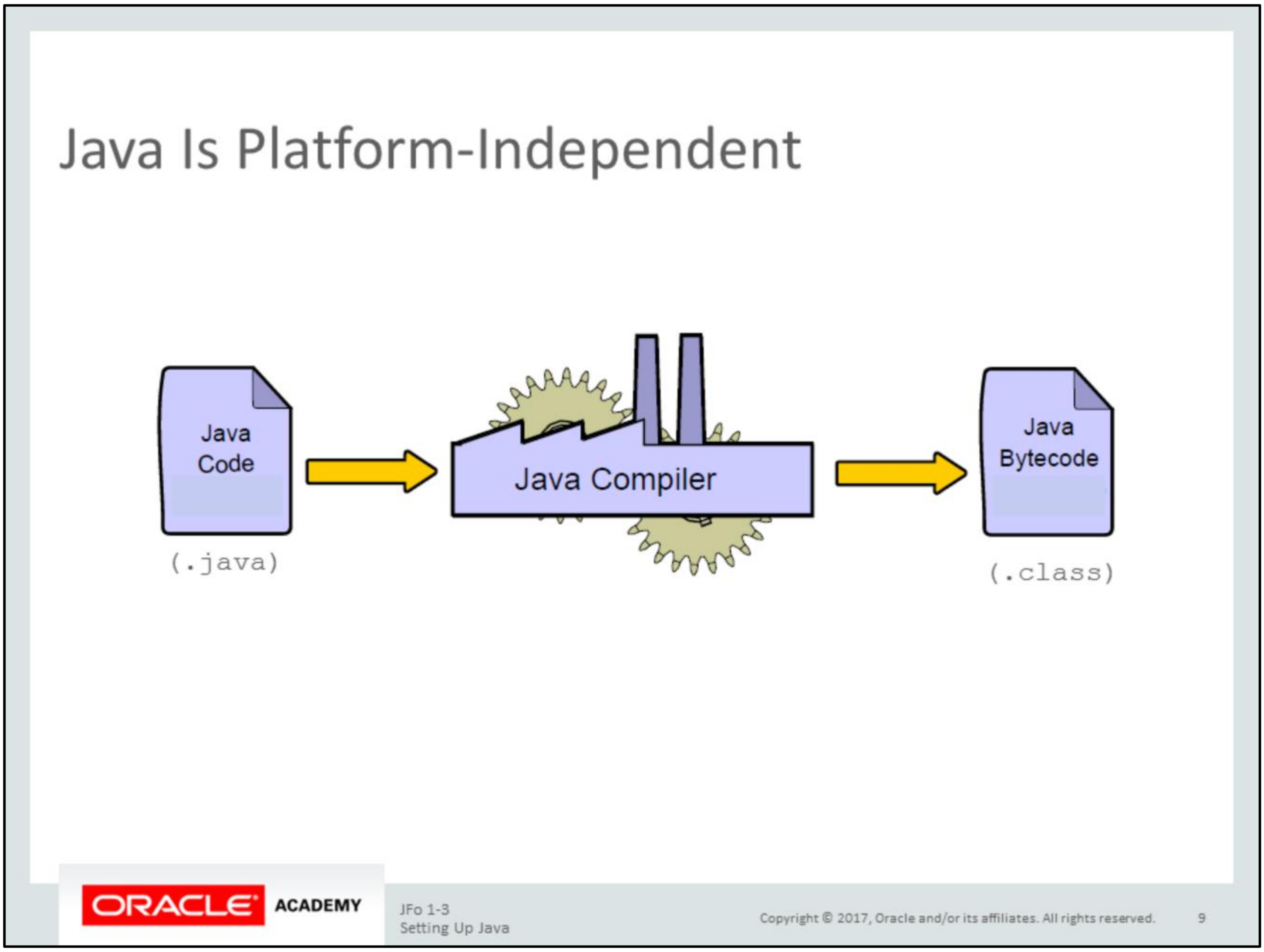
ORACLE ACADEMY

The image illustrates how a binary file is linked with libraries to create a platform-dependent executable.



Platform-Dependent Programs

The image illustrates how platform-dependent executables can execute only on one platform.



Java Is Platform-Independent

Java Code

(.java)

Java Compiler

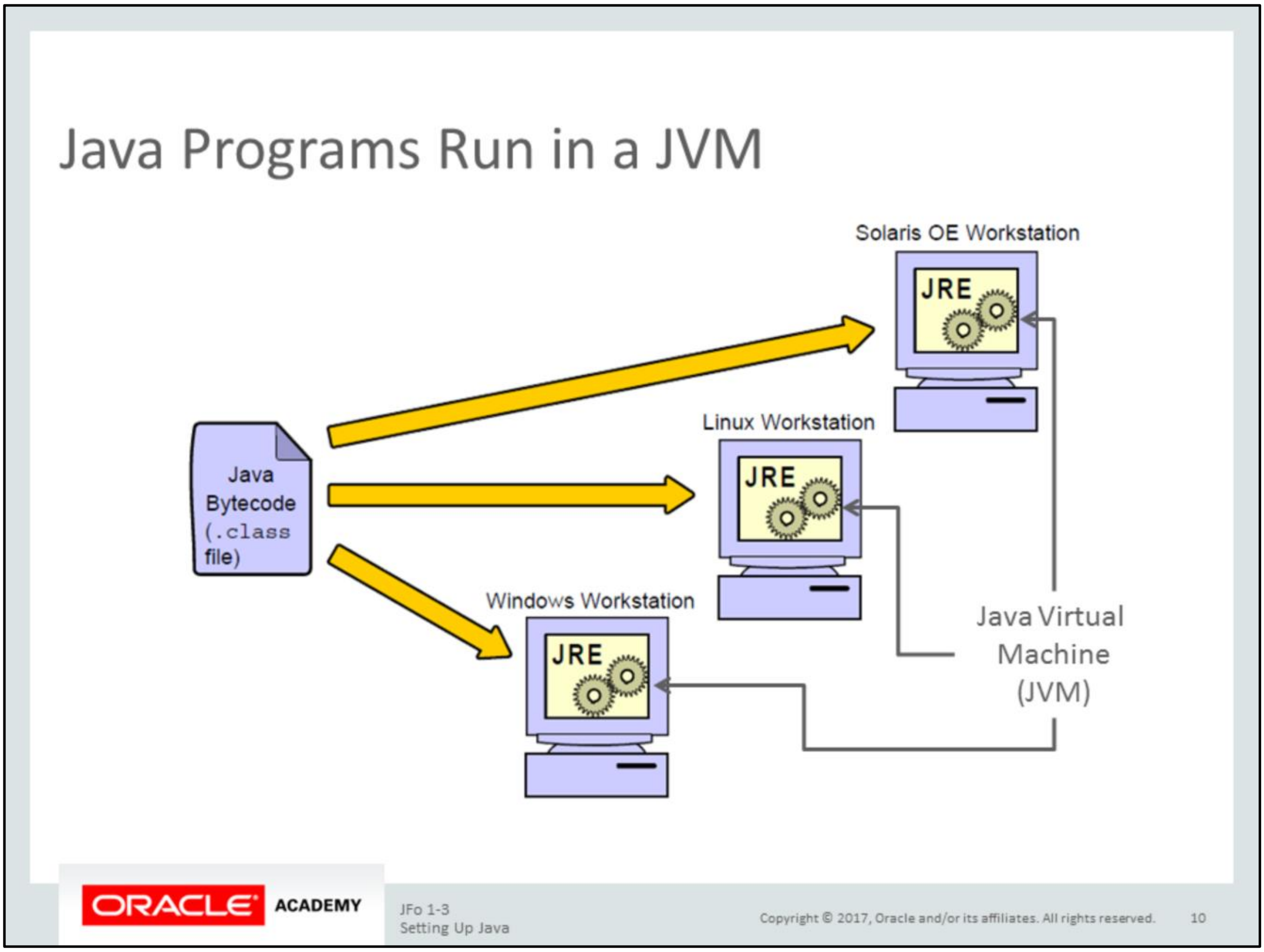
Java Bytecode

(.class)

A Java program can run on several different CPUs and operating system combinations, such as the Solaris OS on a SPARC chip, MacOS X on an Intel chip, and Microsoft Windows on an Intel chip, usually with few or no modifications.

As illustrated in the image, Java programs are compiled with a Java compiler. The resulting format of a compiled Java program is platform-independent Java bytecode instead of CPU-specific machine code.

The created bytecode is interpreted by a bytecode interpreter called the Java Virtual Machine (JVM). A virtual machine is a platform-specific program that understands platform-independent bytecode and can execute it on a particular platform. For this reason, the Java programming language is often referred to as an interpreted language, and Java technology programs are said to be portable or executable on any platform. Another interpreted language is Perl.



Java Programs Run in a JVM

ORACLE ACADEMY

JFo 1-3
Setting Up Java

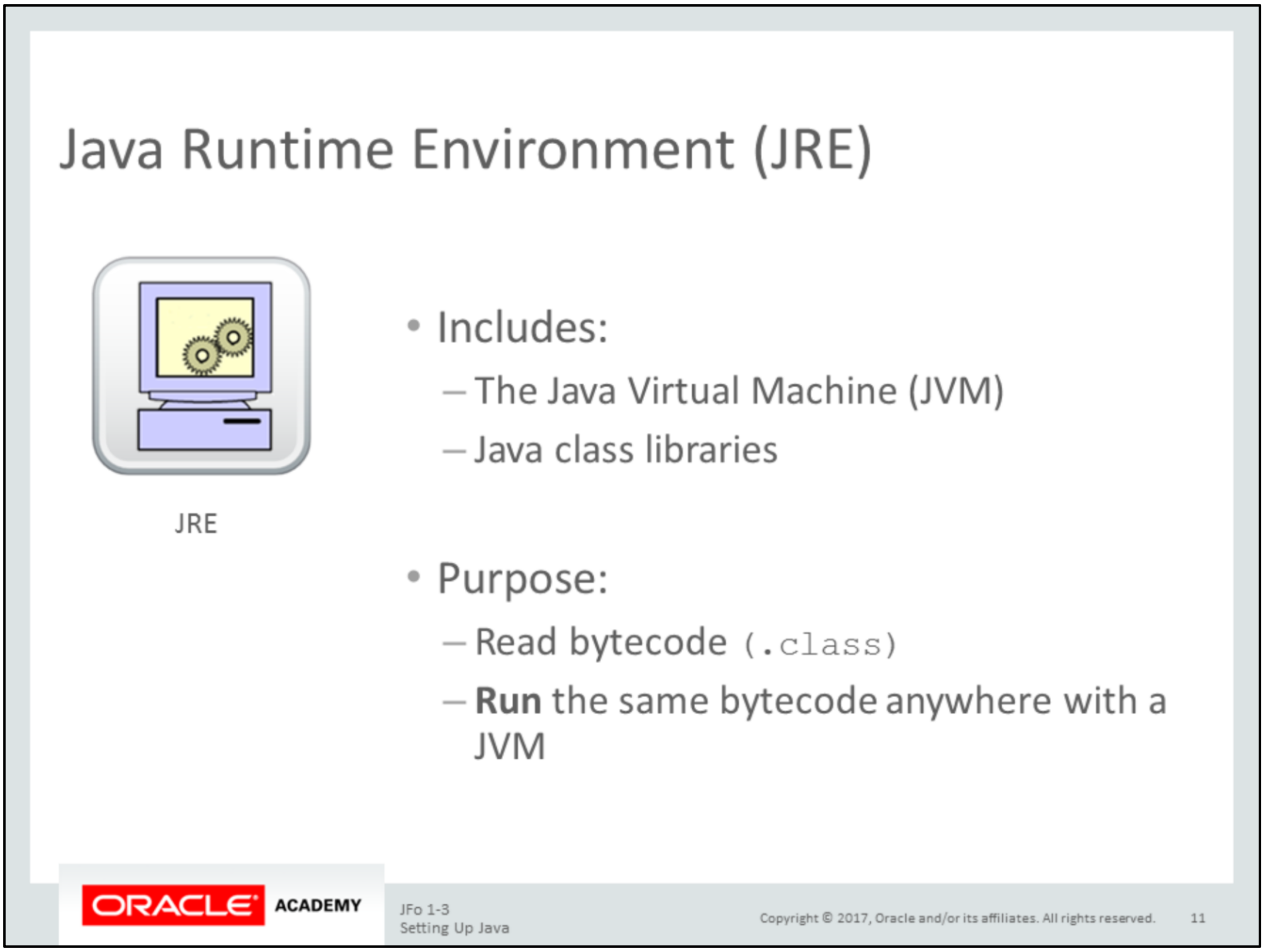
10

The image illustrates a Java bytecode file executing on several platforms where a Java runtime environment exists.

A virtual machine gets its name because it’s a piece of software that runs code, a task usually accomplished by the CPU or hardware machine. For Java programs to be platform-independent, the JVM is required on every platform where your program will run. The JVM is responsible for interpreting Java code, loading Java classes, and executing Java programs.

However, a Java program needs more than just a JVM to execute. It also needs a set of standard Java class libraries for the platform. Java class libraries are libraries of prewritten code that can be combined with the code that you write to create robust applications.

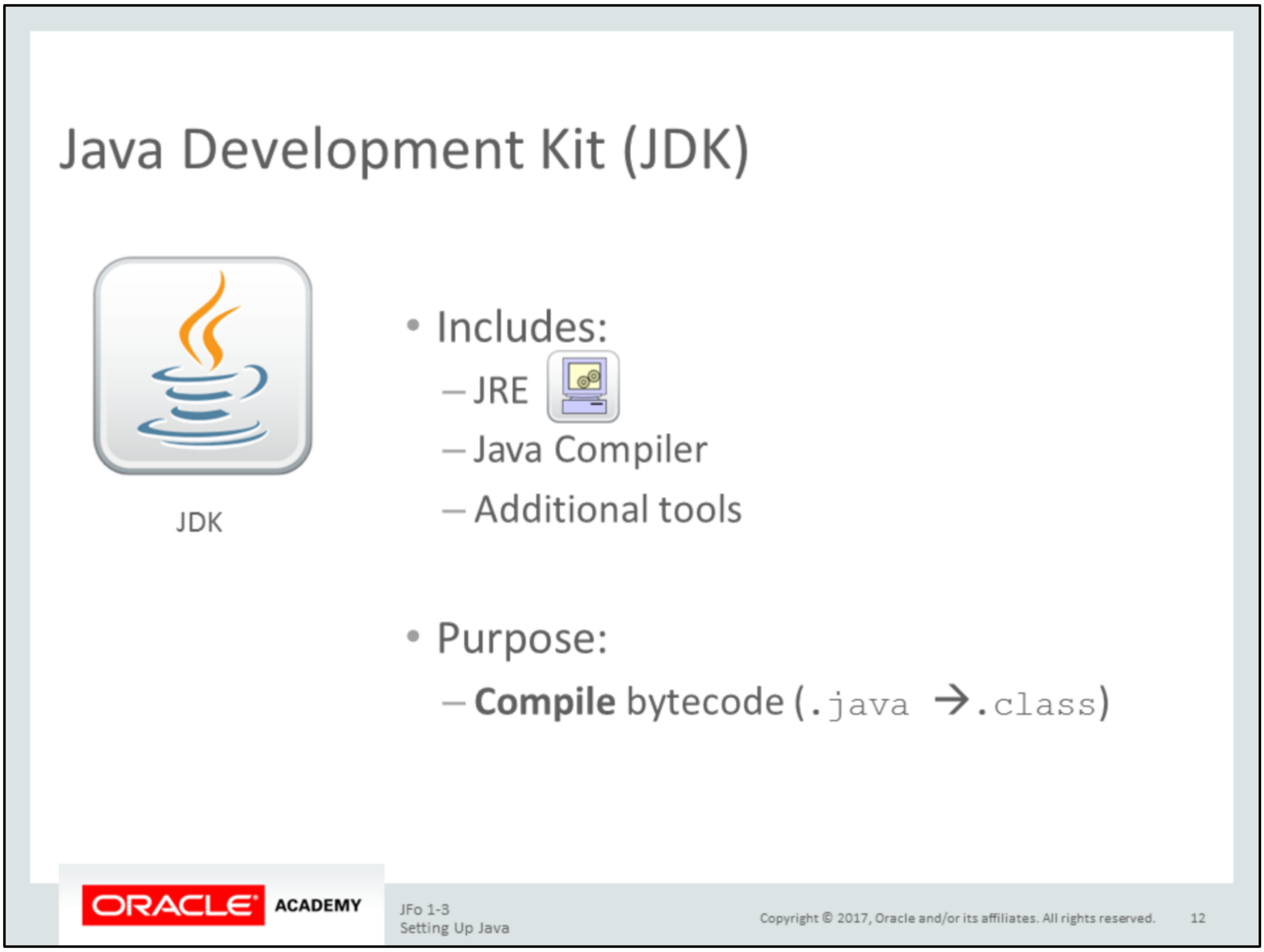
Combined, the JVM software and Java class libraries are referred to as the **Java Runtime Environment (JRE)**. JREs are available from Oracle for many common platforms.



Java Runtime Environment (JRE)

JRE

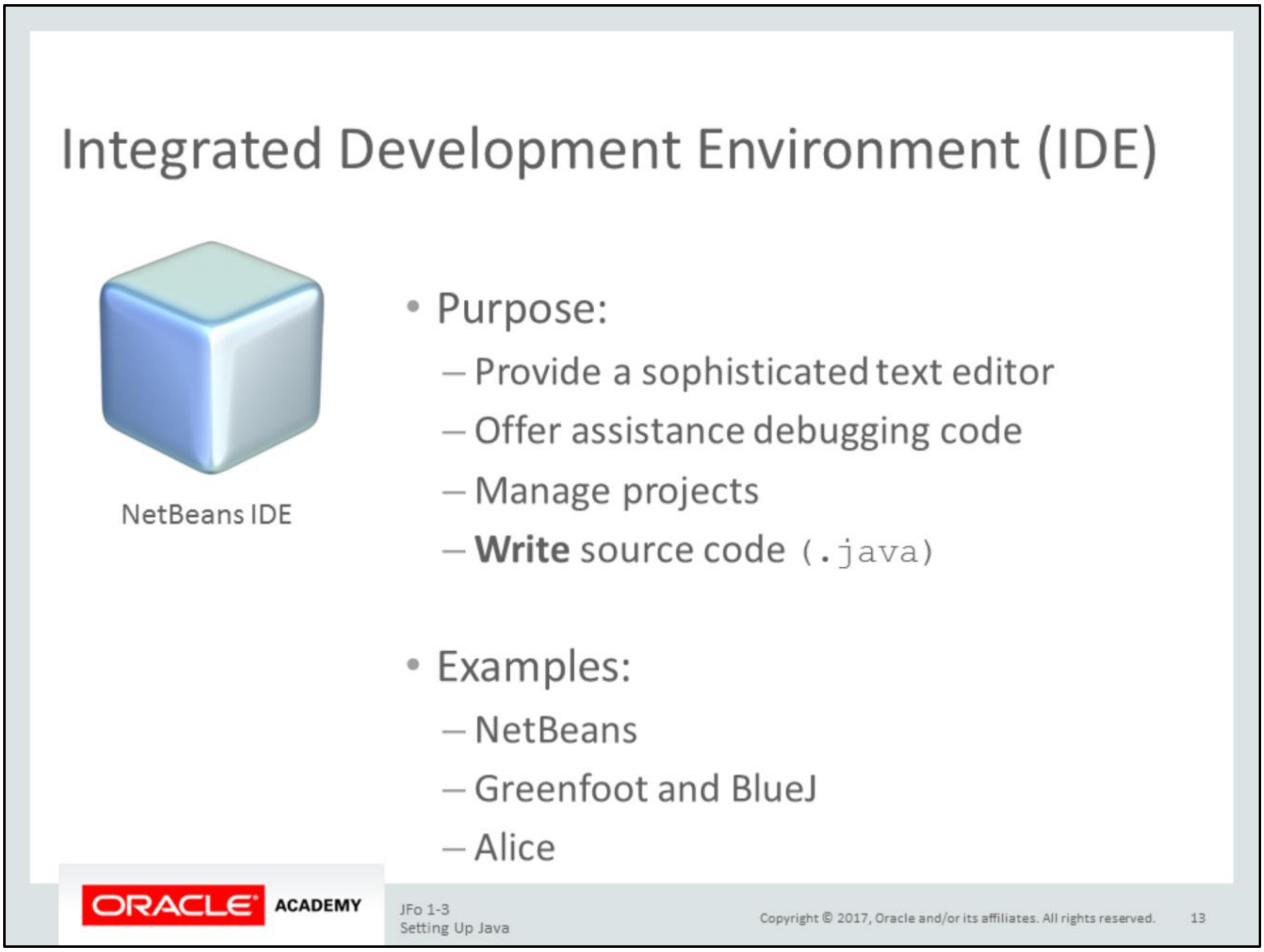
- Includes:
  - The Java Virtual Machine (JVM)
  - Java class libraries
- Purpose:
  - Read bytecode `(.class)`
  - **Run** the same bytecode anywhere with a JVM



Java Development Kit (JDK)

JDK

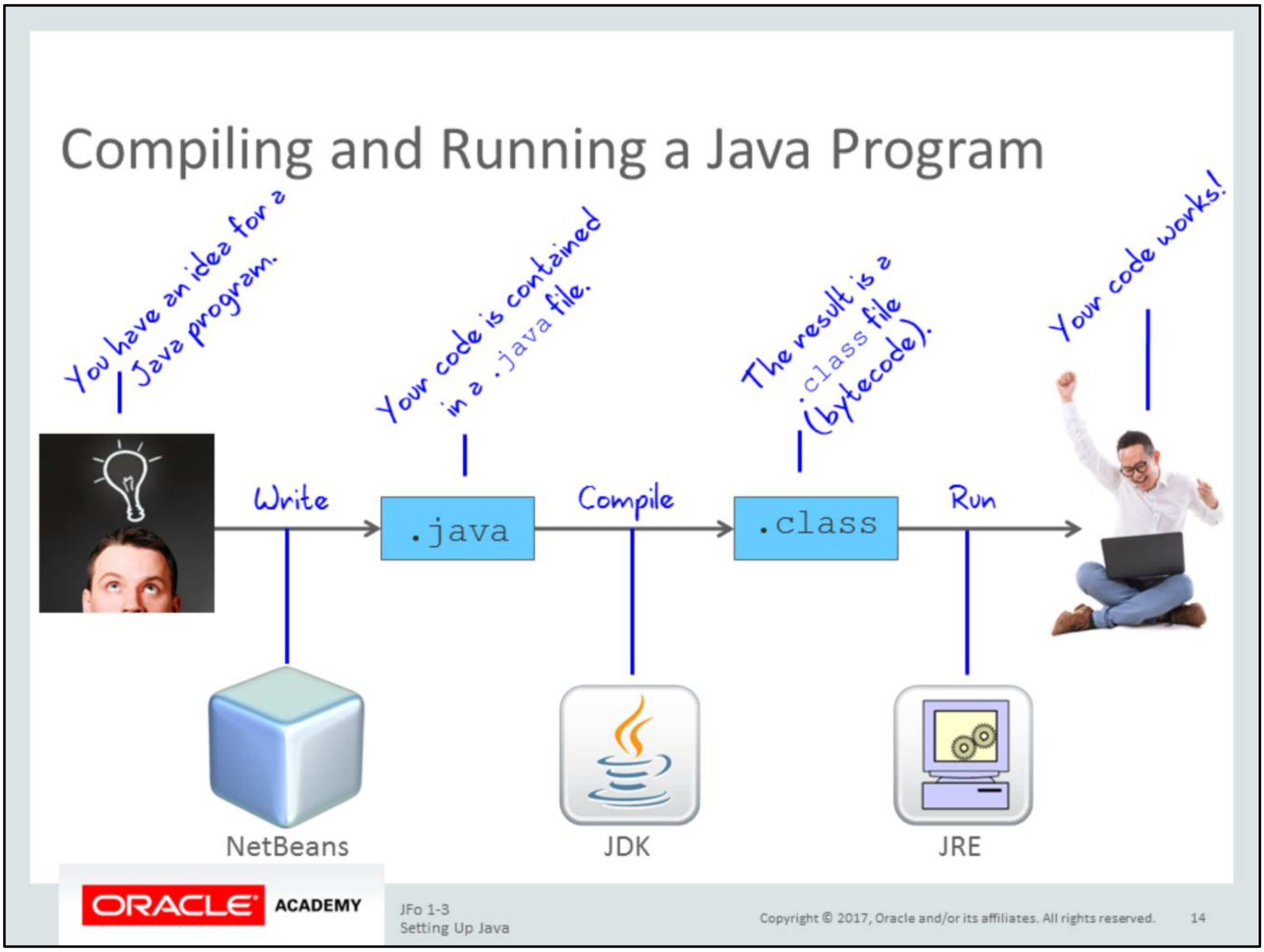
- Includes:
  - JRE
  - Java Compiler
  - Additional tools
- Purpose:
  - **Compile** bytecode (`.java` → `.class`)



Integrated Development Environment (IDE)

NetBeans IDE

- Purpose:
  - Provide a sophisticated text editor
  - Offer assistance debugging code
  - Manage projects
  - **Write** source code `(.java)`
- Examples:
  - NetBeans
  - Greenfoot and BlueJ
  - Alice



Compiling and Running a Java Program

You have an idea for a Java program.
Your code is contained in a .java file.
The result is a .class file (bytecode).
Your code works!
Write
.java
Compile
.class
Run
NetBeans
JDK
JRE

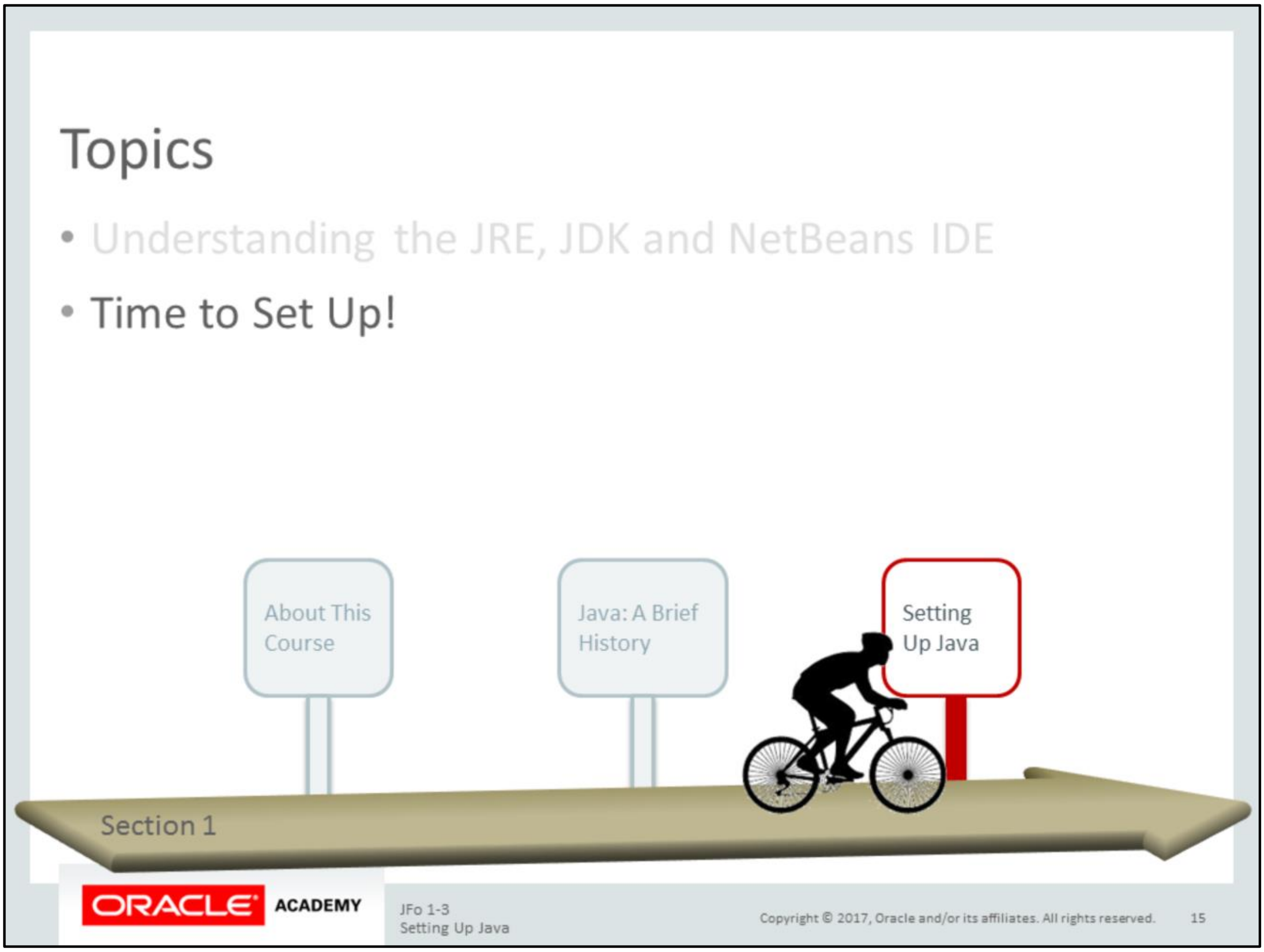
ORACLE ACADEMY

JFo 1-3
Setting Up Java

Copyright © 2017, Oracle and/or its affiliates. All rights reserved. 14

The diagram shows what happens when you compile and run a Java program:

- You have an idea for a Java program.
- The Java code is written in NetBeans, and the file has a `.java` extension. This is called the “Java source code.”
- The compiler component of the JDK compiles the source code into a bytecode file with a `.class` extension. This is called a Java class.
- The JVM component of the JRE runs the Java class. This is your Java program.
- Celebrate triumphantly when your code works, because most of the time it won’t work. This diagram oversimplifies the debugging aspect of development.



Topics

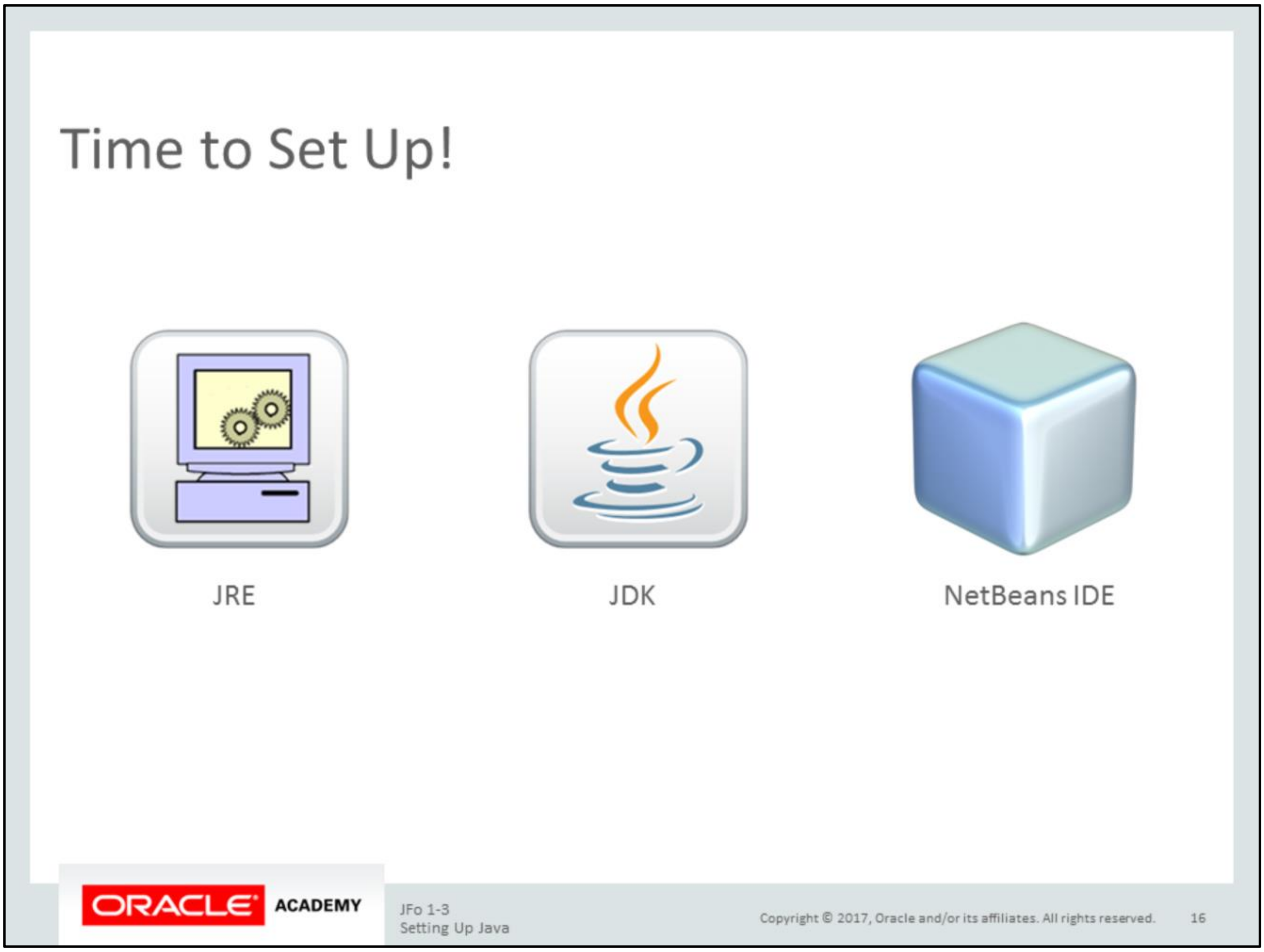
- Understanding the JRE, JDK and NetBeans IDE
- Time to Set Up!

About This Course

Java: A Brief History

Setting Up Java

Section 1



Time to Set Up!

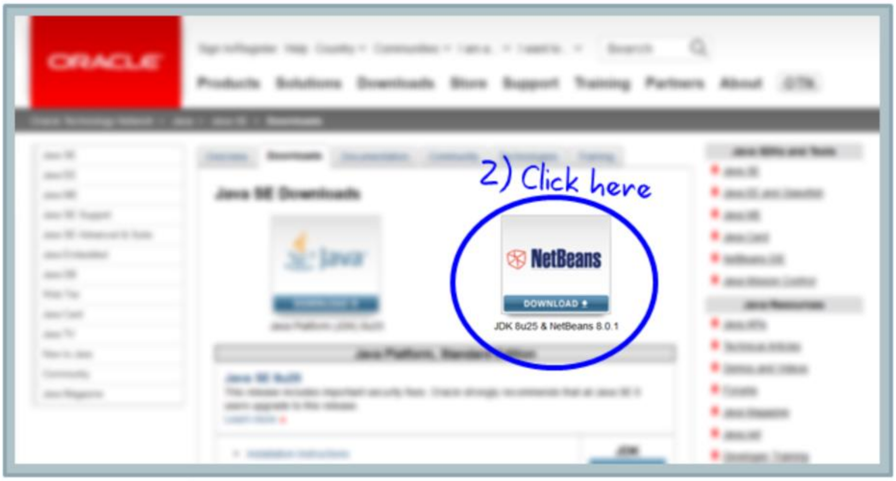
JRE

JDK

NetBeans IDE

Downloading JDK 8 & NetBeans

1. Go to the Java SE Downloads page at http://www.oracle.com/technetwork/java/javase/downloads/index.html.
2. Select the combined JDK & NetBeans.



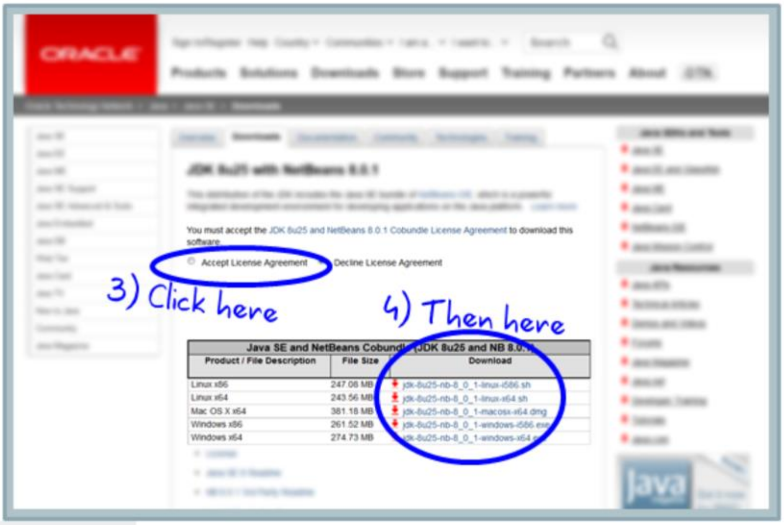
ORACLE
Products Solutions Downloads Store Support Training Partners About OTN
Java SE Downloads
Java
NetBeans
DOWNLOAD
JDK 8u25 & NetBeans 8.0.1

2) Click here

You’ll need Java SE Development Kit 8 (JDK 8) or later for this course.

Downloading JDK 8 & NetBeans

3. Accept the license agreement.
4. Select the correct download for your system.



JDK 8u25 with NetBeans 8.0.1

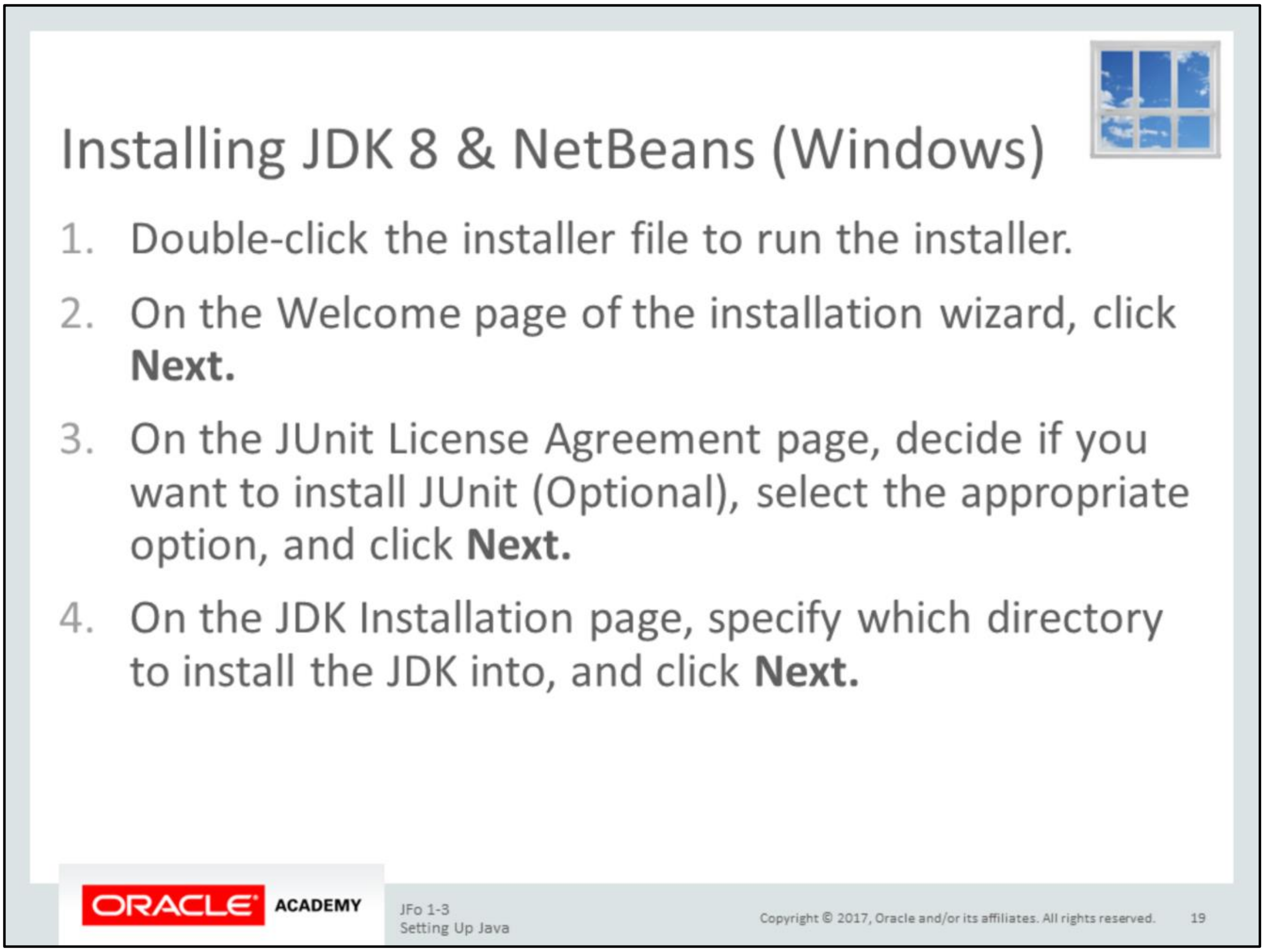
You must accept the JDK 8u25 and NetBeans 8.0.1 Cobundle License Agreement to download this software.

○ Accept License Agreement ○ Decline License Agreement

3) Click here

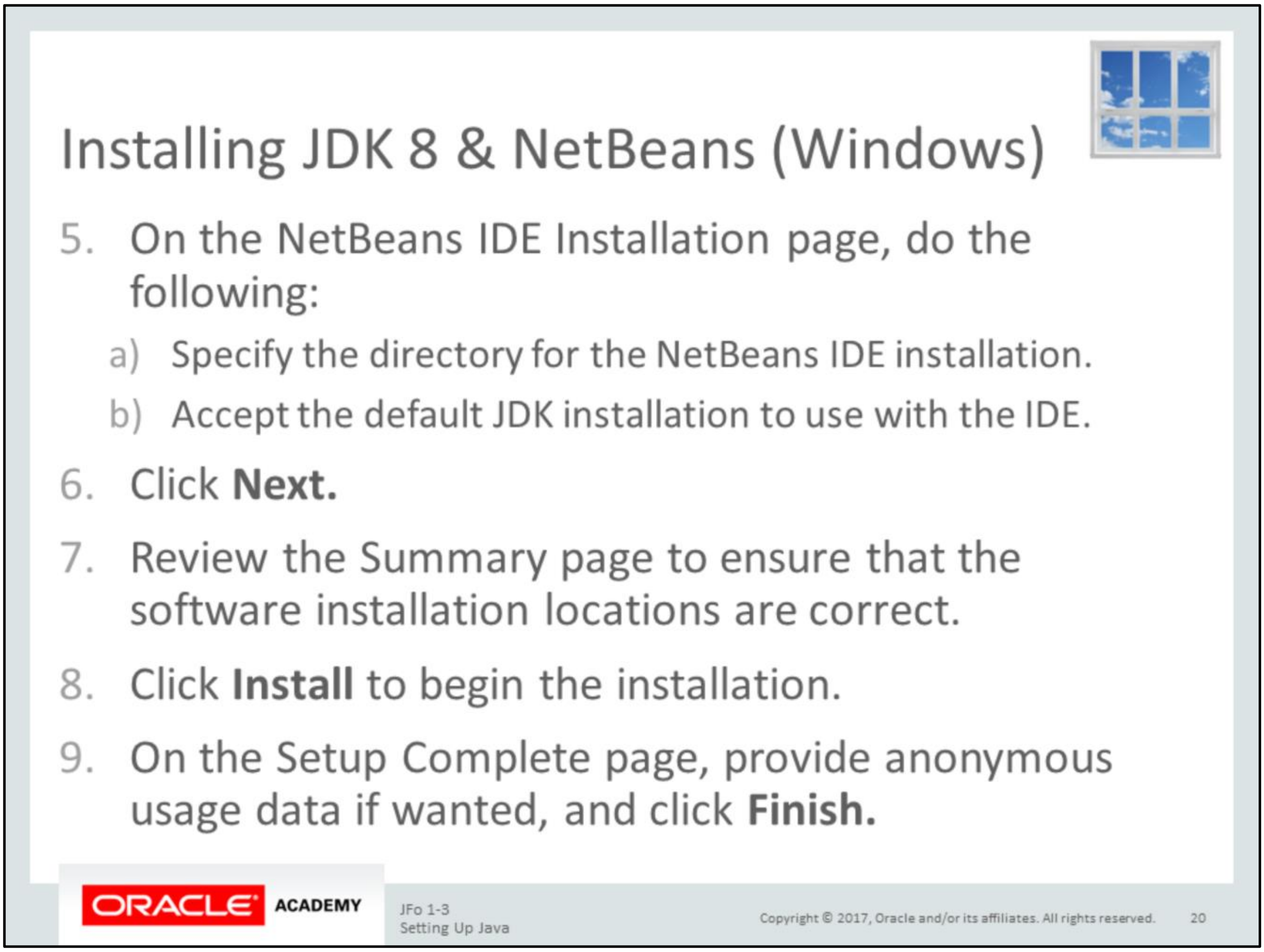
4) Then here

Java SE and NetBeans Cobundle (JDK 8u25 and NB 8.0.1)		
Product / File Description	File Size	Download
Linux x86	247.08 MB	jdk-8u25-nb-8_0_1-linux-i586.sh
Linux x64	243.56 MB	jdk-8u25-nb-8_0_1-linux-x64.sh
Mac OS X x64	381.18 MB	jdk-8u25-nb-8_0_1-macosx-x64.dmg
Windows x86	261.52 MB	jdk-8u25-nb-8_0_1-windows-i586.exe
Windows x64	274.73 MB	jdk-8u25-nb-8_0_1-windows-x64.exe



Installing JDK 8 & NetBeans (Windows)

1. Double-click the installer file to run the installer.
2. On the Welcome page of the installation wizard, click **Next.**
3. On the JUnit License Agreement page, decide if you want to install JUnit (Optional), select the appropriate option, and click **Next.**
4. On the JDK Installation page, specify which directory to install the JDK into, and click **Next.**

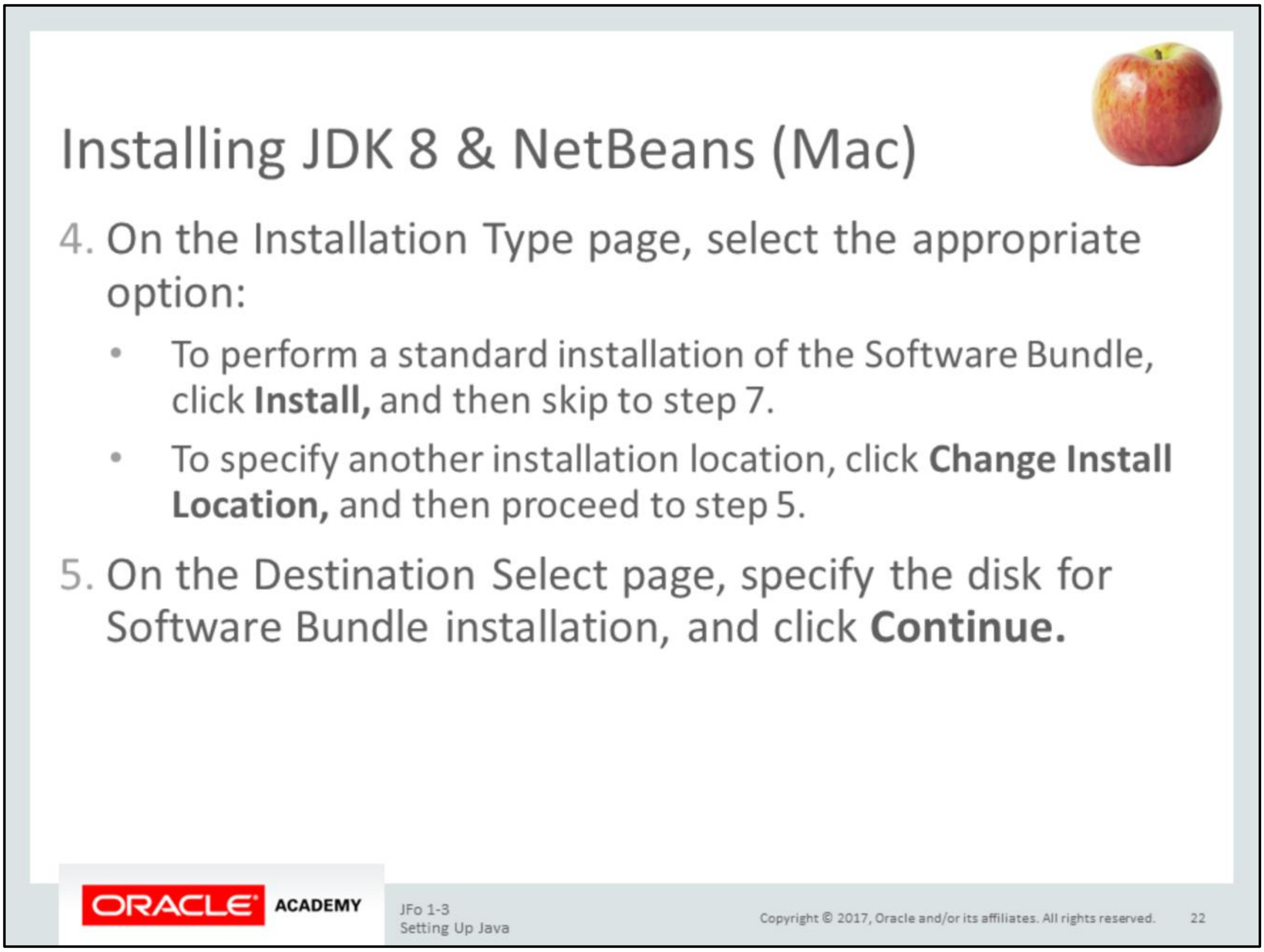


Installing JDK 8 & NetBeans (Windows)

5. On the NetBeans IDE Installation page, do the following:
   a) Specify the directory for the NetBeans IDE installation.
   b) Accept the default JDK installation to use with the IDE.
6. Click **Next.**
7. Review the Summary page to ensure that the software installation locations are correct.
8. Click **Install** to begin the installation.
9. On the Setup Complete page, provide anonymous usage data if wanted, and click **Finish.**

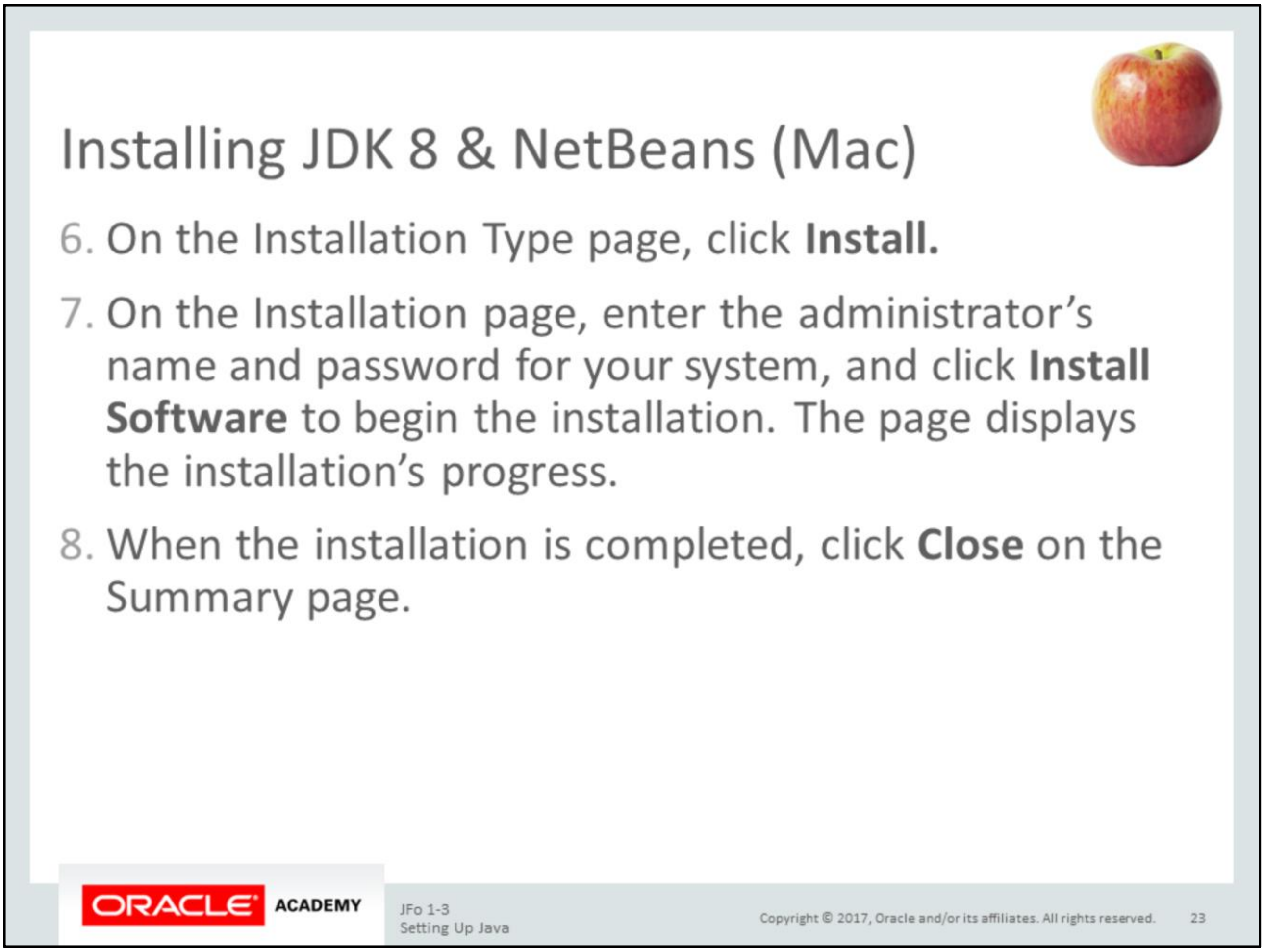
Installing JDK 8 & NetBeans (Mac)

1. After the download is completed, run the installer. The installer file has the `.dmg` extension.
2. On the panel that opens, double-click the package icon. The package has the `.mpkg` extension. The installation wizard starts.
3. On the Introduction page of the installation wizard, click **Continue.**



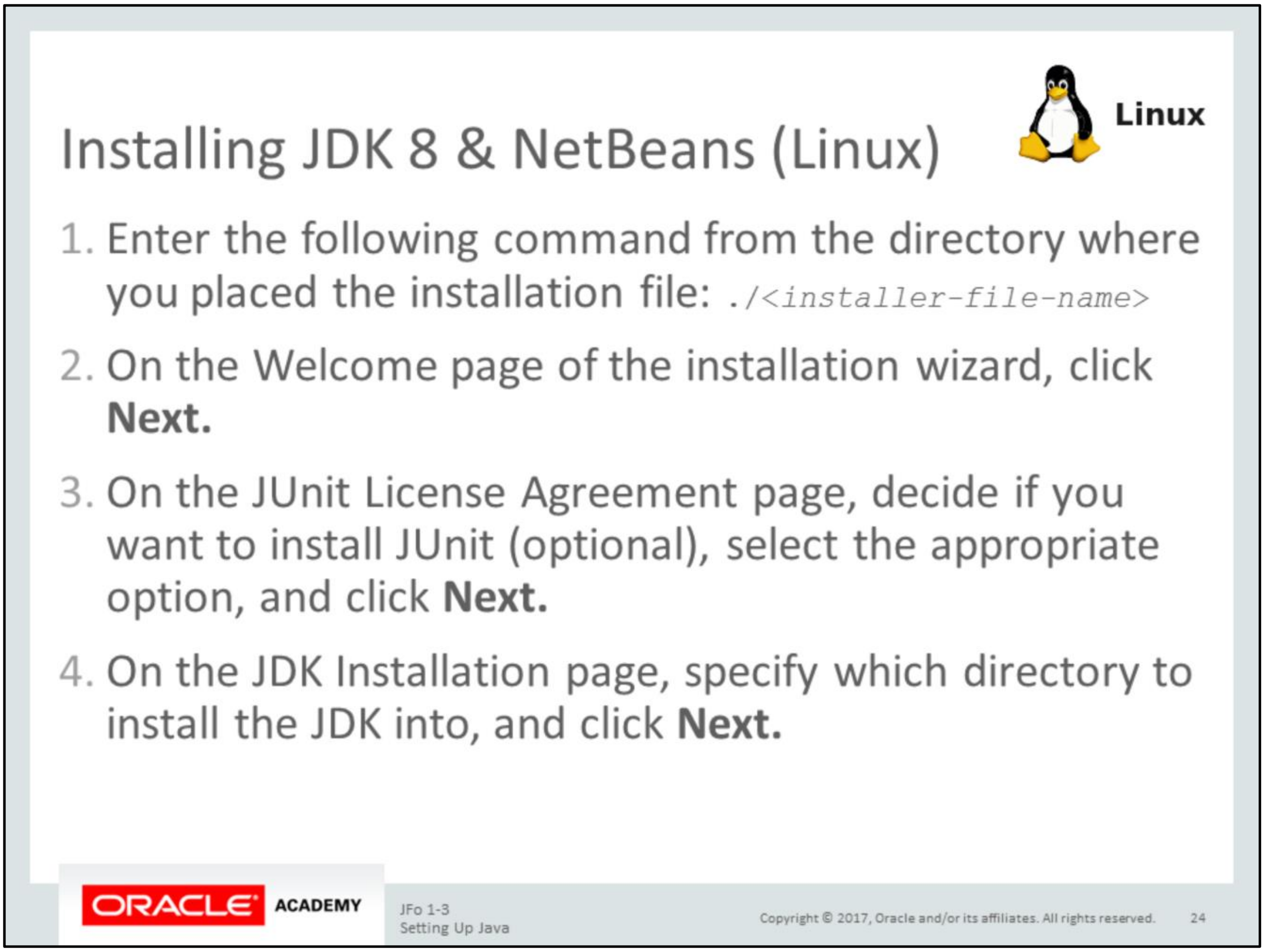
Installing JDK 8 & NetBeans (Mac)

4. On the Installation Type page, select the appropriate option:
   - To perform a standard installation of the Software Bundle, click **Install,** and then skip to step 7.
   - To specify another installation location, click **Change Install Location,** and then proceed to step 5.
5. On the Destination Select page, specify the disk for Software Bundle installation, and click **Continue.**



Installing JDK 8 & NetBeans (Mac)

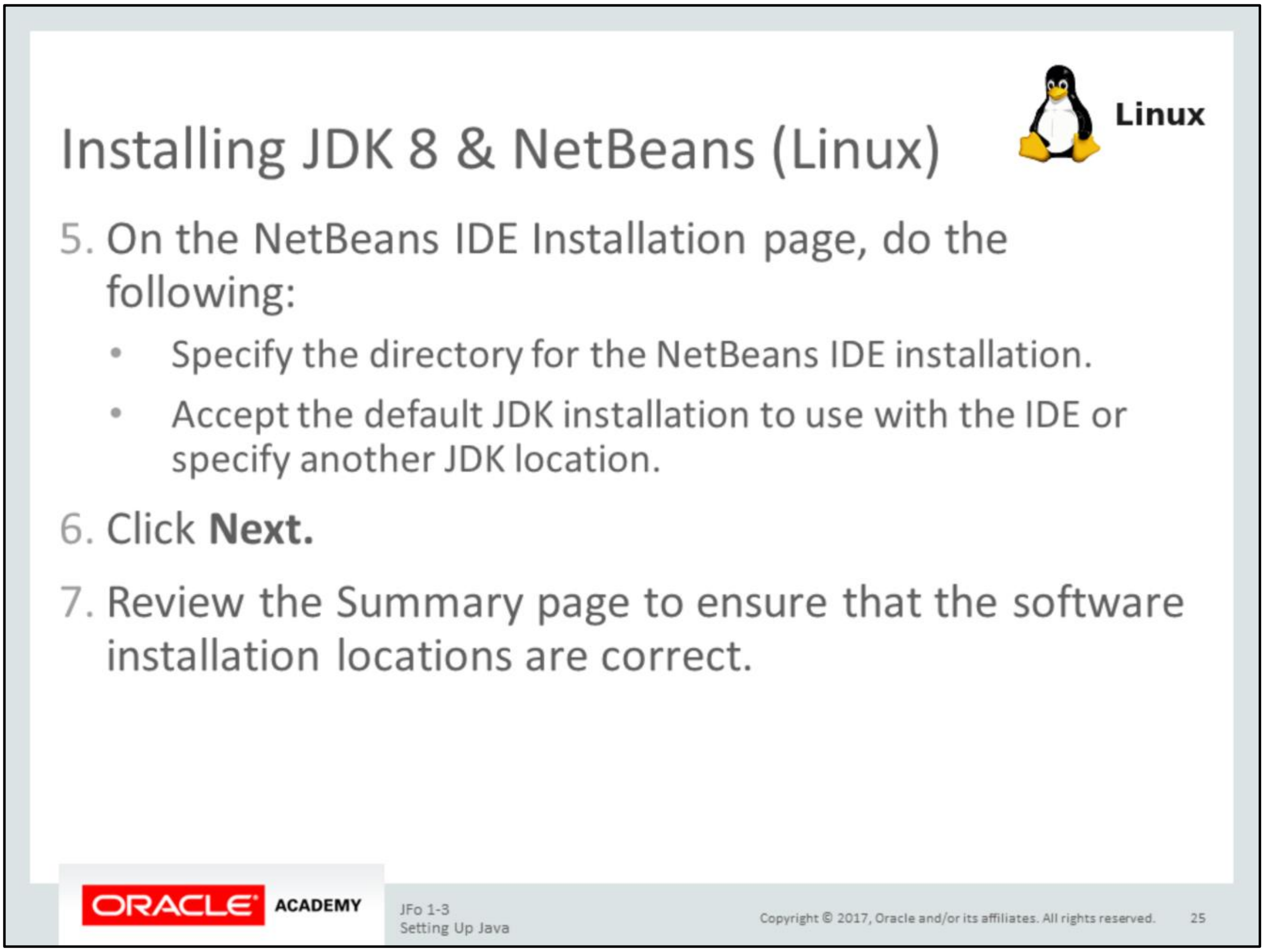
6. On the Installation Type page, click **Install.**
7. On the Installation page, enter the administrator's name and password for your system, and click **Install Software** to begin the installation. The page displays the installation's progress.
8. When the installation is completed, click **Close** on the Summary page.



Linux

Installing JDK 8 & NetBeans (Linux)

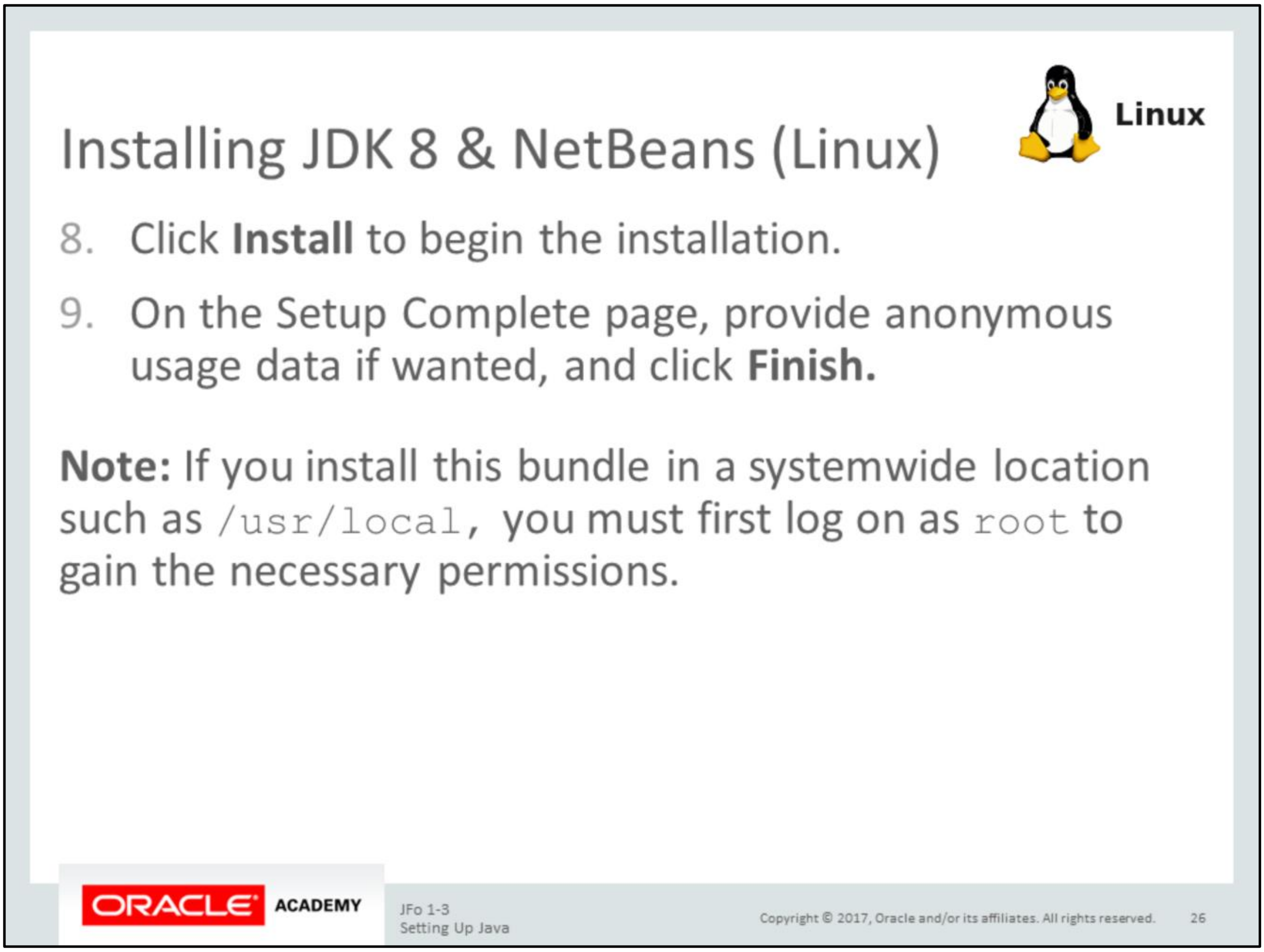
1. Enter the following command from the directory where you placed the installation file: `./<installer-file-name>`
2. On the Welcome page of the installation wizard, click **Next.**
3. On the JUnit License Agreement page, decide if you want to install JUnit (optional), select the appropriate option, and click **Next.**
4. On the JDK Installation page, specify which directory to install the JDK into, and click **Next.**



Installing JDK 8 & NetBeans (Linux)

Linux

5. On the NetBeans IDE Installation page, do the following:
   - Specify the directory for the NetBeans IDE installation.
   - Accept the default JDK installation to use with the IDE or specify another JDK location.
6. Click **Next.**
7. Review the Summary page to ensure that the software installation locations are correct.



Linux

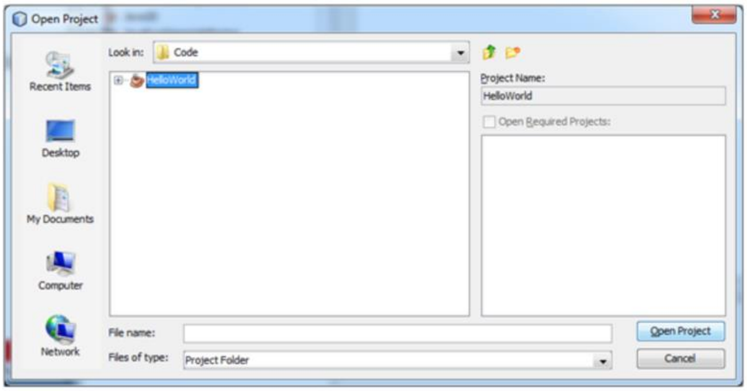
Installing JDK 8 & NetBeans (Linux)

8. Click **Install** to begin the installation.
9. On the Setup Complete page, provide anonymous usage data if wanted, and click **Finish.**

Note: If you install this bundle in a systemwide location such as `/usr/local`, you must first log on as `root` to gain the necessary permissions.

Testing NetBeans

1. Download and unzip the `HelloWorld` project: HelloWorld .zip
2. Launch NetBeans: NetBeans IDE 8.0
3. Select **File > Open Project,** and then select **HelloWorld.**



Open Project

Look in: Code

HelloWorld

Project Name: HelloWorld

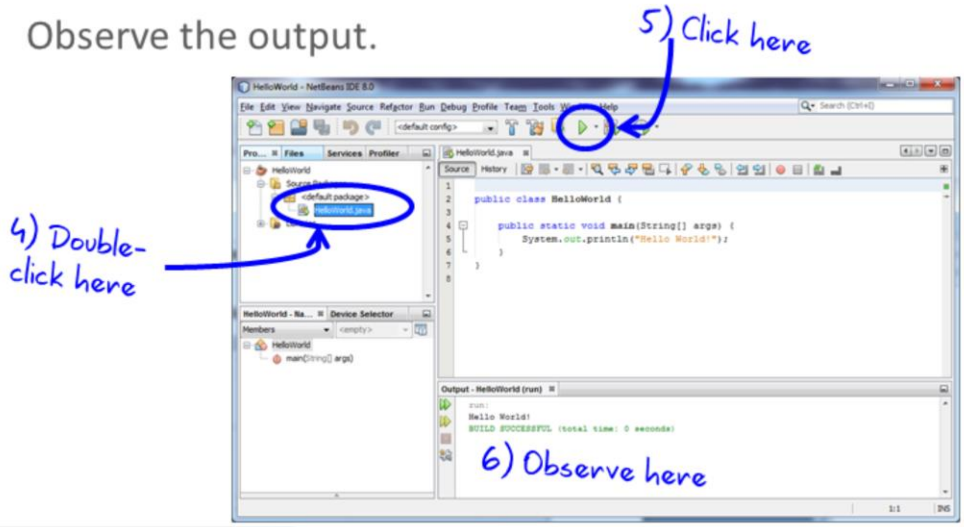
□ Open Required Projects:

File name:

Files of type: Project Folder

Testing NetBeans

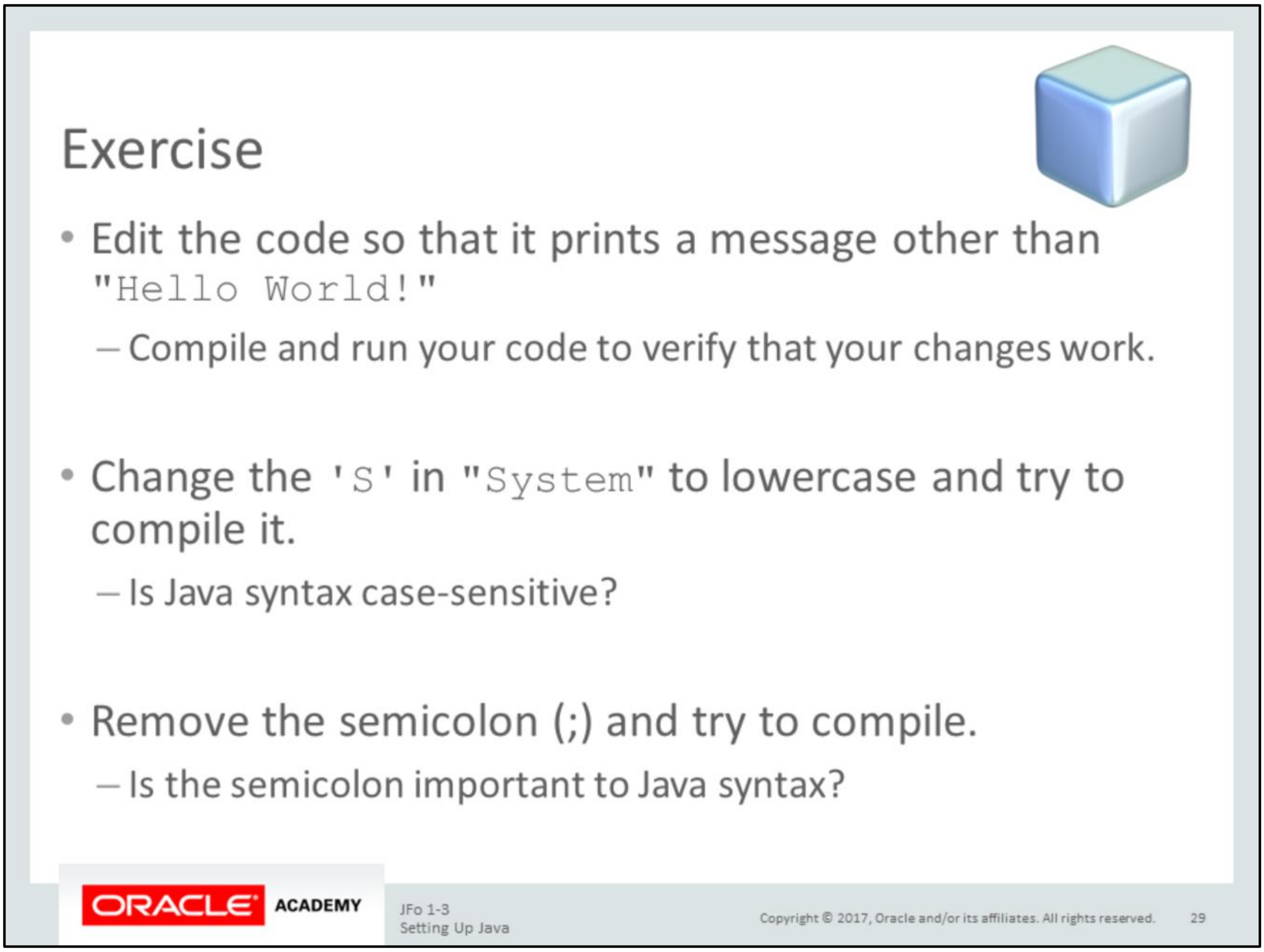
4. Expand the project and open `HelloWorld.java.`
5. Compile and run the project.
6. Observe the output.



```
public class HelloWorld {

    public static void main(String[] args) {
        System.out.println("Hello World!");
    }
}
```

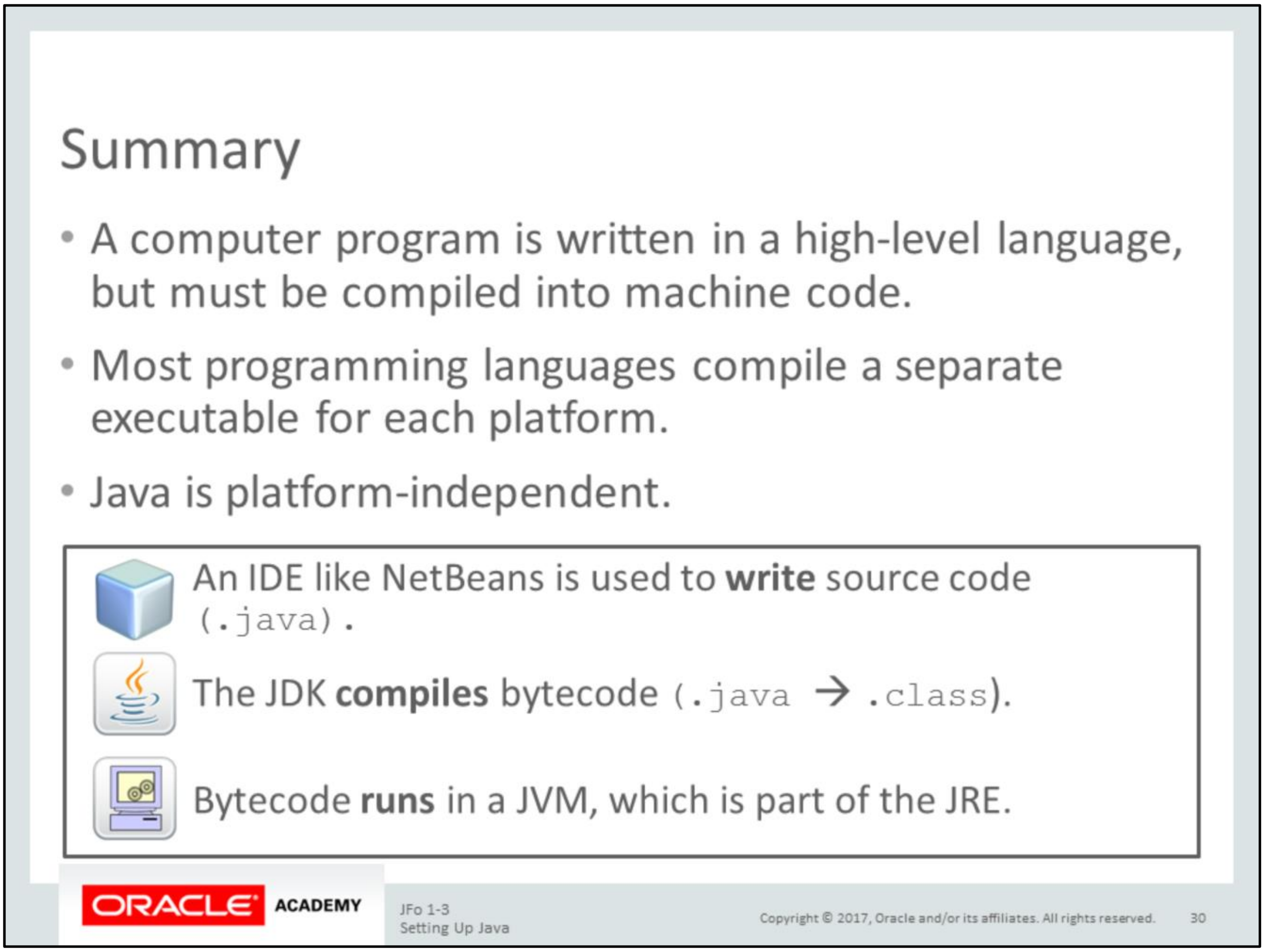
```
run:
Hello World!
BUILD SUCCESSFUL (total time: 0 seconds)
```



Exercise

- Edit the code so that it prints a message other than `"Hello World!"`
  - Compile and run your code to verify that your changes work.
- Change the `'S'` in `"System"` to lowercase and try to compile it.
  - Is Java syntax case-sensitive?
- Remove the semicolon (;) and try to compile.
  - Is the semicolon important to Java syntax?

Note to Instructors: Print statements are formally introduced in Section 2. The goal of this exercise to check that everything is installed properly. The exercise also offers students a chance to learn through experimentation, which should make them more receptive when print statements are discussed later.



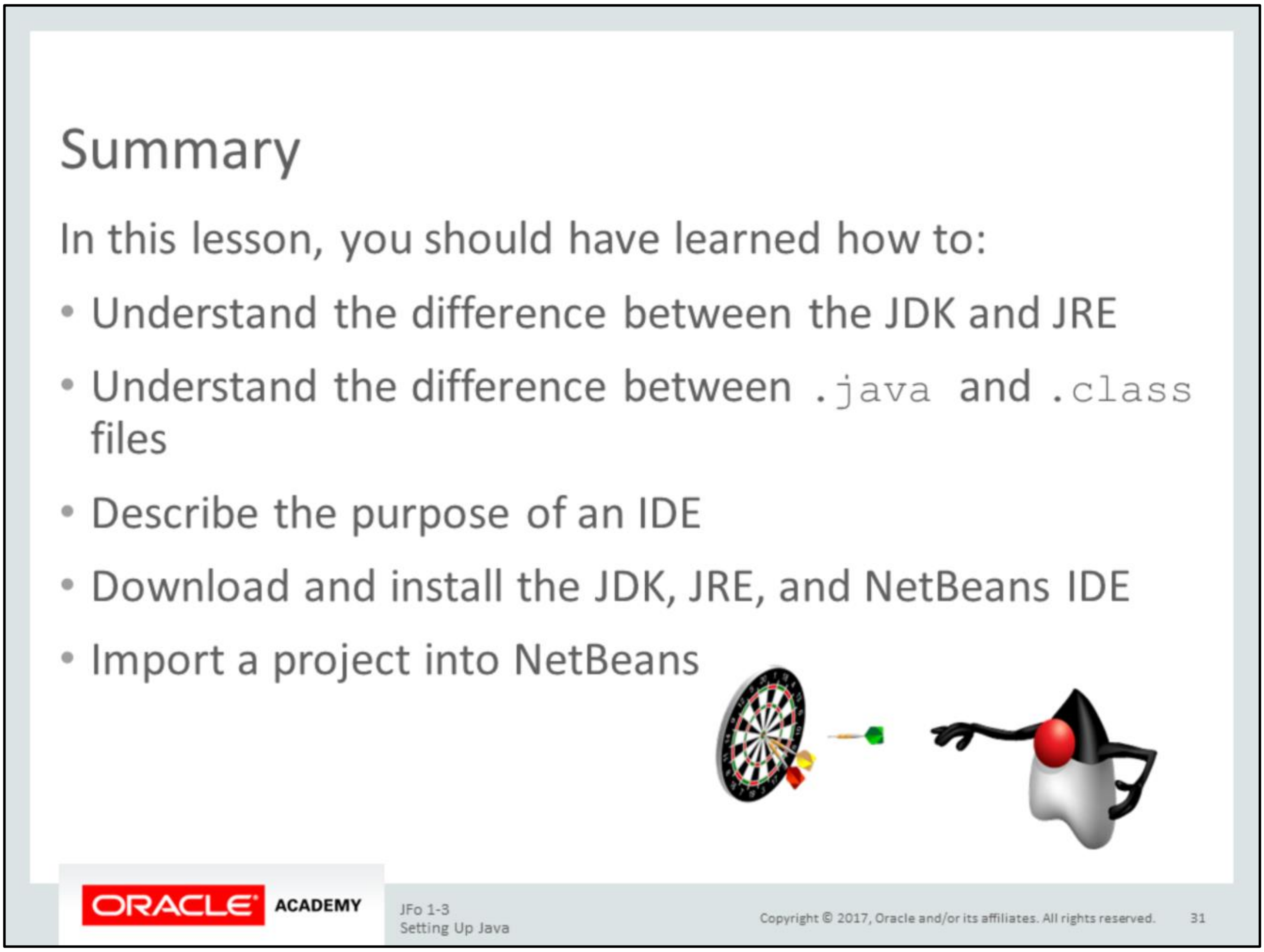
Summary

- A computer program is written in a high-level language, but must be compiled into machine code.
- Most programming languages compile a separate executable for each platform.
- Java is platform-independent.

An IDE like NetBeans is used to **write** source code (`.java`).

The JDK **compiles** bytecode (`.java` → `.class`).

Bytecode **runs** in a JVM, which is part of the JRE.



Summary

In this lesson, you should have learned how to:

- Understand the difference between the JDK and JRE
- Understand the difference between `.java` and `.class` files
- Describe the purpose of an IDE
- Download and install the JDK, JRE, and NetBeans IDE
- Import a project into NetBeans

ORACLE® ACADEMY