ORACLE

Academy

Java Foundations

1-3

Setting Up Java

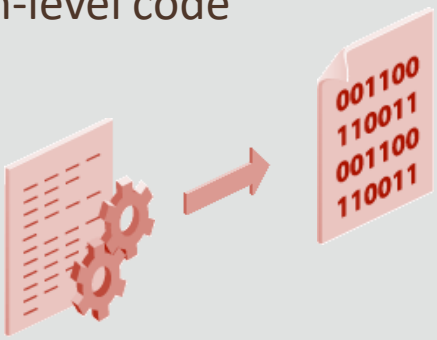
ORACLE
Academy

Objectives

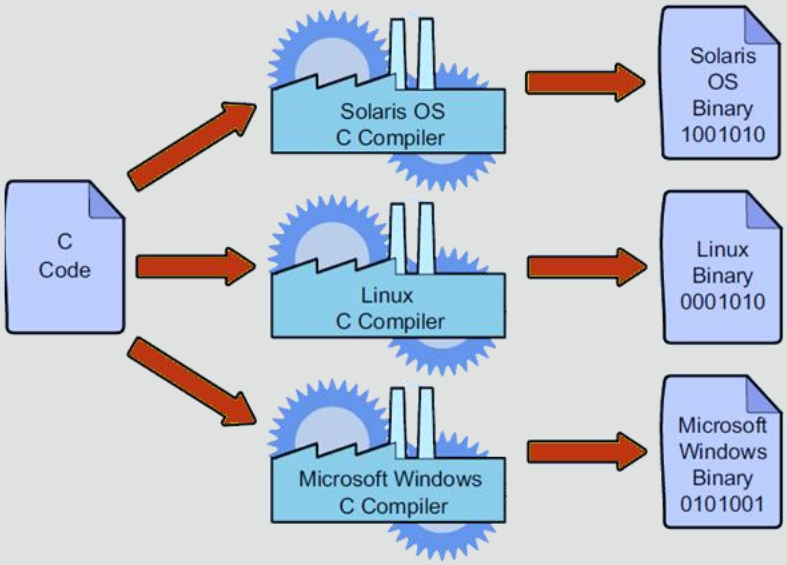
- This lesson covers the following objectives:
  - Understand the difference between the JDK and JRE
  - Understand the difference between .java and .class files
  - Describe the purpose of an integrated development environment (IDE)
  - Add an existing .java file into a Java project

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



Translating High-Level Code to Machine Code



ORACLE
Academy

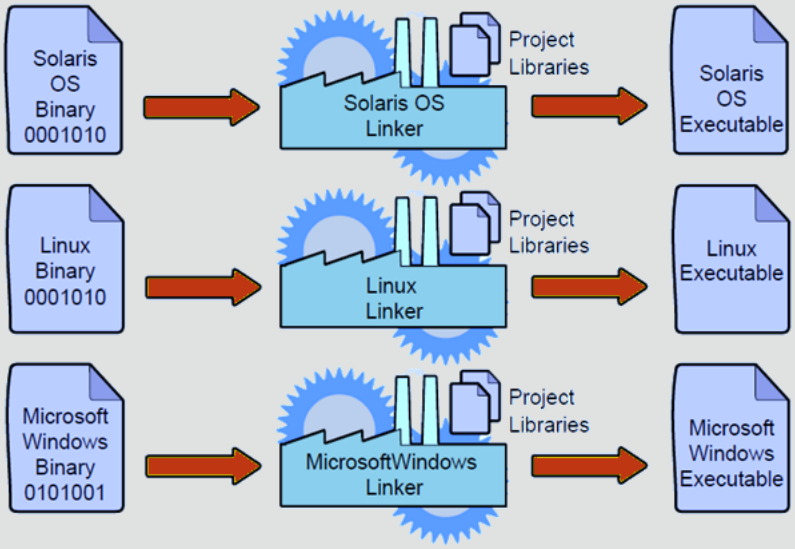
Programs written in most languages usually require numerous modifications to run on more than one type of computing platform (a combination of a CPU and an operating system).

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.

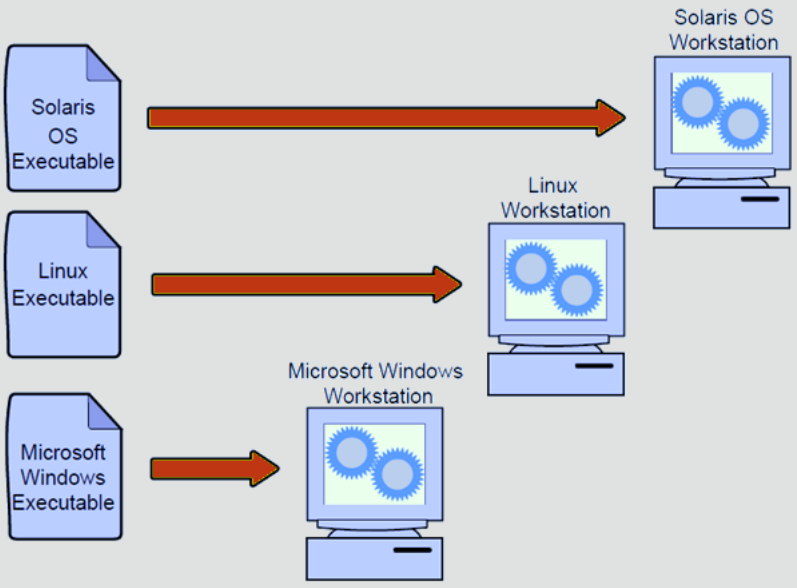
The image illustrates how a compiler creates a binary file.

Linked to Platform-Specific Libraries



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

Platform-Dependent Programs

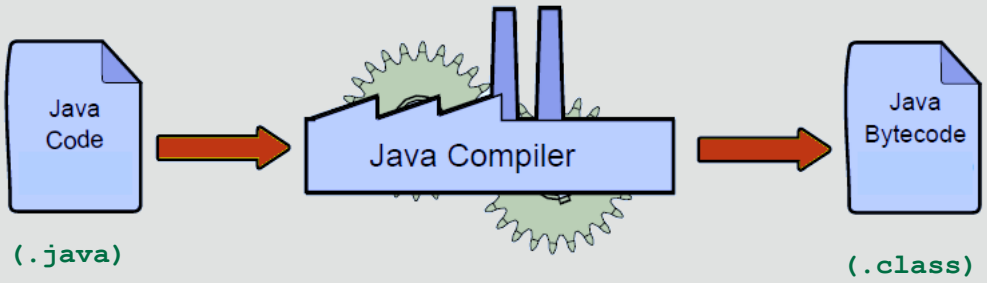


ORACLE Academy

JFo 1-3
Setting up Java

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

Java Is Platform-Independent



ORACLE Academy

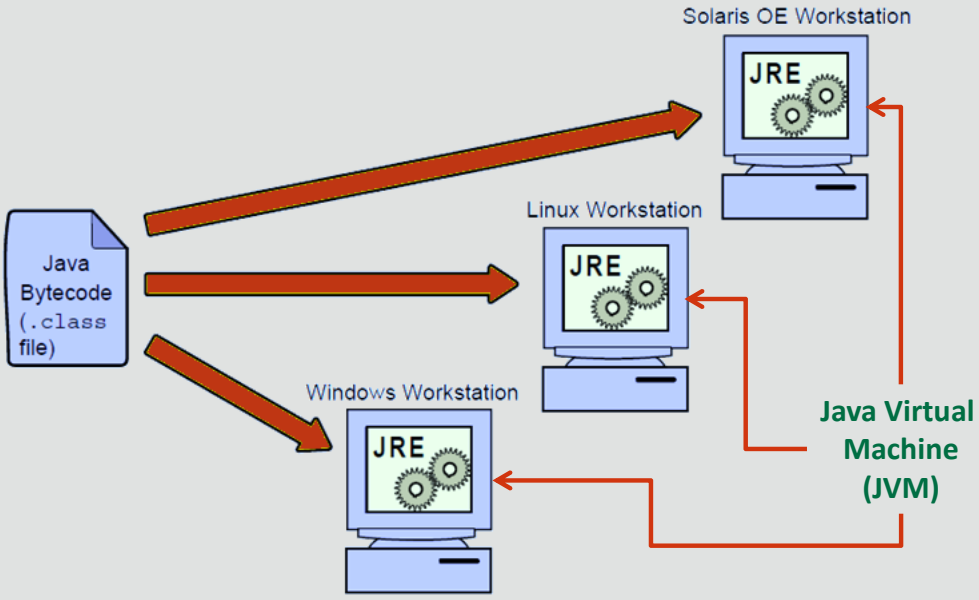
JFo 1-3
Setting up Java

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

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)

- Includes:
  - The Java Virtual Machine (JVM)
  - Java class libraries

JRE

- Purpose:
  - Read bytecode (.class)
  - Run the same bytecode anywhere with a JVM

Java Development Kit (JDK)

- Includes:
  - JRE
  - Java Compiler
  - Additional tools

JRE

JDK

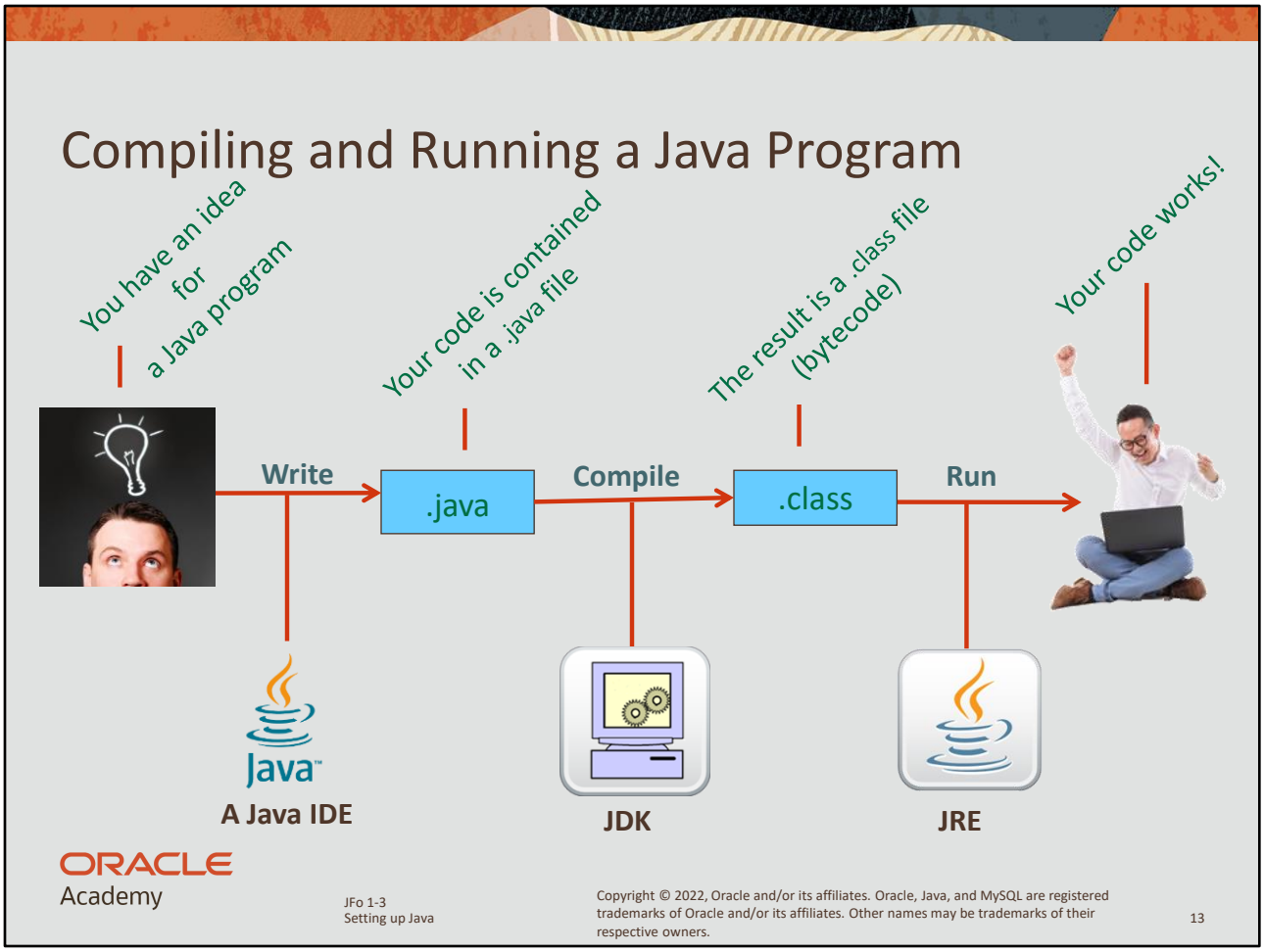
- Purpose:
  - Compile bytecode (.java →.class)

Integrated Development Environment (IDE)

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







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 a Java IDE, 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.

Working with existing code files

- Throughout this course, sample code and project starter code are supplied as .java files
- Complete the practice for this lesson as it demonstrates how to add existing .java files to a project in commonly used Java IDEs. (If you are using a different Java IDE refer to the IDE's documentation for instructions on how to do this)

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

A Java IDE 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 integrated development environment (IDE)
  - Add an existing .java file into a Java project

ORACLE

Academy