JAVA: Lesson 1

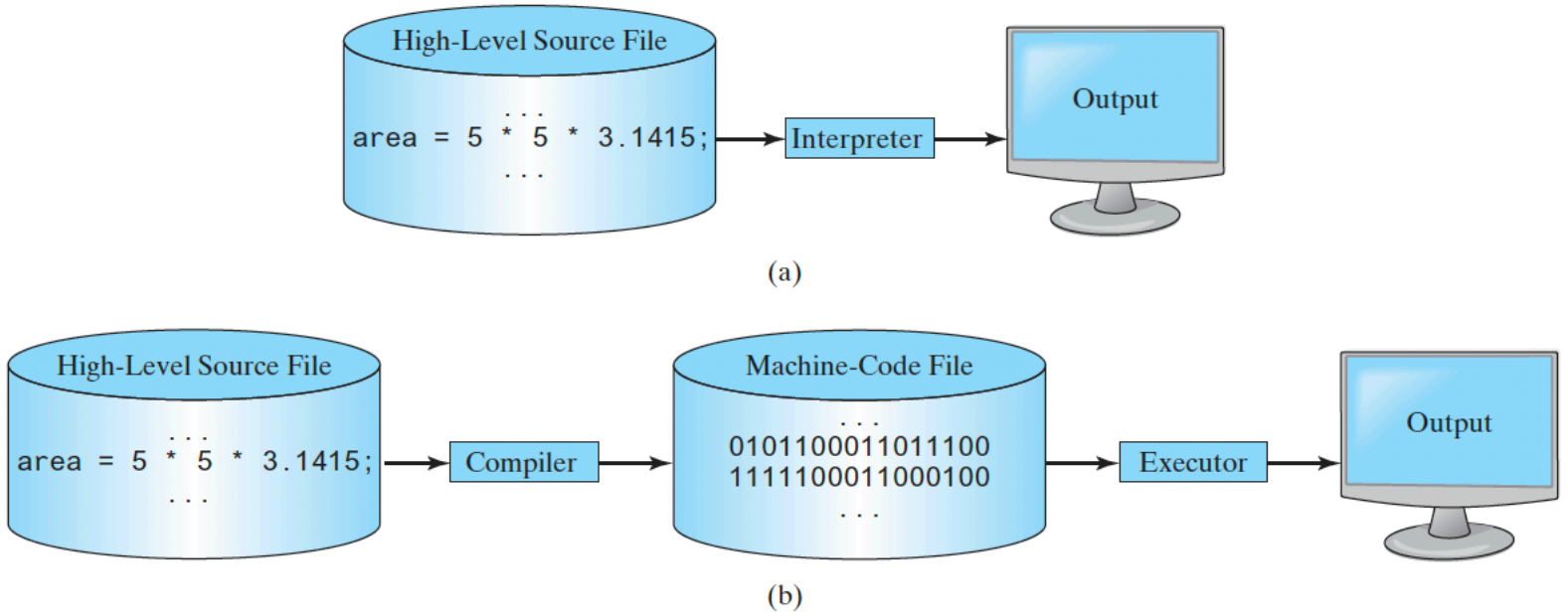
Avanindra Kumar Pandeya

Useful Commands

Command	Meaning
ls	List Directories/Files
cd directoryName	Change to DirectoryName
cd ..	Change to previous Directory
cp file1 ../file2	Copy file1 from current directory as file2 in previous directory
mv ../../file1 file2	Move File from 2 level previous directory to current Directory
pwd	Show Present Working Directory
clear	Clear the console
rm file1	Deletes File1
rmdir dirname	Removes Directory if it empty
rm –r dirname	Delete a non-empty directory with all the files in it. If a directory or a file within the directory is write-protected, you will be prompted to confirm the deletion.
rm -rf dirname	Remove non-empty directories and all the files without being prompted, use rm with the -r (recursive) and -f options
mkdir dirname	Make a directory at the current location

Types of Languages

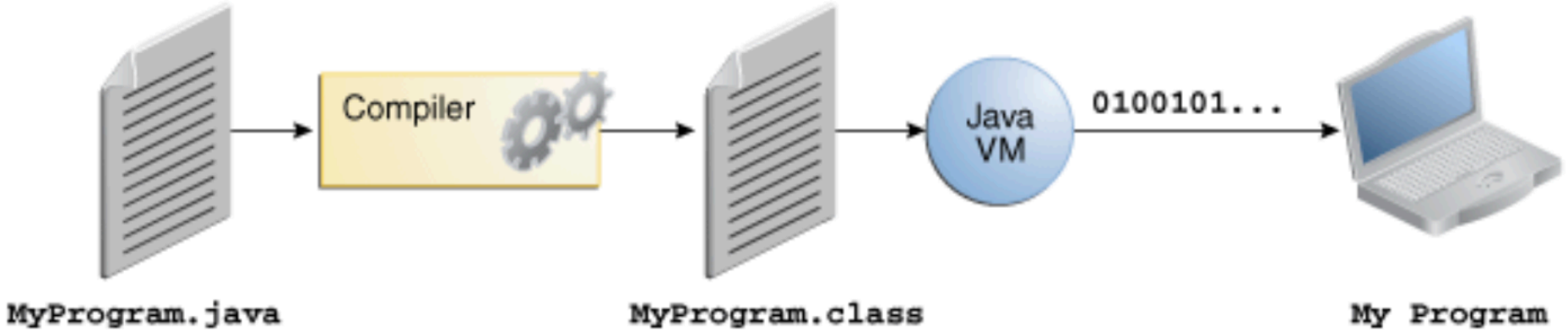
- Machine Language
- Assembly Language
- High Level Language



Machine Code is CPU dependent!

Fig Courtesy: Textbook

Software Development in Java

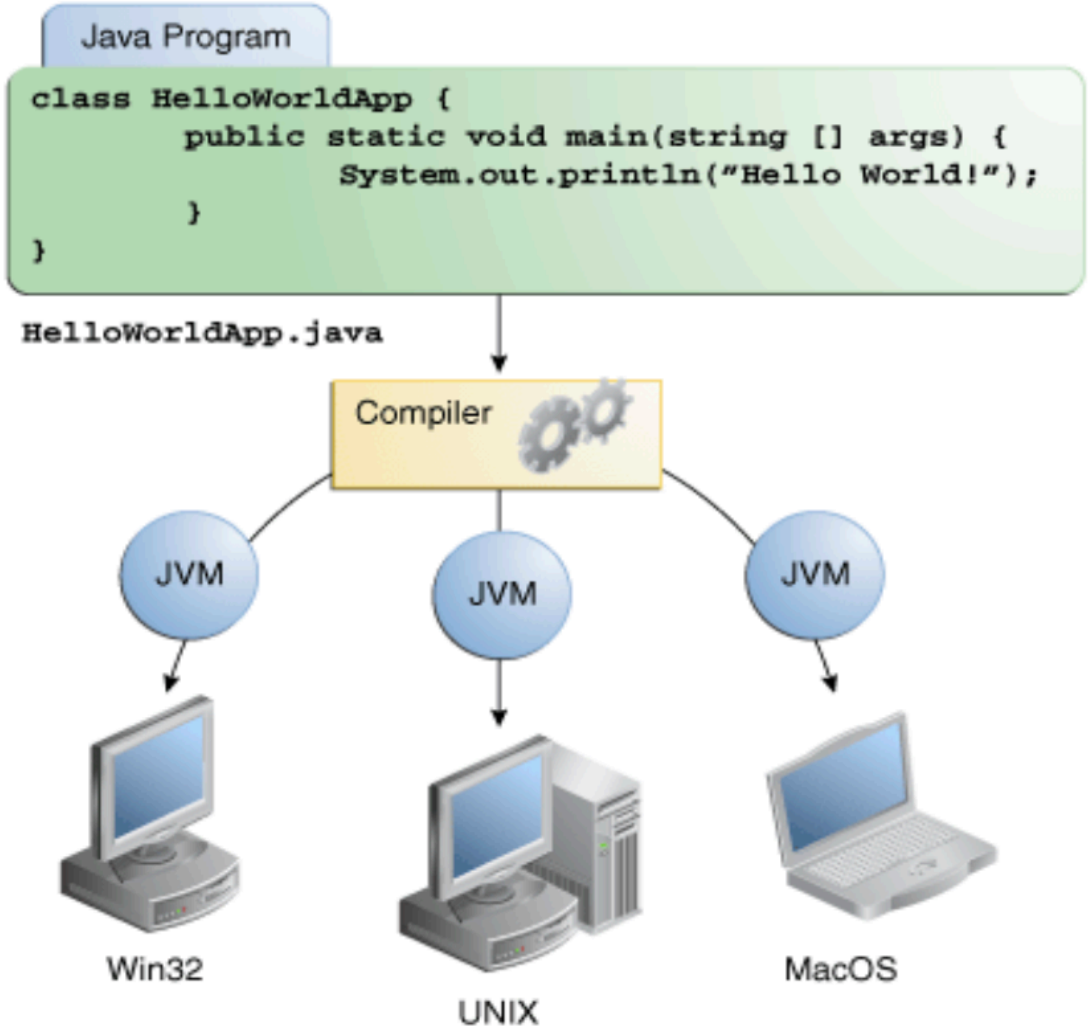


An overview of the software development process.

Fig Courtesy: Java Website

Architecture neutral and Portable

Fig Courtesy: Java Website



Through the Java VM, the same application is capable of running on multiple platforms.

Demonstration: Hello World


```java
/**
 * The HelloWorldApp class implements an application that
 * simply prints "Hello World!" to standard output.
 */


public class HelloWorldApp
{
    Run | Debug
    public static void main(final String[] args)  // Order of public and static does not matter

    {
        System.out.println("Hello World!"); // Display the string.
    }
}
```

Function Block

Class Block

Java Development Kit

- java - launch a Java application
- javac - read Java class and interface definitions and compile them into bytecode and class files
- javadoc - generate HTML pages of API documentation from Java source files
- Many More

Comments

- Comments are ignored by the compiler but are useful to other programmers. The Java programming language supports three kinds of comments:
- */* *text* */*

The compiler ignores everything from /* to */.

- */** *documentation* */*

This indicates a documentation comment (*doc comment*, for short). The compiler ignores this kind of comment, just like it ignores comments that use /* and */. The javadoc tool uses doc comments when preparing automatically generated documentation. For more information on javadoc, see the Javadoc™ tool documentation .

- *// text*

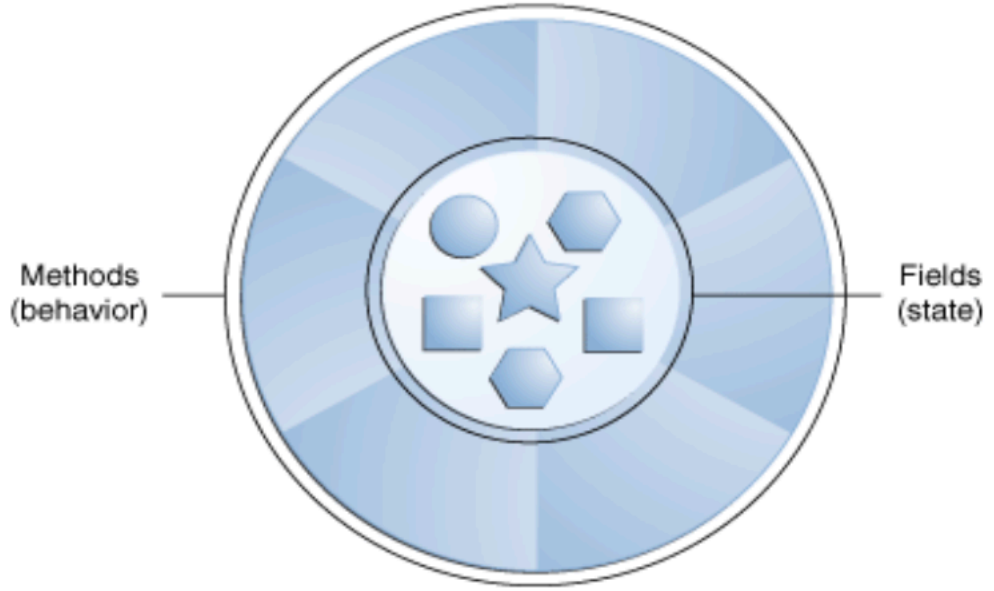
The compiler ignores everything from // to the end of the line.

Java is an object oriented programming language. But… What is an object?

What is an Object

- Real World objects have *state* and *behavior*.

Object	State	Behavior
Dogs	Name, Color, Breed, Hungry	Barking, Fetching, Wagging Tail
Bicycles	Current Gear, Current Cadence, Current Speed	Change Gear, Change the cadence, Apply Brakes



A software object.

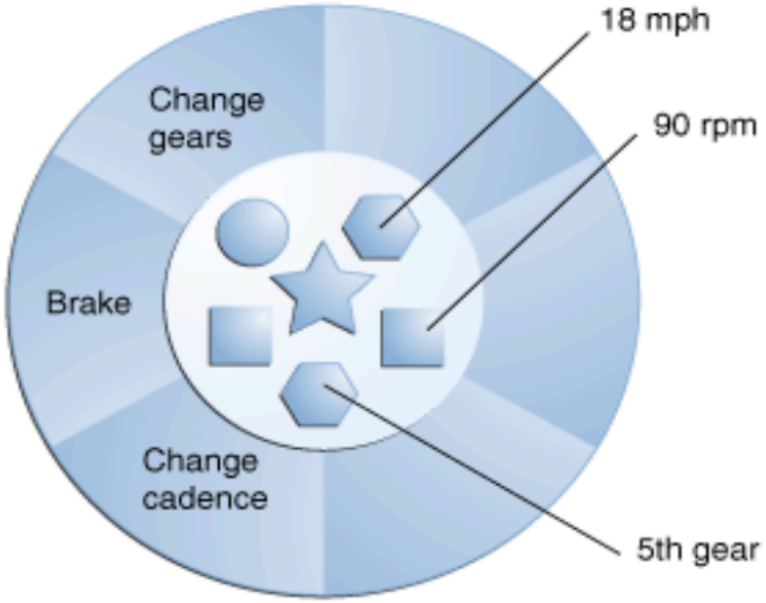
Fig Courtesy: Java Website

Demonstration

- Bicycle class demonstration

Data Encapsulation

- Hiding internal state and requiring all interaction to be performed through an object's methods is known as *data encapsulation* — a fundamental principle of object-oriented programming



A bicycle modeled as a software object.

Fig Courtesy: Java Website

Benefits of OOP

- Bundling code into individual software objects provides a number of benefits, including:
- **Modularity**: The source code for an object can be written and maintained independently of the source code for other objects. Once created, an object can be easily passed around inside the system.
- **Information-hiding**: By interacting only with an object's methods, the details of its internal implementation remain hidden from the outside world.
- **Code re-use**: If an object already exists (perhaps written by another software developer), you can use that object in your program. This allows specialists to implement/test/debug complex, task-specific objects, which you can then trust to run in your own code.
- **Pluggability and debugging ease**: If a particular object turns out to be problematic, you can simply remove it from your application and plug in a different object as its replacement. This is analogous to fixing mechanical problems in the real world. If a bolt breaks, you replace *it*, not the entire machine.