engineering.vips.edu

VIPS Technical Campus
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION

SCHOOL OF ENGINEERING AND TECHNOLOGY

Course : OBJECT ORIENTED PROGRAMMING

Paper Code: AIML-202,IIOT-202

Faculty : Dr. Shivanka

Assistant Professor

VIPS

SCHOOL OF ENGINEERING AND TECHNOLOGY

- Introduction to Object Oriented Programming
- Benefits of Object Oriented Programming
- Java Features
- How Java Differs from other OO languages
- Java Environment
- Execution Process organization of the Java virtual Machine Model of Java
- Build your first Java Program
- General Structure of Java program
- Classes and objects
- Inheritance
- Data Encapsulation and Abstraction:

Introduction to Object Oriented Programming

Object-oriented programming (OOP) is a programming paradigm using "objects" – data structures consisting of data fields and methods together with their interactions – to design applications and computer programs. Programming techniques may include features such as data abstraction, encapsulation, messaging, modularity, polymorphism, and inheritance. Many modern programming languages now support OOP.

Benefits of Object Oriented Programming

- Ability to reflect the real world
- Easy to Maintain the code
- Information Hiding
- Reusability of Code
- Flexibility of Code
- Security
- Troubleshooting is easy

Ability to reflect the real world

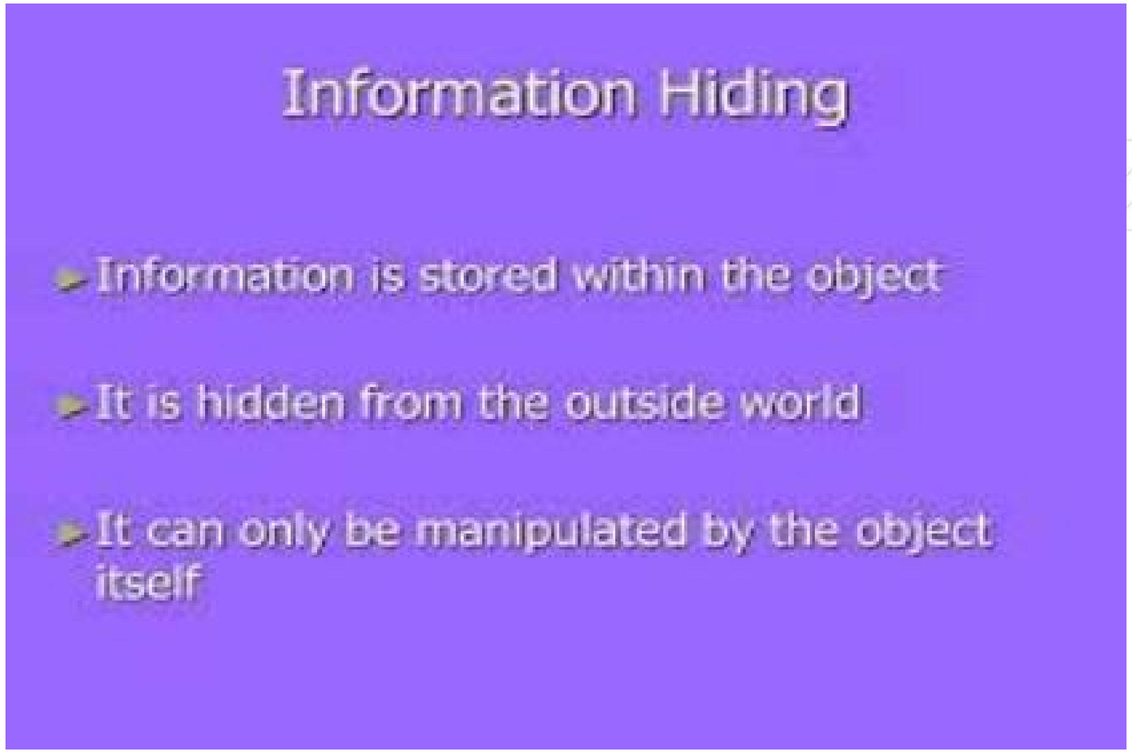
- ❖ Every Entity in the real world can be expressed as an object in OOP.
- ❖ An Object is an entity with has state and behavior.
- ❖ Eg. Dog is a real world entity
- It has state: breed, color etc.,
- behavior: barks, eats etc.,
- It can be expressed as Object, by defining state and behavior in the class.

Easy to Maintain the code

- Object Oriented Programming relies on classes, creation and communication among objects.
- It is easy to maintain the code.
- Part of the code can be modified without affect the entire program.
- Eg. Overriding, inheritance etc.,

Information Hiding

- Variables,Methods are bundled in class and accessed byobjects.
- Variables of a class are used by methods of that class only.
- Data Abstraction provide the concept of hiding implementation details and workingonly with the requiredmethods.Eg. Though users do not know the internal mechanism ofTV-Remote,they operate them by pressing buttons of the remote.



Information Hiding

- Information is stored within the object
- It is hidden from the outside world
- It can only be manipulated by the object itself

Reusability of Code

- The existing classes properties can be acquired by subclasses through the concept of inheritance.
- Reusability of Code adds essence to the Object Oriented Paradigm.
- Several types of Inheritances like single level, multilevel , hierarchical etc., help in reusability of code

Flexibility of the code

It is easy to modify the a part of the code without affecting the entire program. It is easy to reuse the existing code. It is easy to extend the code for performing new operations.

Security

- Object Oriented Programming provides security.
- The concept of Encapsulation bundles variables and methods into a component.
- The code can be protected from further modification, enhancement if required.

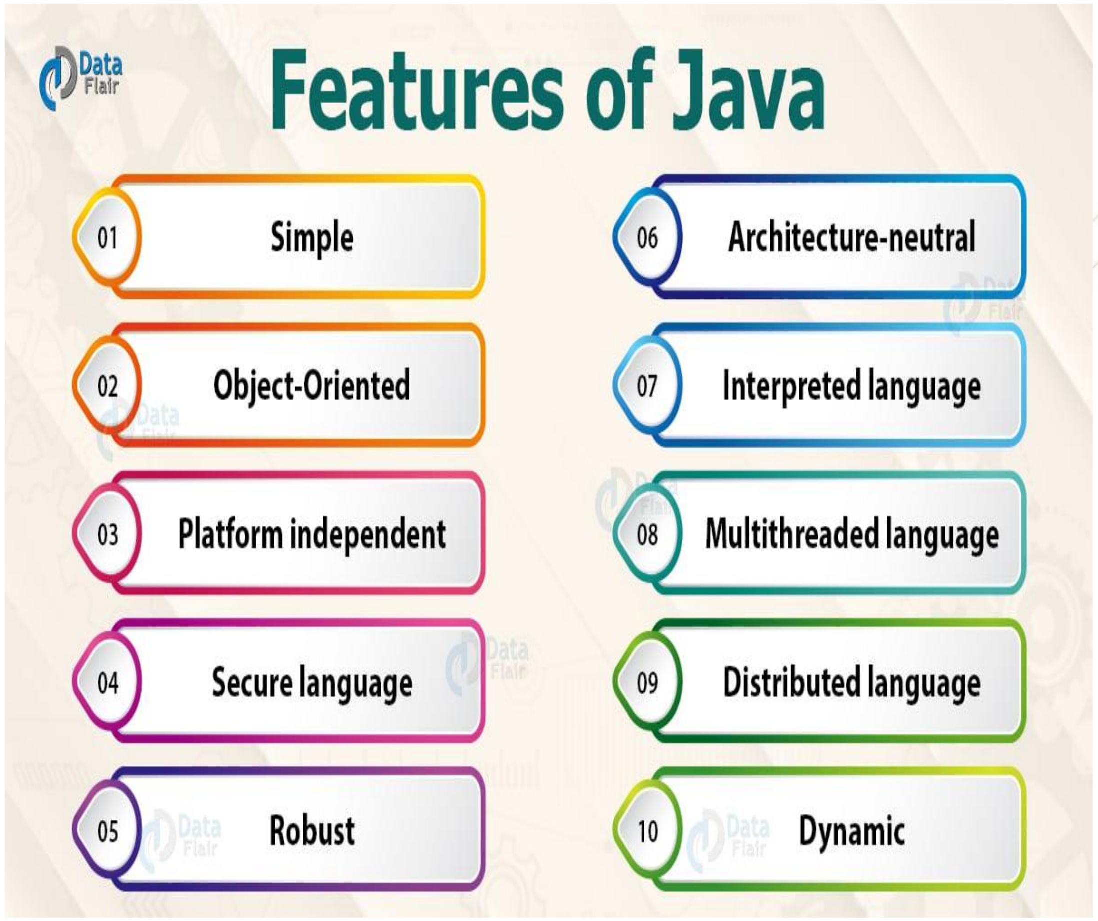
For example: final keyword is used for not letting the class to be used for inheritance.

Troubleshooting is easy

- Object Oriented Programming follows modular division and bottom up programming approach.
- Hence, makes troubleshooting easy.

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic



Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Java is partially modeled on C++, but greatly simplified and improved. Some people refer to Java as "C++--++" because it is like C++ but with more functionality and fewer negative aspects.

- fixes some clumsy features of C++
- no pointers
- automatic garbage collection
- rich pre-defined class library

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Java is inherently object-oriented. Object-oriented programming (OOP) is a popular programming approach that is replacing traditional procedural programming techniques.

One of the central issues in software development is how to reuse code. Object-oriented programming provides great flexibility, modularity, clarity, and reusability through encapsulation, inheritance, and polymorphism.

- focus on the data (objects) and methods manipulating the data
- all functions are associated with objects
- almost all datatypes are objects (files, strings, etc.)
- potentially better code organization and reuse

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Distributed computing involves several computers working together on a network. Java is designed to make distributed computing easy. Since networking capability is inherently integrated into Java, writing network programs is like sending and receiving data to and from a file.

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

You need an interpreter to run Java programs. The programs are compiled into the Java Virtual Machine code called bytecode. The bytecode is machine-independent and can run on any machine that has a Java interpreter, which is part of the Java Virtual Machine (JVM).

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Java compilers can detect many problems that would first show up at execution time in other languages. It has extensive compile-time and runtime error checking

- It has eliminated certain types of error-prone programming constructs found in other languages. There are no pointers. Memory corruptions or unauthorized memory accesses are impossible
- Java has a runtime exception-handling feature to provide programming support for robustness.
- It has automatic garbage collection tracks objects usage over time

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Java implements several security mechanisms to protect your system against harm caused by stray programs.

- usage in networked environments requires more security
- memory allocation model is a major defense
- access restrictions are forced (private, public)

Characteristics of Java

- Java Is Simple
- Java Is Object-Oriented
- Java Is Distributed
- Java Is Interpreted
- Java Is Robust
- Java Is Secure
- Java Is Architecture-Neutral
- Java Is Portable
- Java's Performance
- Java Is Multithreaded
- Java Is Dynamic

Write once, run anywhere

With a Java Virtual Machine (JVM), you can write one program that will run on any platform.

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Because Java is architecture neutral, Java programs are portable. They can be run on any platform without being recompiled.

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Java's performance is very high because of its architecture neutral and portable features.

Characteristics of Java

- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Multithread programming is smoothly integrated in Java, whereas in other languages you have to call procedures specific to the operating system to enable multithreading.

Characteristics of Java

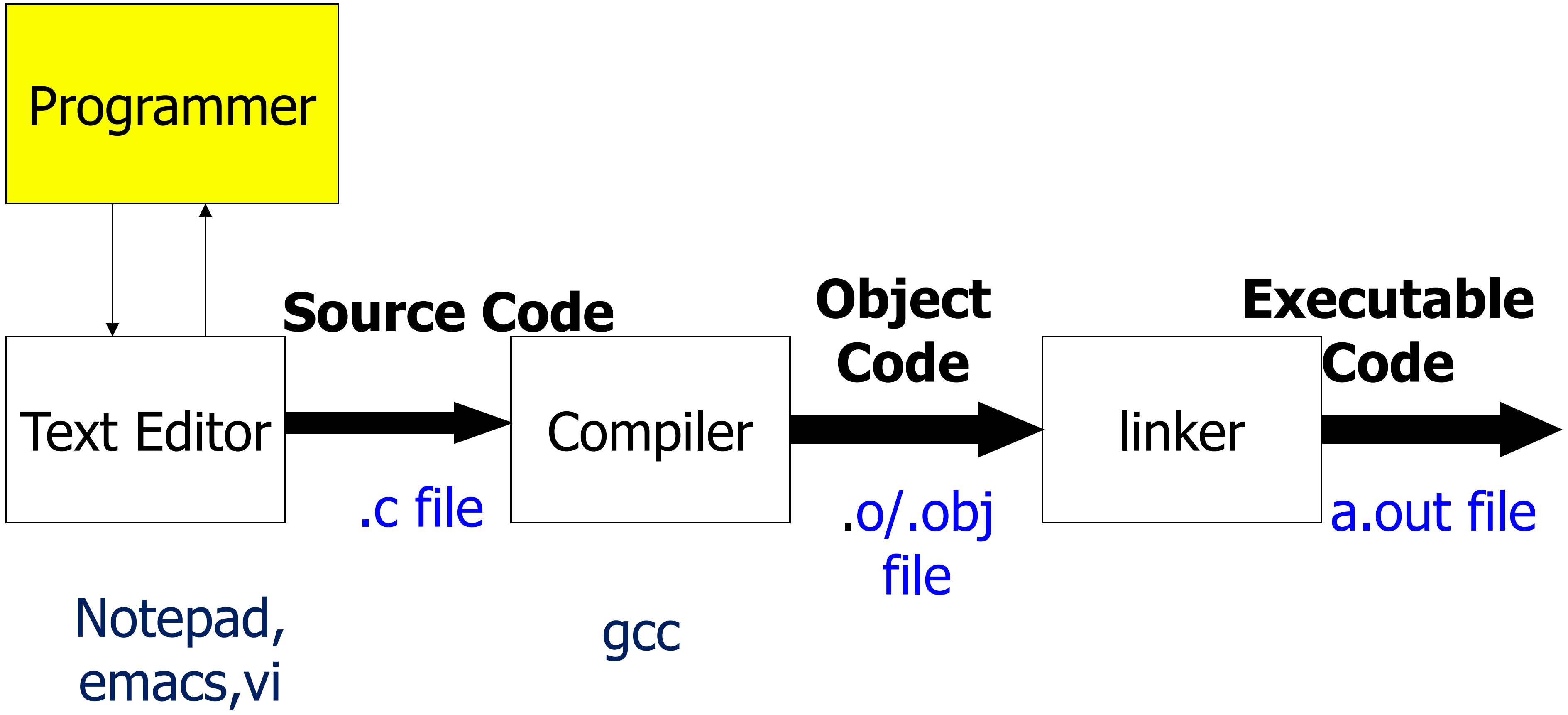
- ❖Java Is Simple
- ❖Java Is Object-Oriented
- ❖Java Is Distributed
- ❖Java Is Interpreted
- ❖Java Is Robust
- ❖Java Is Secure
- ❖Java Is Architecture-Neutral
- ❖Java Is Portable
- ❖Java's Performance
- ❖Java Is Multithreaded
- ❖Java Is Dynamic

Java was designed to adapt to an evolving environment. New code can be loaded on the fly without recompilation. There is no need for developers to create, and for users to install, major new software versions. New features can be incorporated transparently as needed.

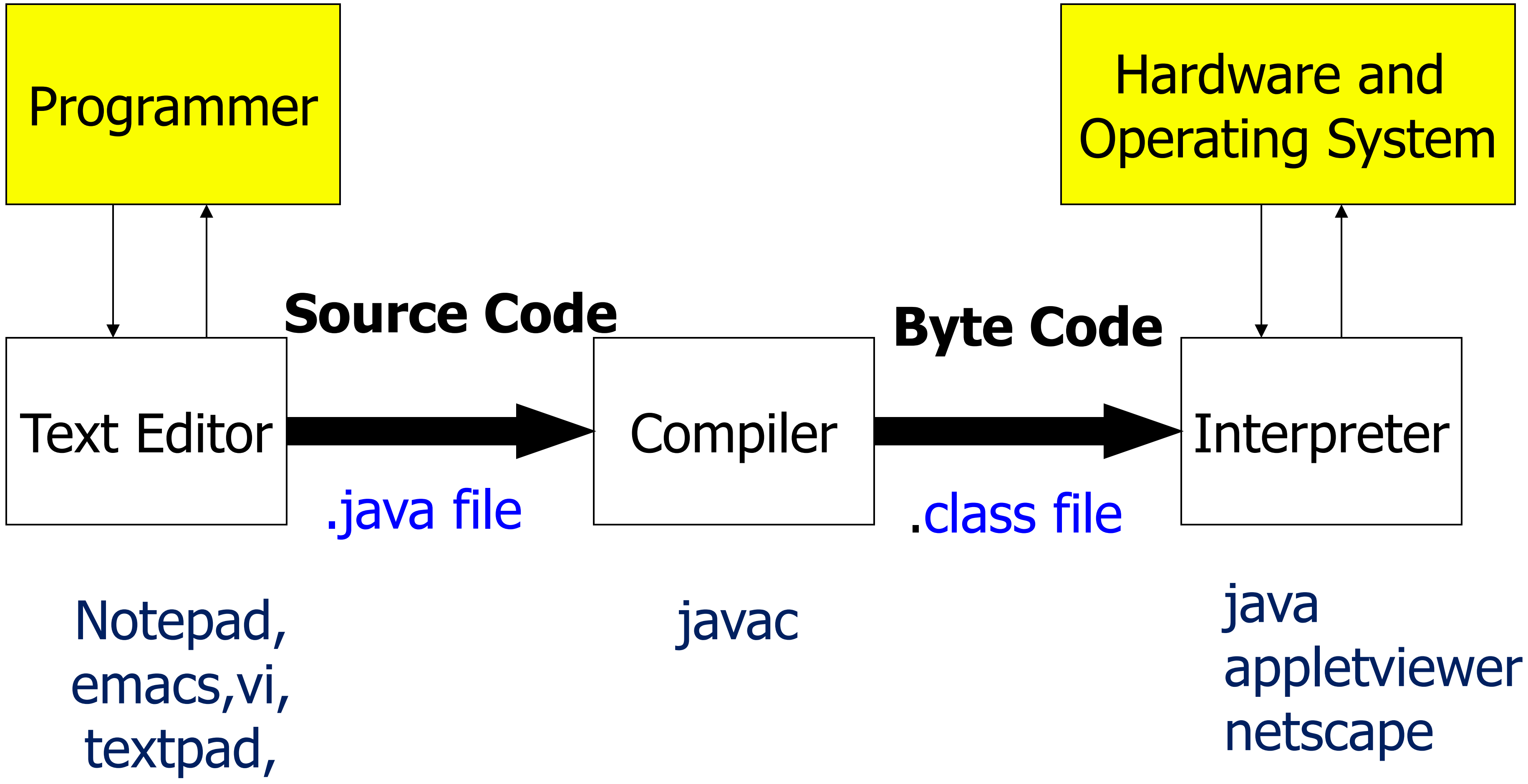
Java Disadvantages

- **Slower than compiled language such as C**
  - ✓ an experiment in 1999 showed that Java was 3 or 4 times slower than C or C++
  - ✓ *title of the article: "Comparing Java vs. C/C++ Efficiency Issues to Interpersonal Issues" (Lutz Prechelt)*
- ✓ Running bytecode through the interpreter is not as fast as running machine code, which is specific to that platform.
- **Because it is platform independent, it is difficult to use platform specific features (e.g., Windows taskbar, quick launch) in Java.**
- **Java interpreter must be installed on the computer in order to run Java programs.**

Compiled Languages

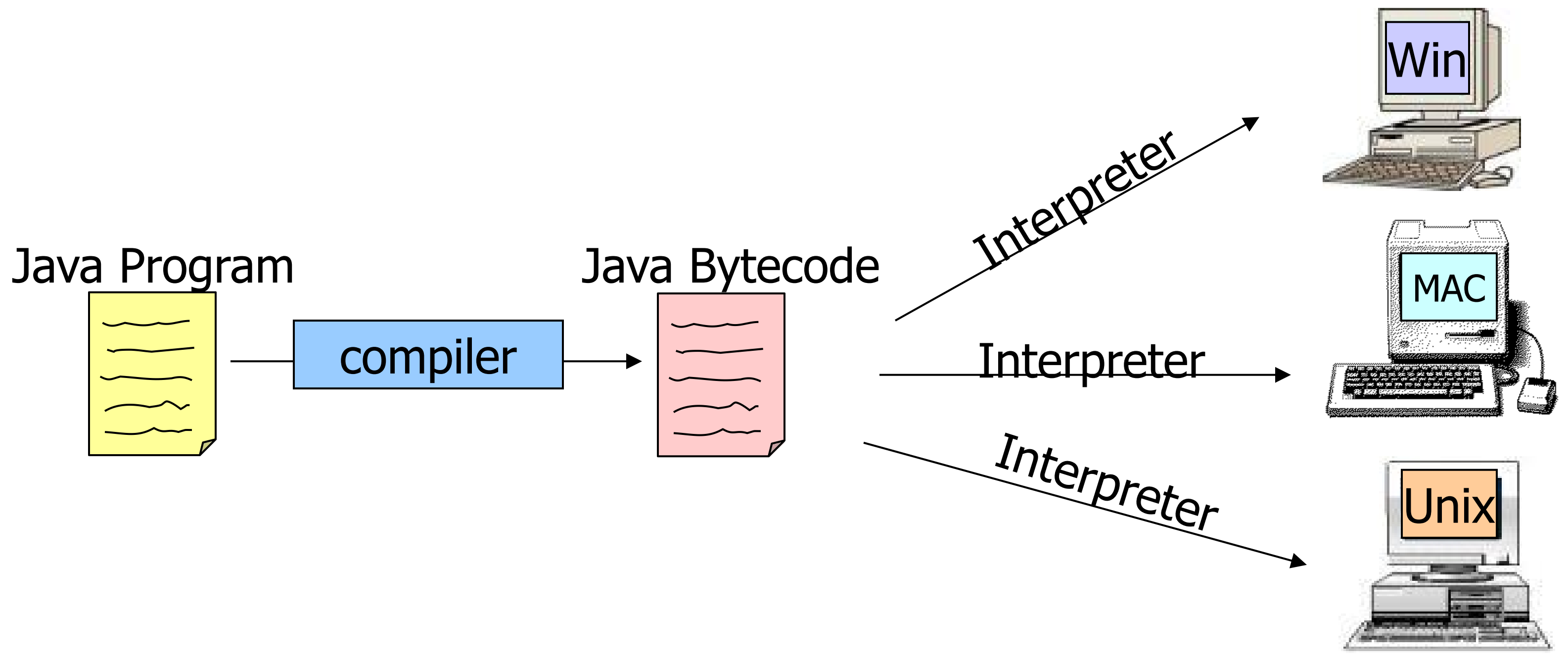


Java is Compiled and Interpreted



Java Interpreter

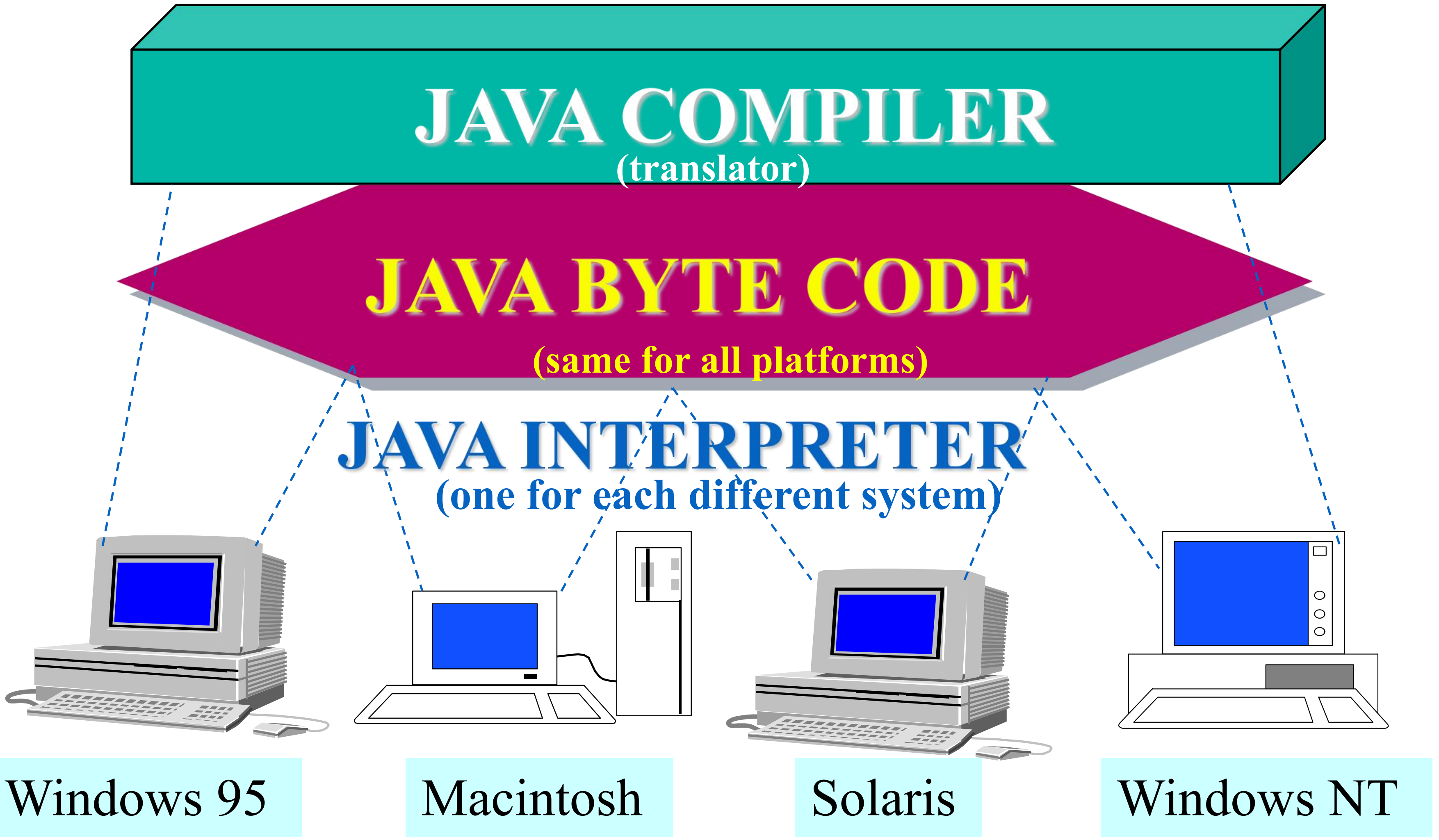
- **Java** is a little different.
- Java compiler produces **bytecode** not machine code.
- Bytecode can be run on any computer with the **Java interpreter** installed.



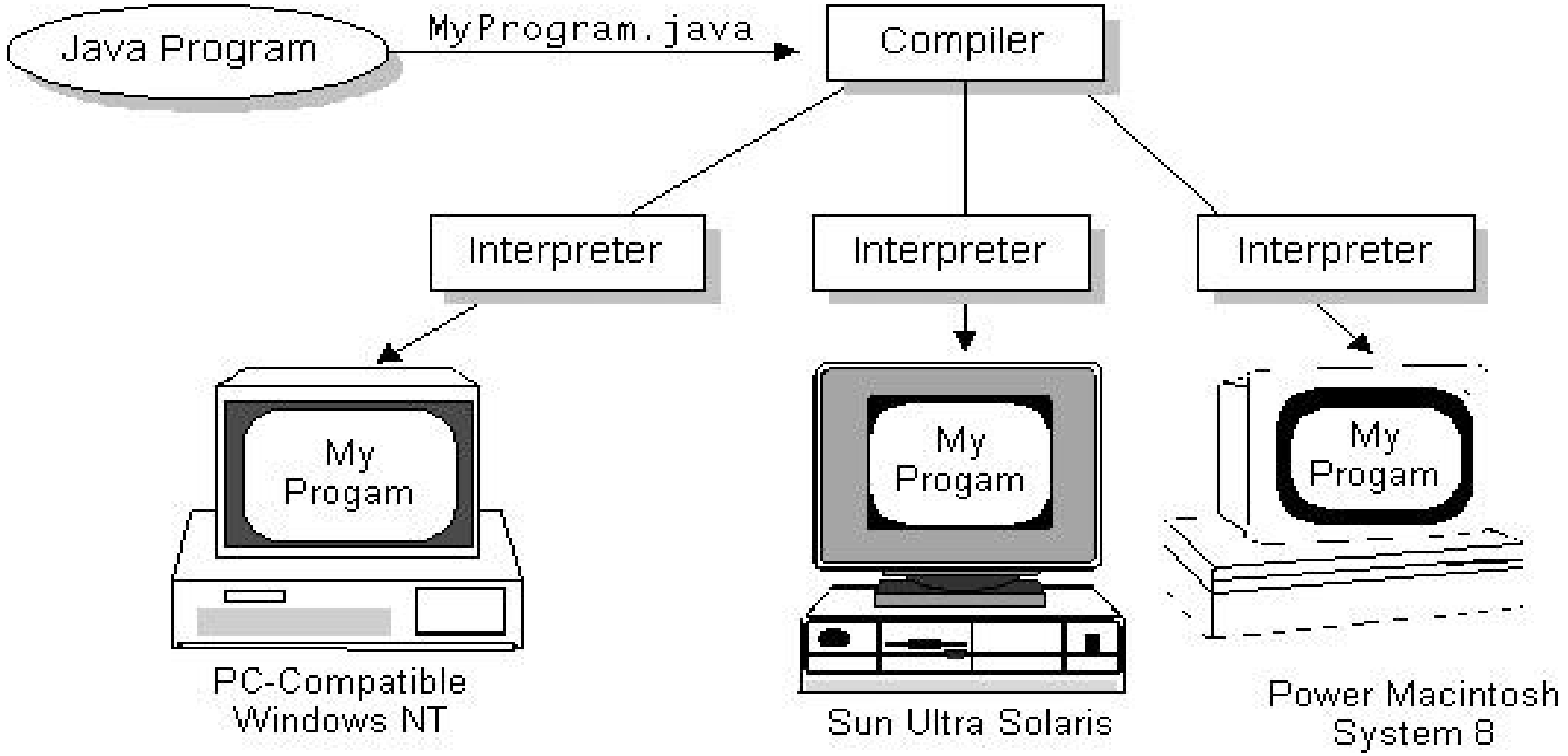
ByteCode: Food for the VM

- ➢For most languages, compilation produces machine code
- ➢Java compilation produces "bytecode"
  - ✓Intermediate code readable by the VM
  - ✓Transferable across the Internet as *applets*
- ➢VM interprets BC into instructions
  - ✓Partly responsible for performance lag
- ➢ByteCode produced on any platform may be executed on any other platform which supports a VM

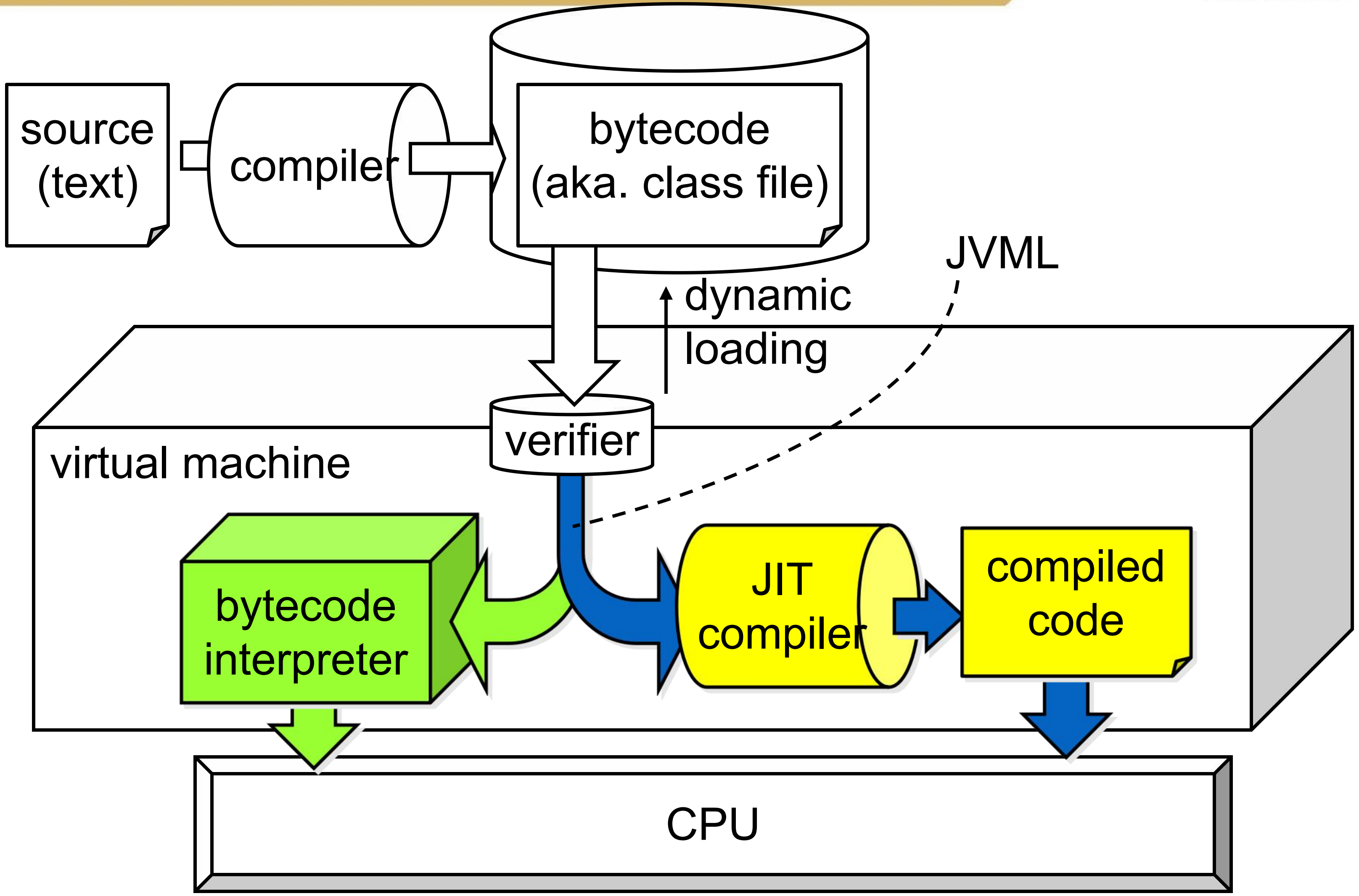
Total Platform Independence



Write Once, Run Anywhere



Execution Model of Java



The JIT

- *Just-In-Time* compiler
- Translates bytecode into machine code at runtime
  - ✓ 1-time overhead when run initiated
  - ✓ Performance increase 10-30 times
- Now the default for most JVM's
  - ✓ Can be turned off if desired
  - ✓ JIT can apply statistical optimizations based on runtime usage profile

Not just one JVM, but a whole family

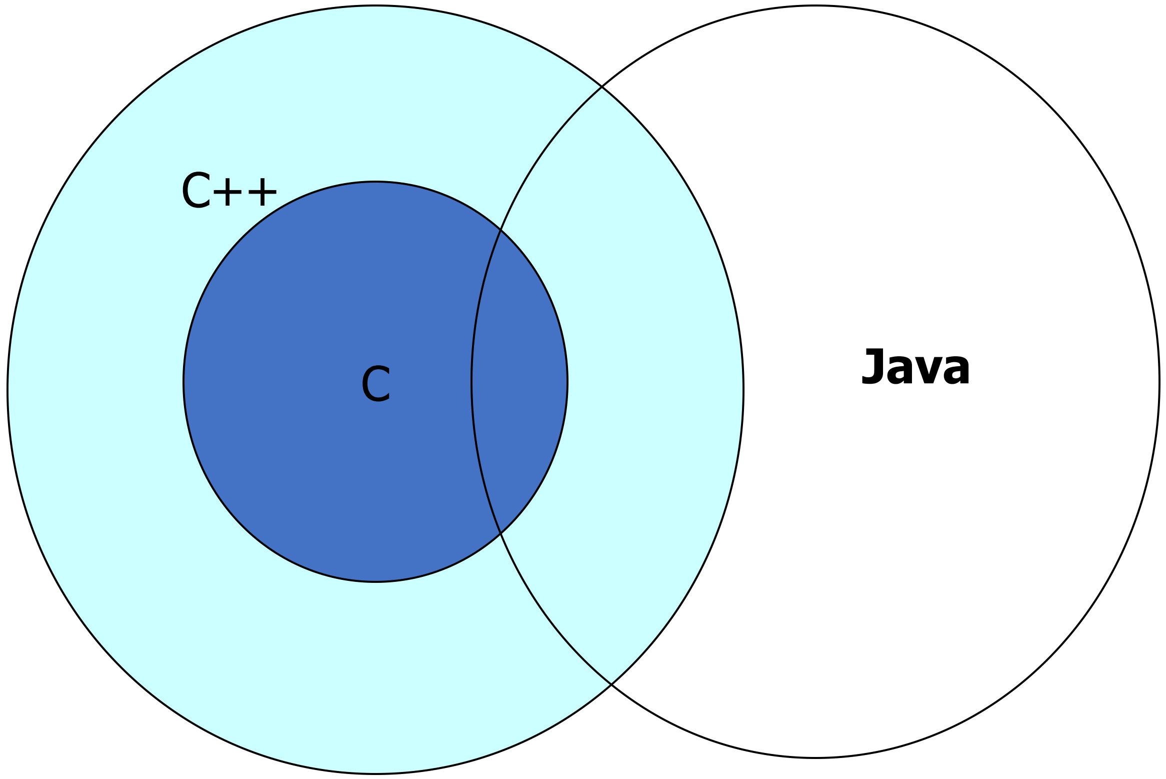
- JVM (J2EE & J2SE)
  - ✓ Well-known Java Virtual Machine.
- CVM , KVM (J2ME)
  - ✓ Small devices.
  - ✓ Reduces some VM features to fit resource-constrained devices.
- JCVM (Java Card)
  - ✓ Smart cards.
  - ✓ It has least VM features.

and there are also lots of other JVMs

How Does Java Compares to C++ and Other OO Languages

Overlap of C, C++, and Java

VIPS Technical Campus
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION
SCHOOL OF ENGINEERING AND TECHNOLOGY



Difference between Java, C and C++

In a first contact, Java seems like C++, and it's logical because Java takes the C and C++ syntax. But Java has some important differences with C++. For instance, you can't use pointers in Java, neither operators overload, neither multiple inheritance, neither predefined types. These features of C++ that Java doesn't has, make it a simplest and more robust language. Interpreted Java is furthermore slower than C++ (even 20 to 50 times slower than C in the original Java interpreters).

Memory administration: Java memory administration is automatic; memory is assigned automatically when you create an object, and also a garbage collector frees the memory when that object is not used.
Functions malloc() and free() don't exist in Java.

Type of data: In Java, Primitive data types (like char, int, long...) have sizes and behaviors which may be different in some platforms and operative systems. In Java language, unsigned data don't exist. The boolean data has two values in Java : true and false. So it isn't an integer type, but you can force "0" and "1" (which are integers) to be booleans.

Difference between Java, C and C++

Operators: The execution order of the operators in Java is same as in C.

Flux Control: The syntax of the following statements if, while, for and do is the same as in C and C++. But there is an important difference : the proof expression for each flux construction should return a boolean value (true or false). In C and C++, the expression can return an integer.

Arguments: In Java all method definitions have to have an specific number of arguments. The arguments in the command line have a different behavior than in C and C++. So, in these languages argv[0] is the name of the program but in Java this is the first of additional arguments.

Java better than C++ ?

- No Typedefs, Defines, or Preprocessor
- No Global Variables
- No Goto statements
- No Pointers
- No Unsafe Structures
- No Multiple Inheritance
- No Operator Overloading
- No Automatic Coercions
- No Fragile Data Types

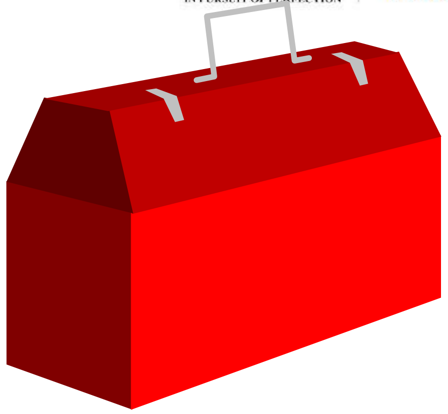
Object Oriented Languages -A Comparison

Feature	C++	Objective C	Ada	Java
Encapsulation	Yes	Yes	Yes	Yes
Inheritance	Yes	Yes	No	Yes
Multiple Inherit.	Yes	Yes	No	No
Polymorphism	Yes	Yes	Yes	Yes
Binding (Early or Late)	Both	Both	Early	Late
Concurrency	Poor	Poor	Difficult	Yes
Garbage Collection	No	Yes	No	Yes
Genericity	Yes	No	Yes	Limited
Class Libraries	Yes	Yes	Limited	Yes

Java Development Kit

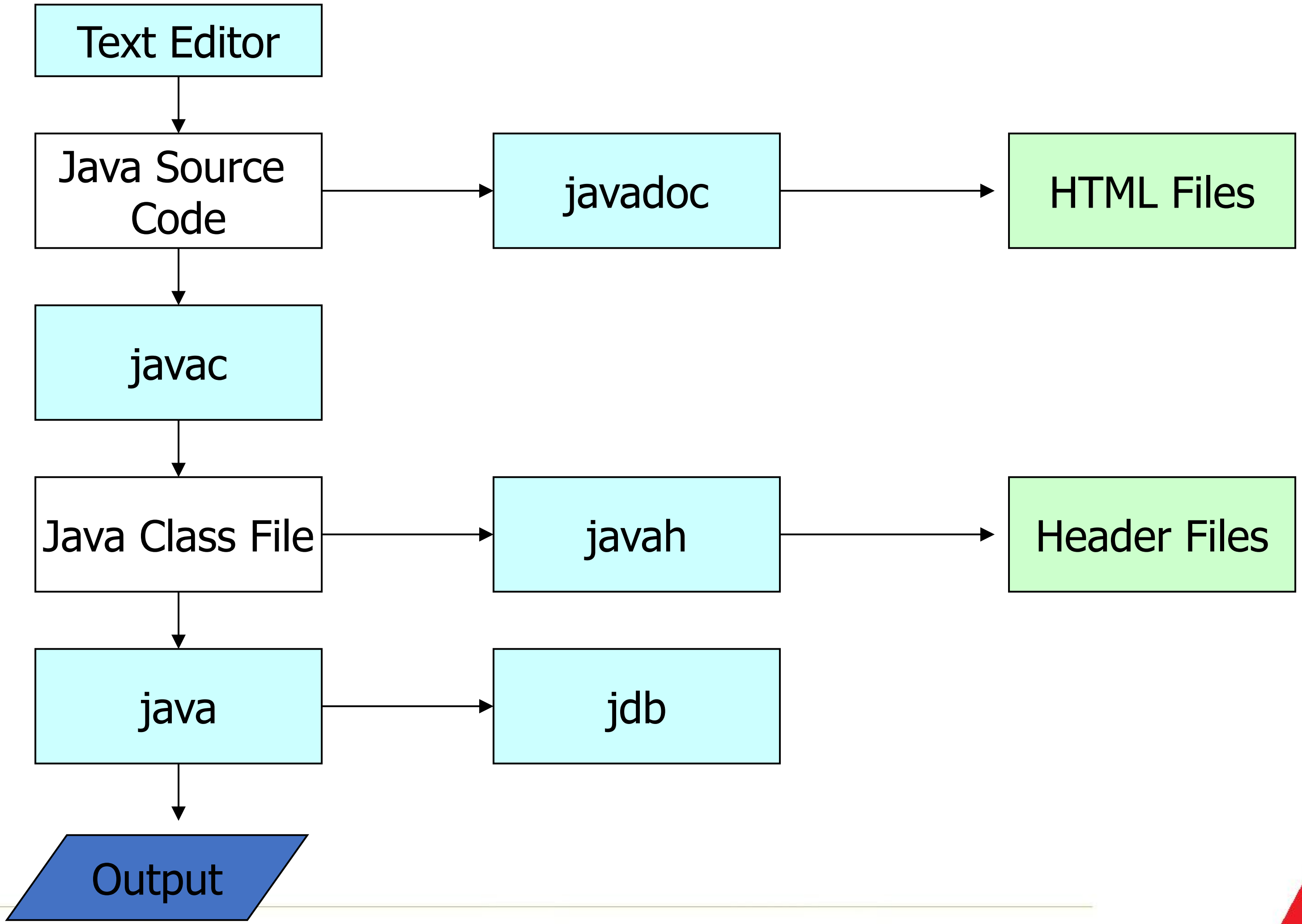
SCHOOL OF ENGINEERING AND TECHNOLOGY

- ✓javac - The Java Compiler
- ✓java - The Java Interpreter
- ✓jdb-   The Java Debugger
- ✓appletviewer -Tool to run the applets
- ✓javap - to print the Java bytecodes
- ✓javaprof - Java profiler
- ✓javadoc - documentation generator
- ✓javah - creates C header files



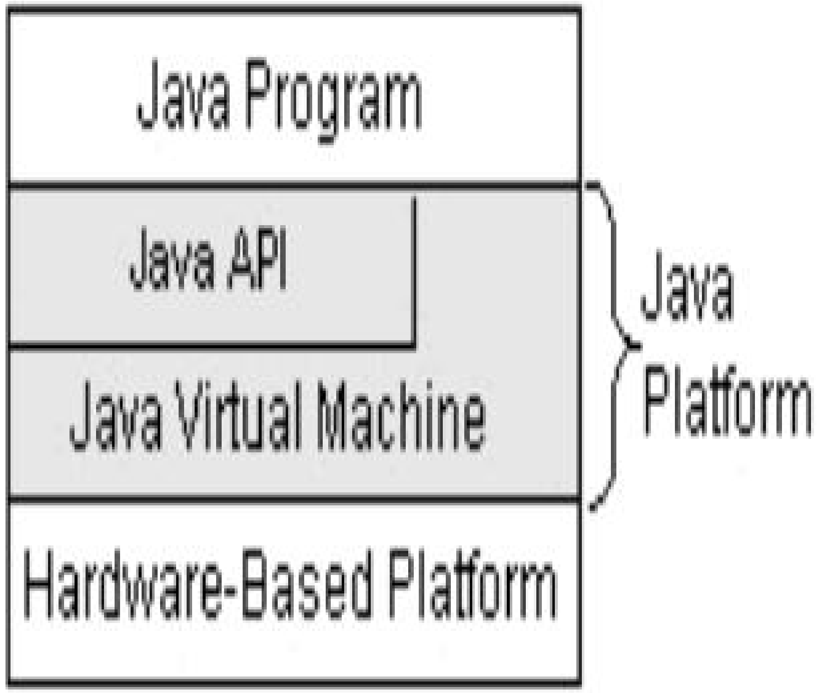
Process of Building and Running Java Programs

VIPS Technical Campus
योगः कर्मसु कौशलम्
IN PURSUIT OF PERFECTION
SCHOOL OF ENGINEERING AND TECHNOLOGY



Java VM and API

- Java API and Virtual Machine insulate the Java program from *hardware dependencies*.
- Java API



Java API

- Collection of ready-made software components that provide many useful capabilities.
- Grouped into libraries (*packages*) of related components.

- Core API
  - Essentials: Object, String, Input and Output...
  - Applets
  - Networking
  - Internationalization
  - Security
  - Software Components
  - Object Serialization
  - Java Database Connectivity (JDBC)

Building your first Java Program

The “Hello World” Application

Create a Java Source File

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

Compile and Run

- ➢Compile
  - ✓javac   HelloWorld.java
- ➢One file named HelloWorld.class is created if the compilation succeeds.
- ➢Run
  - ✓java HelloWorld

The Simplest Java Application: Hello,World!

- Since Java is object-oriented, programs are organized into modules called classes, which may have data in variables and subroutines called methods.

Each program is enclosed in a class definition.

main() is the first method that is run.

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

The notation class.method or package.class.method is how to refer to a public method (with some exceptions).

Syntax is similar to C - braces for blocks, semicolon after each statement. One difference: upper and lower case matter!

Understanding first java program

•**class** keyword is used to declare a class in java.

•**public** keyword is an access modifier which represents visibility, it means it is visible to all.

•**static** is a keyword, if we declare any method as static, it is known as static method. The core advantage of static method is that there is no need to create object to invoke the static method. The main method is executed by the JVM, so it doesn't require to create object to invoke the main method. So it saves memory.

•**void** is the return type of the method, it means it doesn't return any value.

•**main** represents startup of the program.

•**String[] args** is used for command line argument.

•**System.out.println()** is used print statement.

Classes:

In object-oriented programming, a class is a construct that is used as a blueprint to create instances of the class (class instances, class objects, instance objects or just objects). A class defines constituent members which enable class instances to have state and behavior. Data field members (member variables or instance variables) enable a class object to maintain state. Other kinds of members, especially methods, enable a class object's behavior. Class instances are of the type of the associated class.

For example, an instance of the class "Fruit" (a "Fruit" object) would be of the type "Fruit". A class usually represents a noun, such as a person, place or (possibly quite abstract) thing. Programming languages that include classes as a programming construct subtly differ in their support for various class-related features. Most support various forms of class inheritance. Many languages also support advanced encapsulation control features, such as access specifiers.

Object:

Object is an run time entity.

Is an Instance of class

Represents a Place ,Person ,anything that have some attributes.

Objects and Instances:

There is a very important distinction between an object and an instance of an object. An object is actually a definition, or a template for instances of that object. An instance of an object is an actual thing that can be manipulated.

For instance, we could define a Person object, which may include such member data as hair color, eye color, height, weight, etc. An instance of this object could be "Dave" and Dave has values for hair color, eye color, etc. This allows for multiple instances of an object to be created.

Data Encapsulation and Abstraction:

Data encapsulation, sometimes referred to as **data hiding**.

Data Encapsulation and Data Abstraction is one of the most striking feature of object oriented programming.

The wrapping up of data and code into a single unit is called data encapsulation. The data is not accessible to the outside world only those functions which are wrapped into a class can only access the private data of the class.

Contd...

Data Encapsulation and Abstraction:

- The concept of data encapsulation is supported in C++ through the use of the public, protected and private keywords which are placed in the declaration of the class.

- **Note :**

❖ Anything in the class placed after the public keyword is accessible to all the users of the class

❖ Elements placed after the protected keyword are accessible only to the methods of the class or classes derived from that class

❖ Elements placed after the private keyword are accessible only to the methods of the class.

Inheritance :

Inheritance is one of the most striking feature of object oriented programming.

Inheritance is the process by which one class can acquire the properties of another class.

The new classes, known as subclasses (or derived classes), inherit attributes and behavior of the pre-existing classes, which are referred to as superclasses (or ancestor classes). The inheritance relationships of classes gives rise to a hierarchy

Contd...

Inheritance :

A **superclass**, base class, or parent class is a class from which other classes are derived. The classes that are derived from a superclass are known as **child classes, derived classes, or subclasses.**

In object-oriented programming (OOP), inheritance is a way to compartmentalize and reuse code by creating collections of attributes and behaviors called objects which can be based on previously created objects.

Thank You