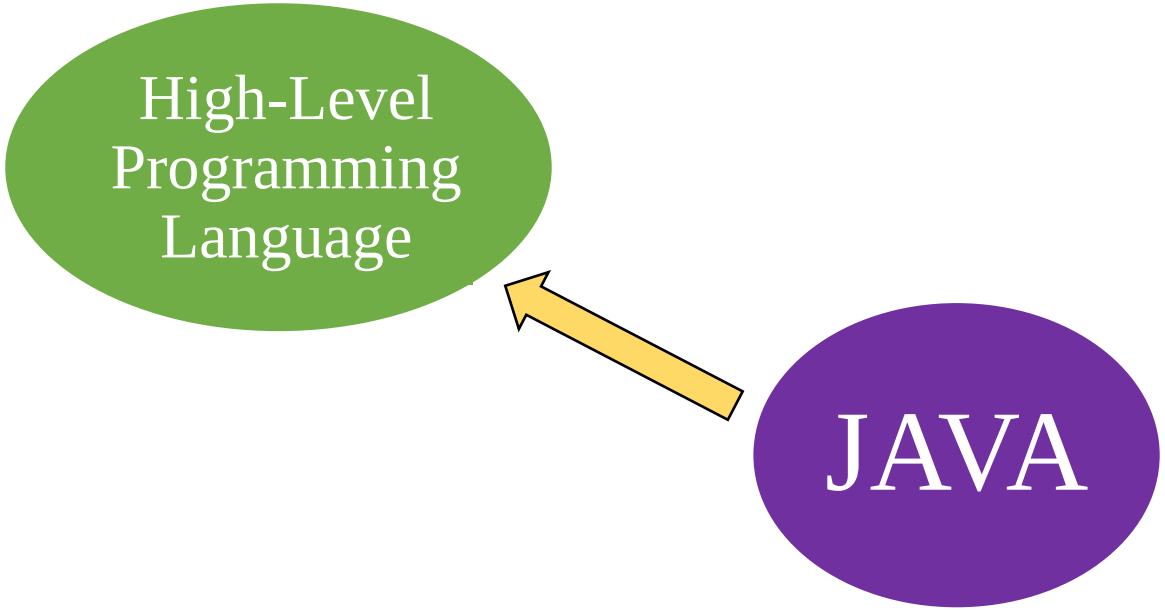
Sumeet Rathod

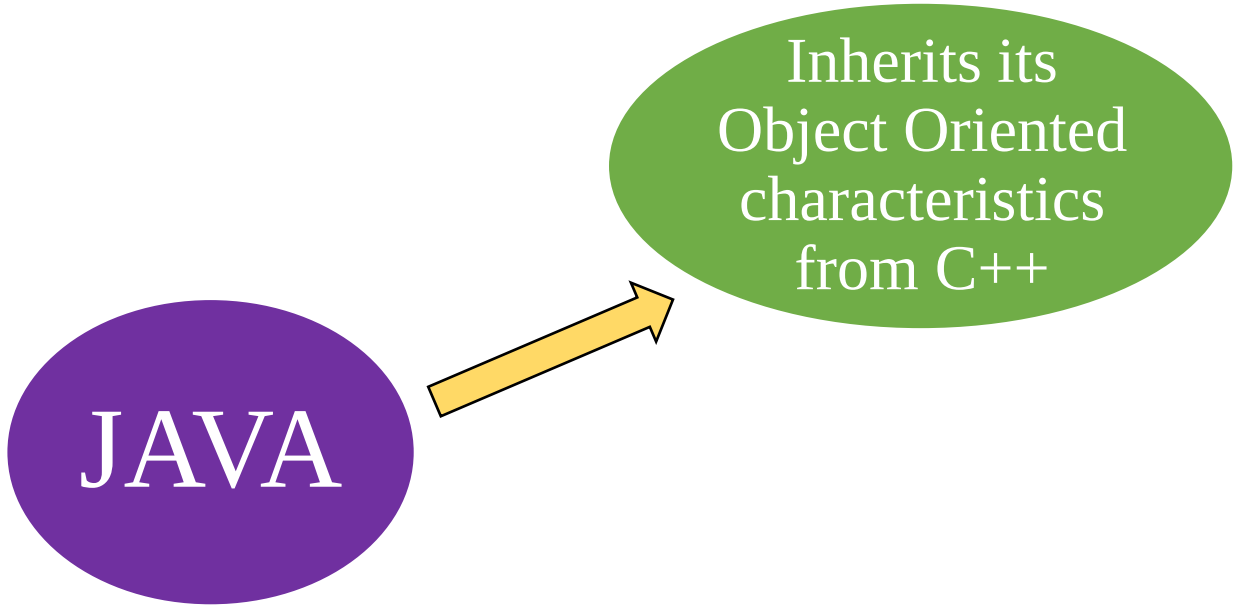
INTRODUCTION TO JAVA

Introduction to Java



High level languages are written in a form that is human-readable, making it programmer friendly and enabling it to more focus on the problem solving. Due to its portable feature, it makes the programs machine-independent.

Introduction to Java



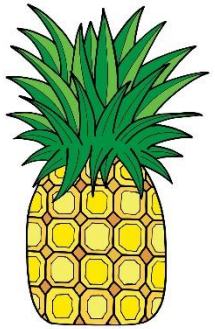
Fruits

Milk

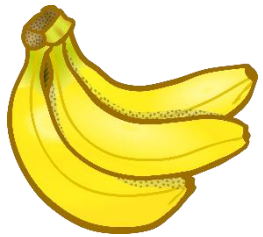
Water

Strawberry

Orange



Pineapple



Banana

Milk

+

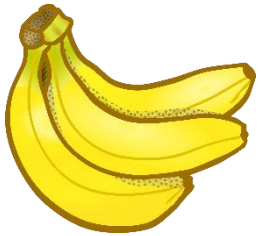
Strawberry

=

Strawberry Milkshake

Milk

+



Banana

=

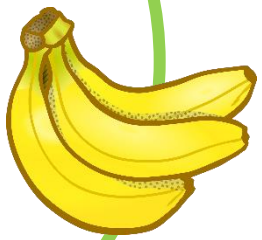
Banana Milkshake

Fruits

Water

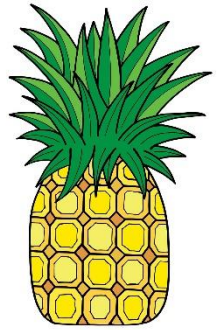
Milk

Strawberry



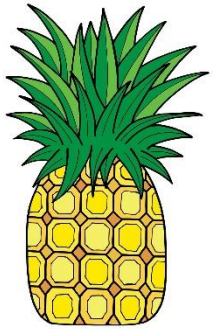
Banana

Orange



Pineapple

Water



Pineapple

Pineapple Juice

Water

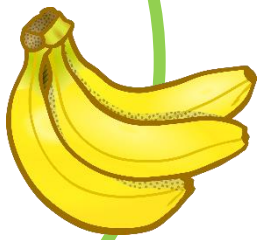
Orange

Orange Juice

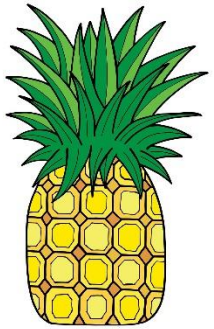
Fruits

Milk

Strawberry



Banana



Pineapple

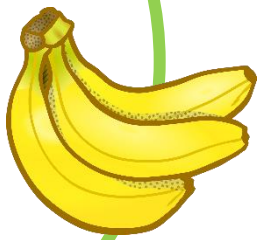
Water

Orange

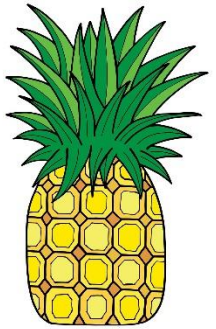
Fruits

Milk

Strawberry



Banana

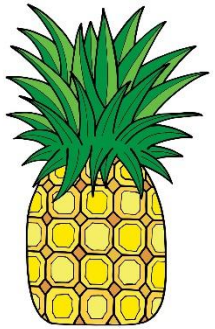


Pineapple

Milk

Orange

Milk



Pineapple

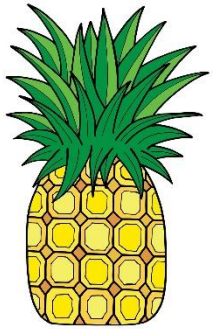
Pineapple Milkshake

Milk

Orange

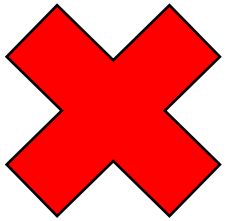
Orange Milkshake

Milk



Pineapple

PROPERTY

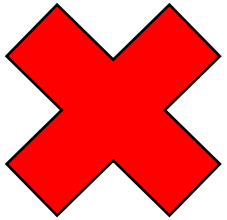


Pineapple Milkshake

Milk

Orange

PROPERTY

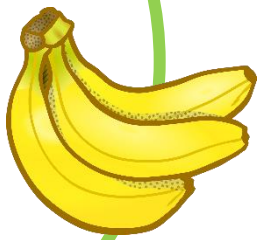


Orange Milkshake

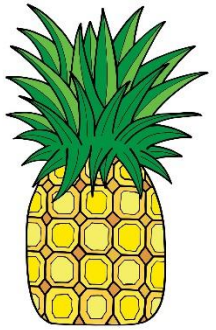
Fruits

Milk

Strawberry

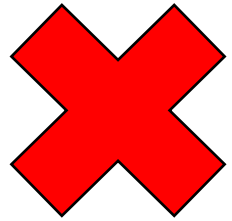


Banana



Pineapple

Milk

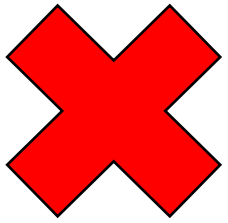


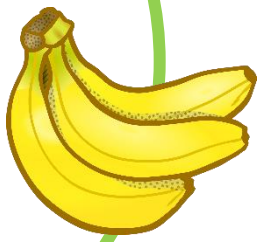
Orange

Fruits

Water

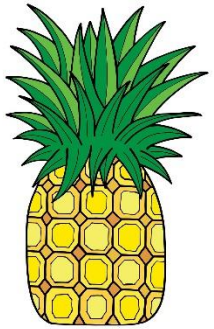
Strawberry



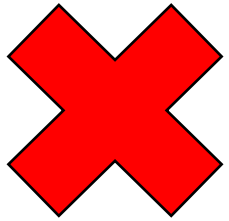


Banana

Milk



Pineapple



Orange

CLASS

Fruits

FUNCTION

Milk

FUNCTION

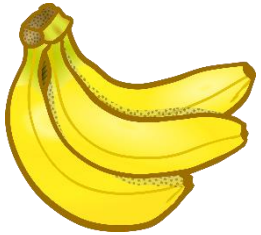
Water

PROPERTIES

Object Oriented

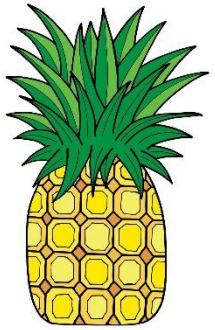
OBJECTS

Strawberry



Banana

OBJECTS



Pineapple

Orange

Introduction to Java

“class” is a template that narrates the behavior of the supported object.

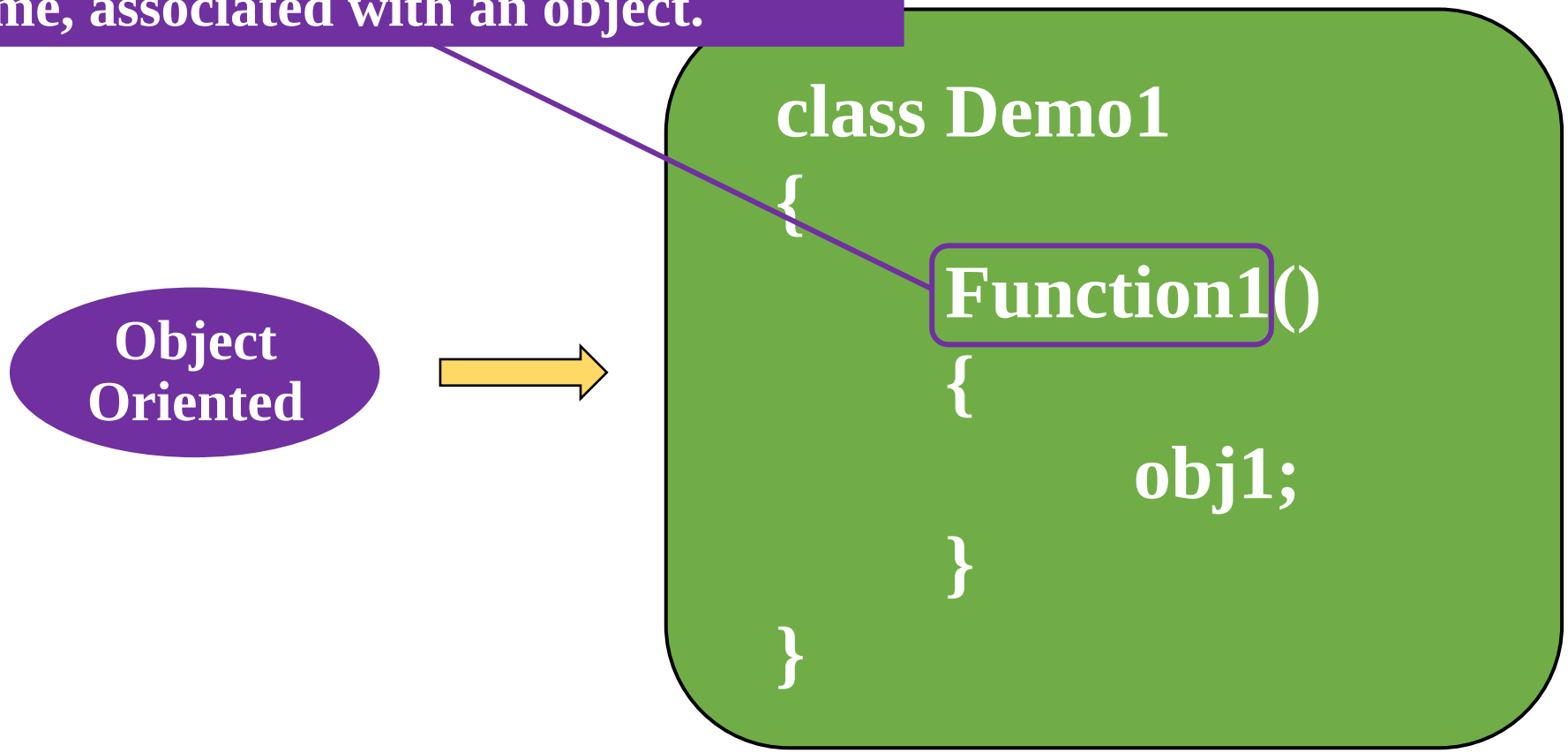
Object Oriented

```
class Demo1
{
        Function1()
        {
                obj1;
        }
}
```

Introduction to Java

“Function” is a block of code that is called by a name, associated with an object.

Object Oriented



Introduction to Java

Object Oriented

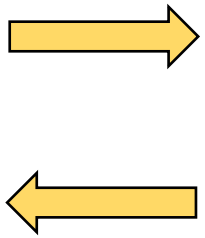
```
class Demo1
{
        Function1()
        {
                obj1;
        }
}
```

"Object" have identity, state and behavior.

Introduction to Java

Object Oriented

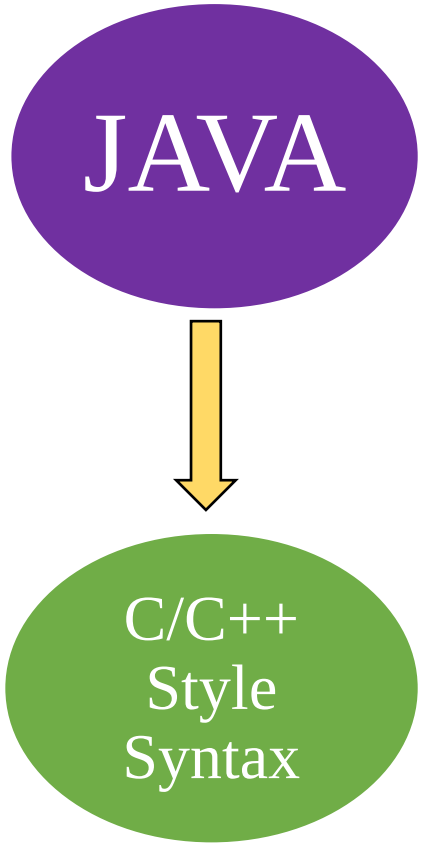
```
class Demo1
{
        Function1()
        {
                obj1;
        }
}
```



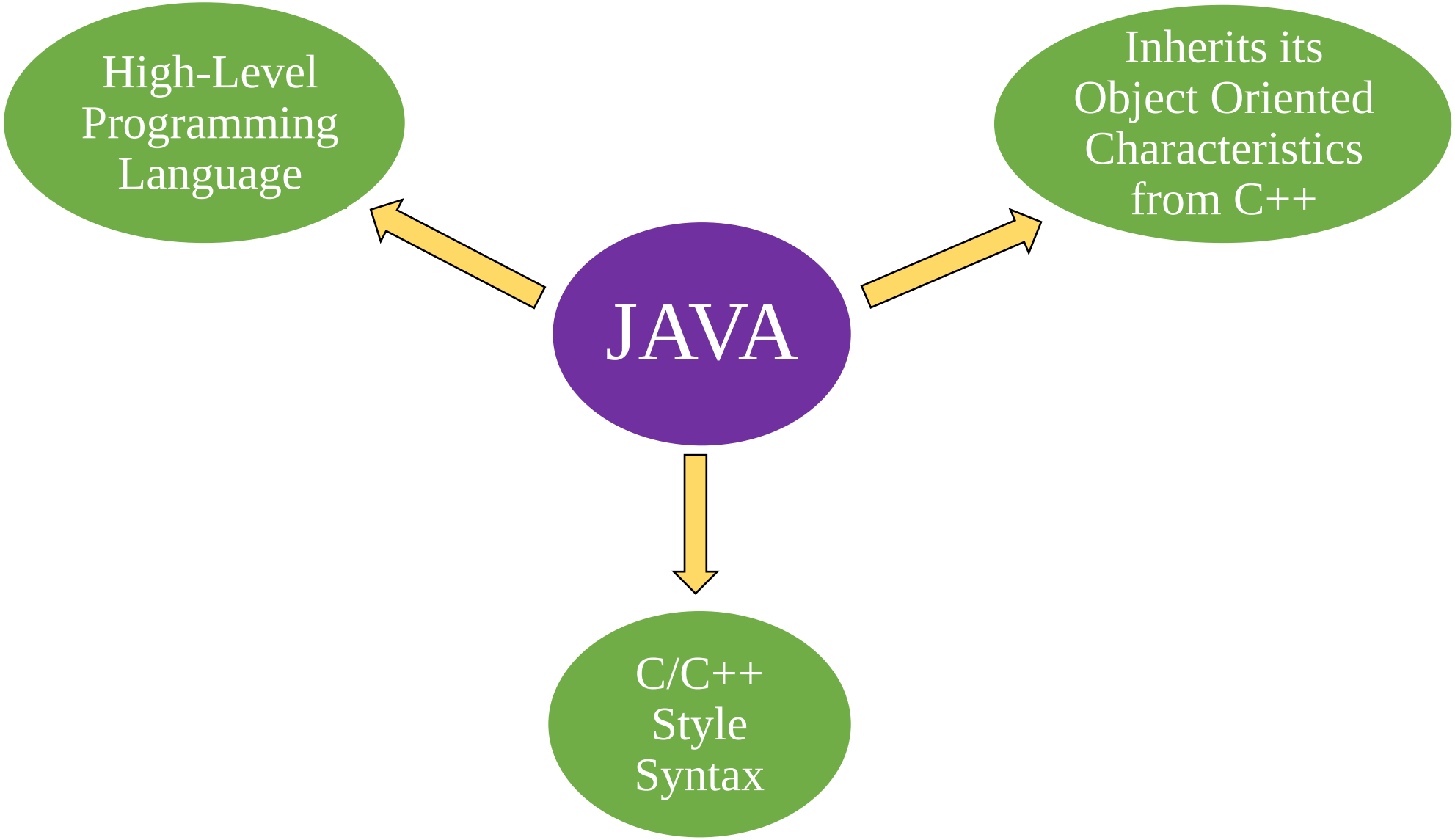
```
class Demo2
{
        Function2()
        {
                obj2;
        }
}
```

Introduction to Java

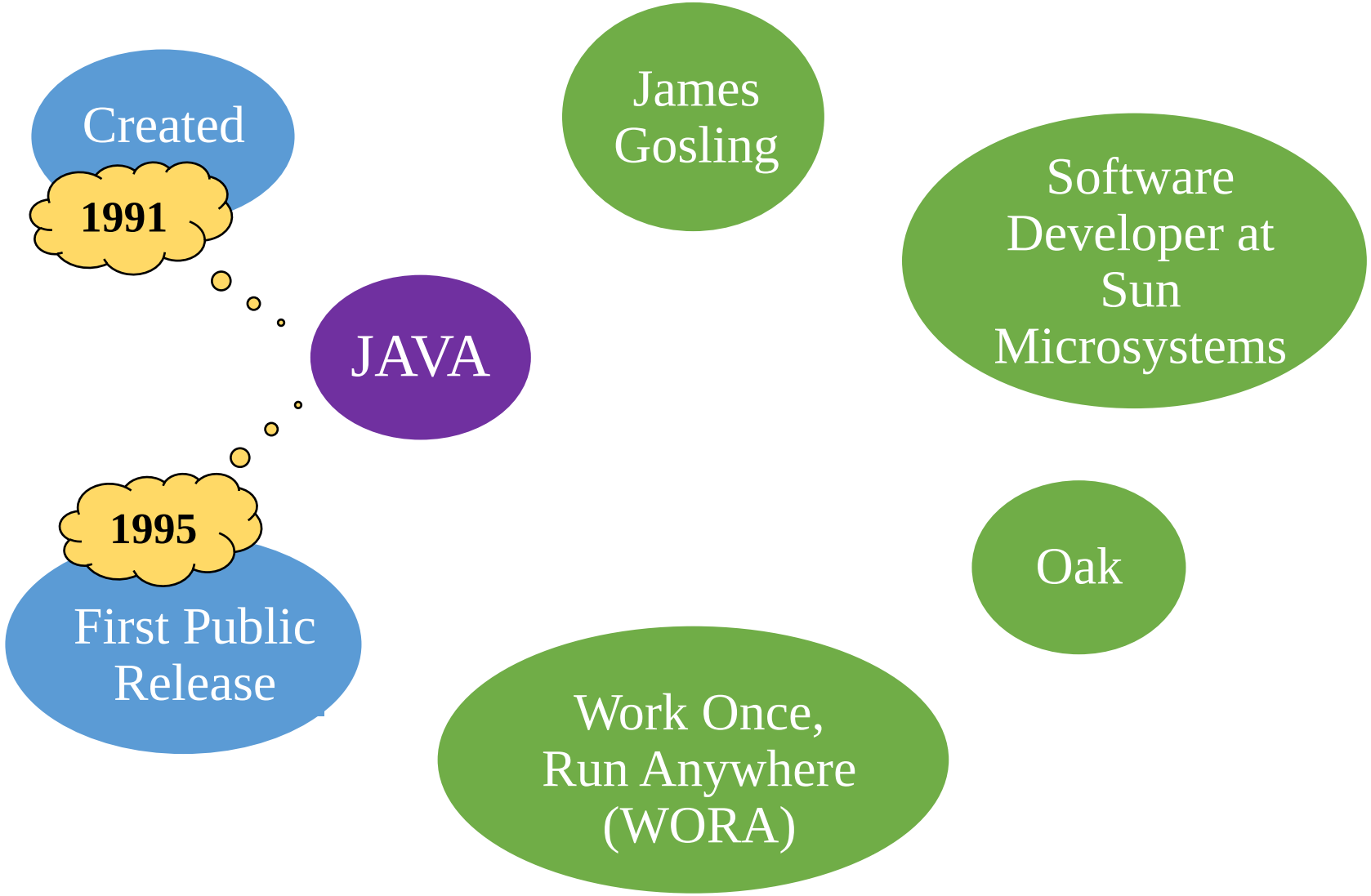
Java was designed based on C/C++ style syntax, as many programmers were familiar to it.



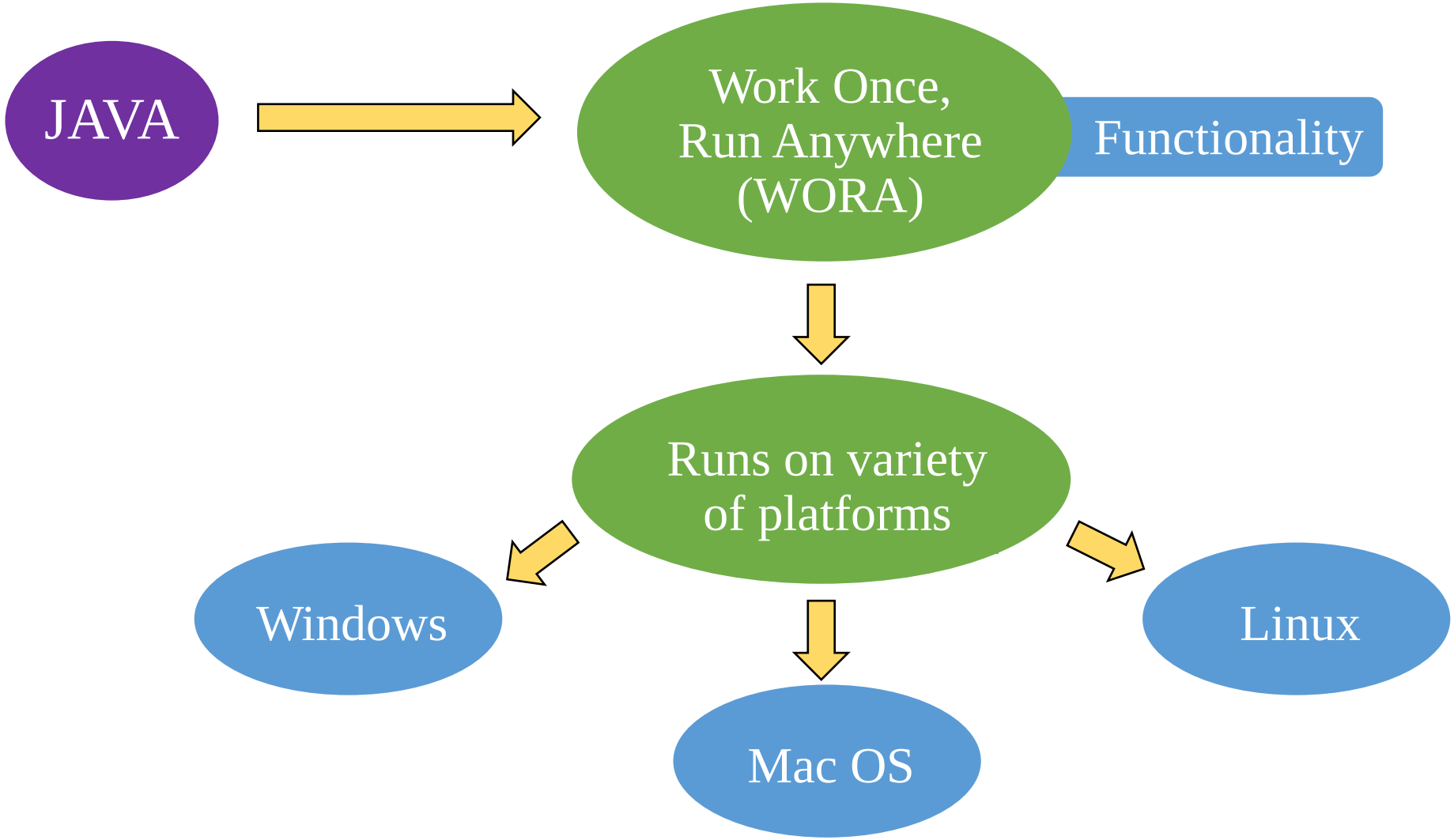
Introduction to Java



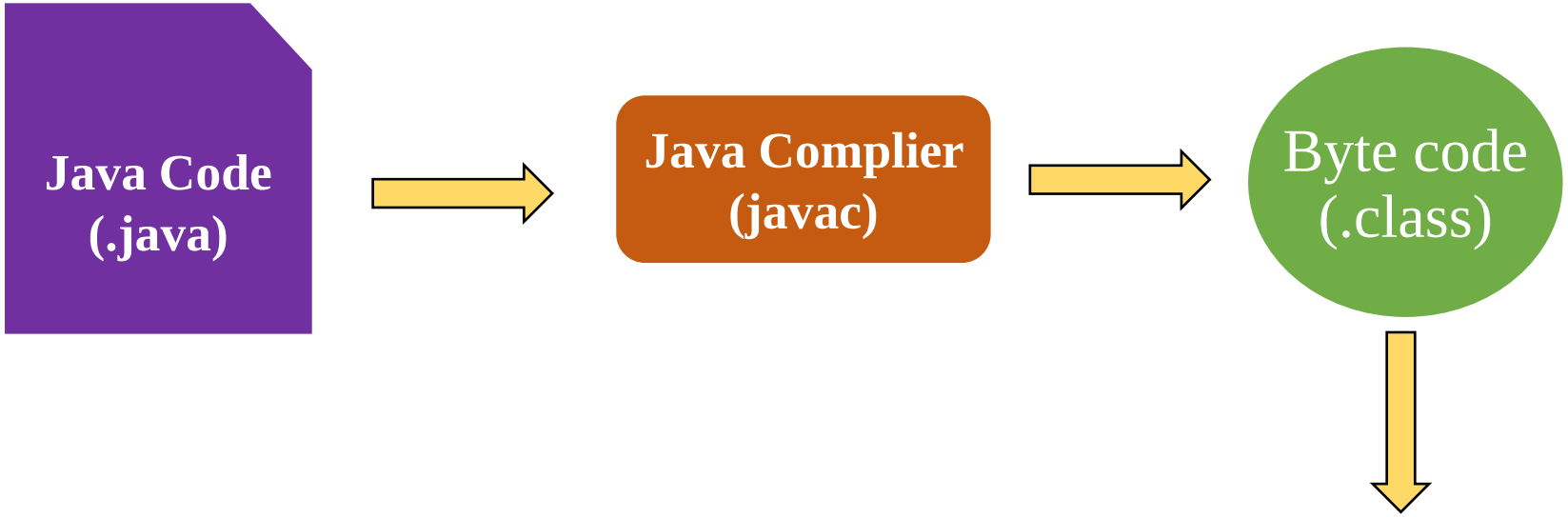
Introduction to Java



Introduction to Java

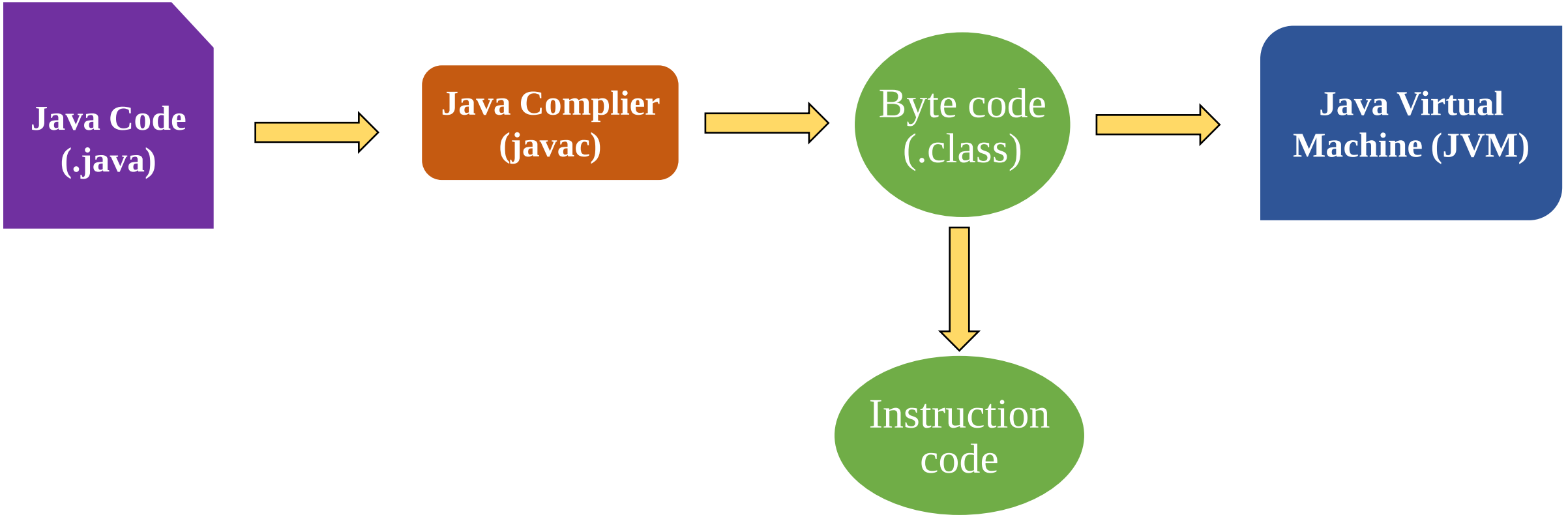


Introduction to Java

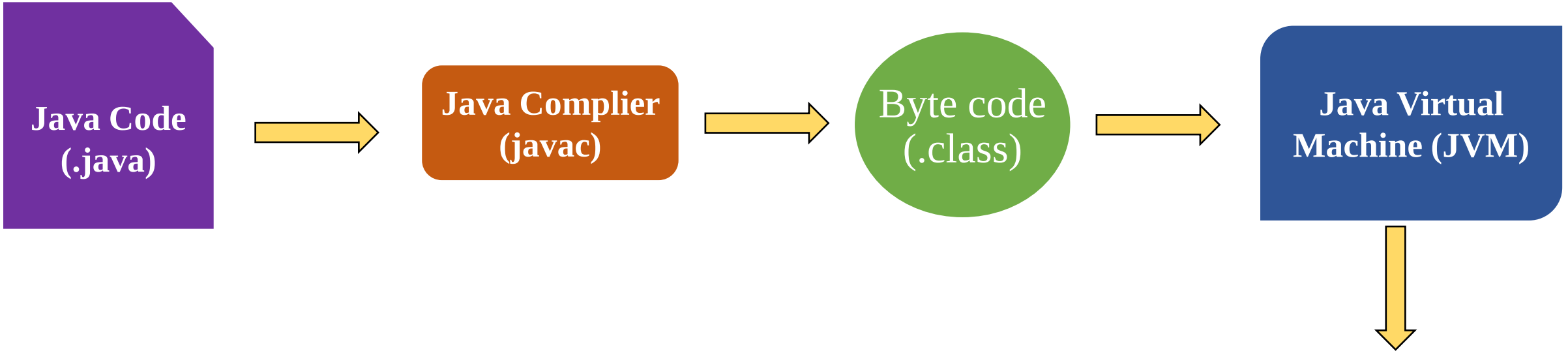


Java byte code is the instruction code that is processed by a Java Virtual Machine (JVM).

Introduction to Java

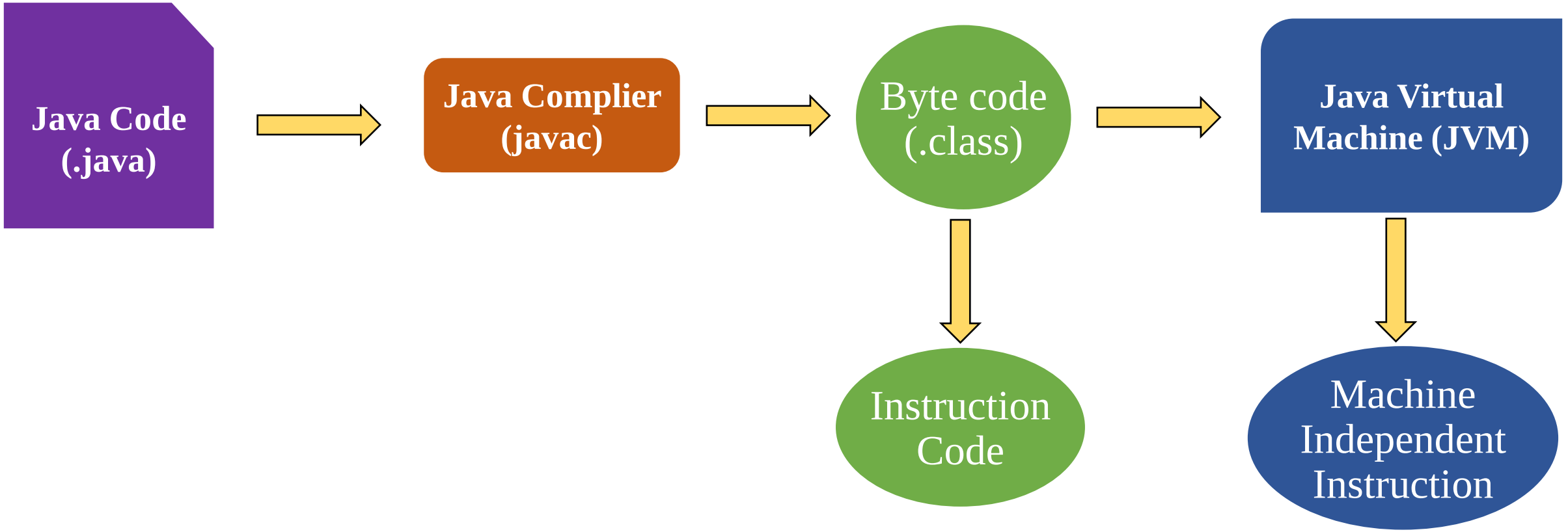


Introduction to Java

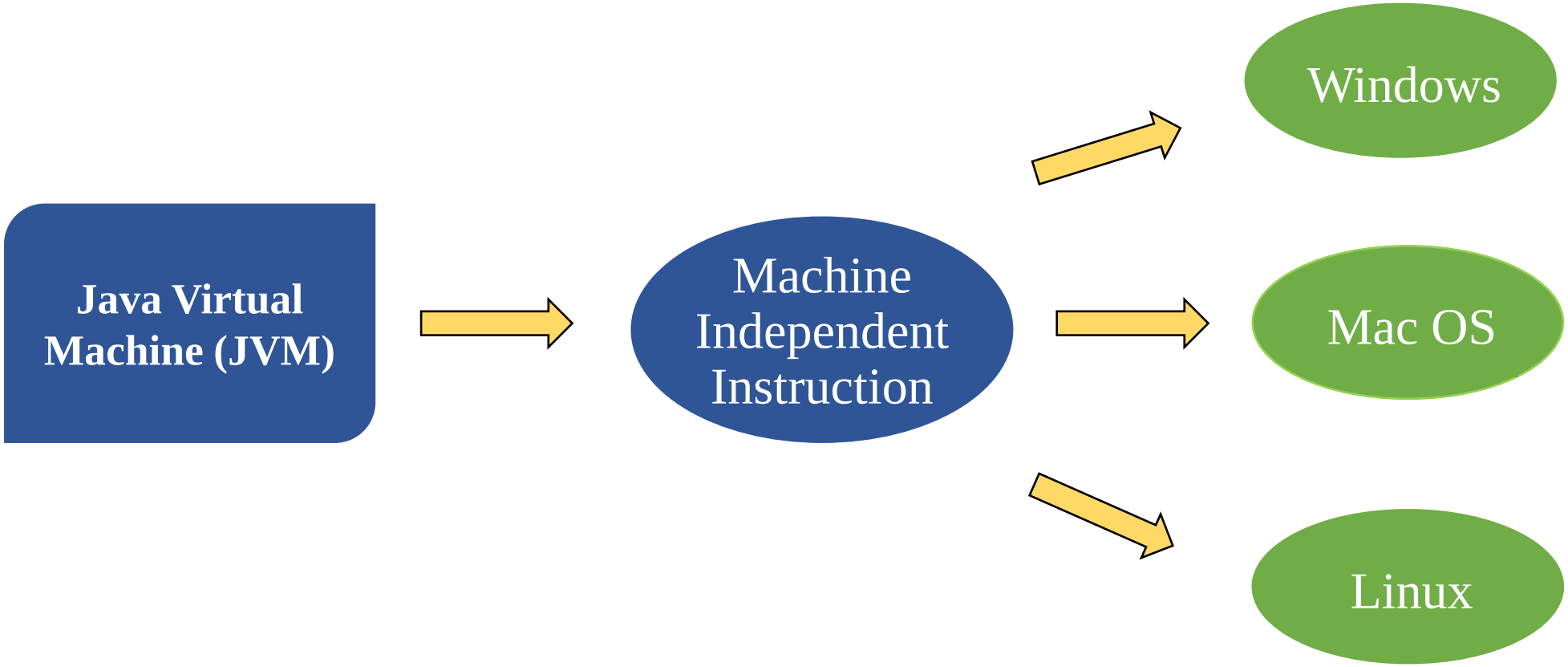


Java Virtual Machine (JVM) converts each instruction code into a machine independent instruction, which can run on any operating system.

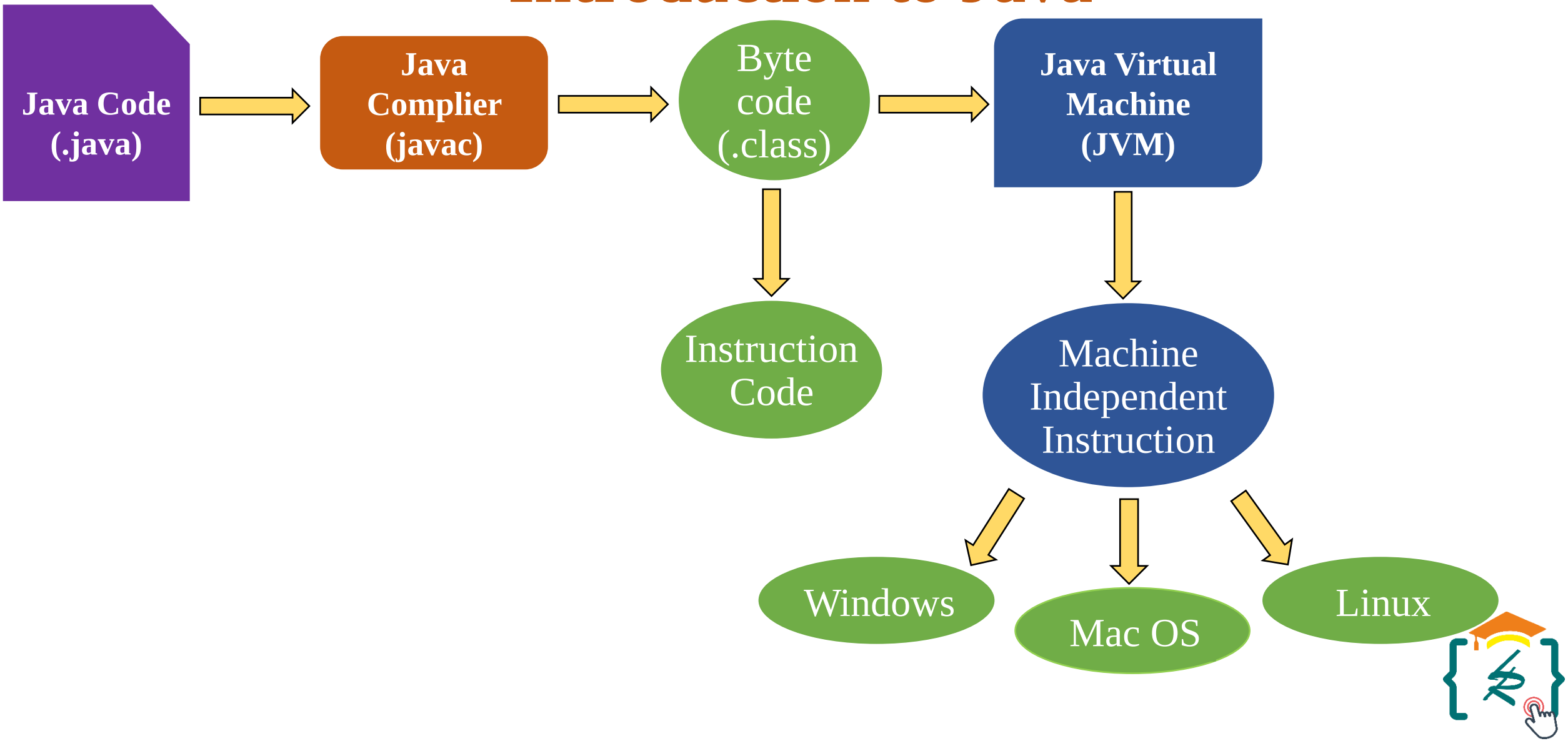
Introduction to Java



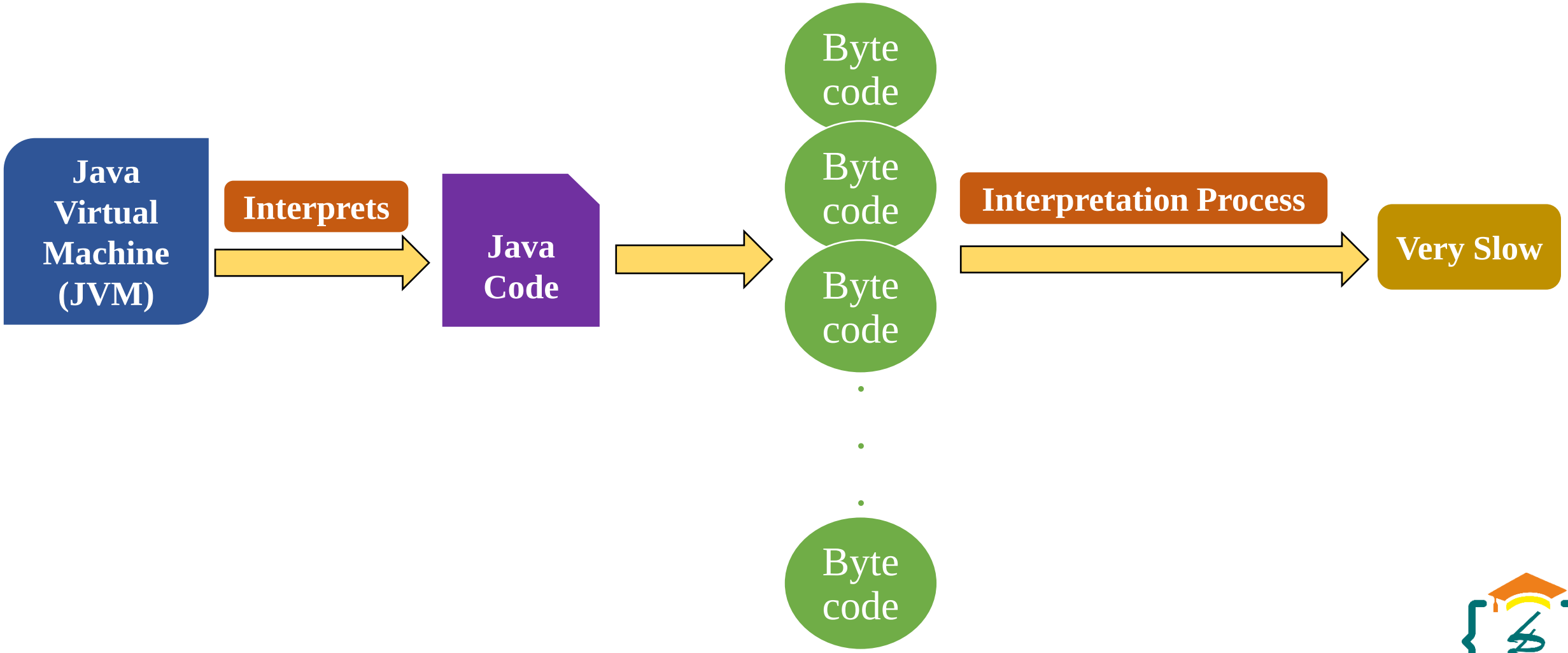
Introduction to Java



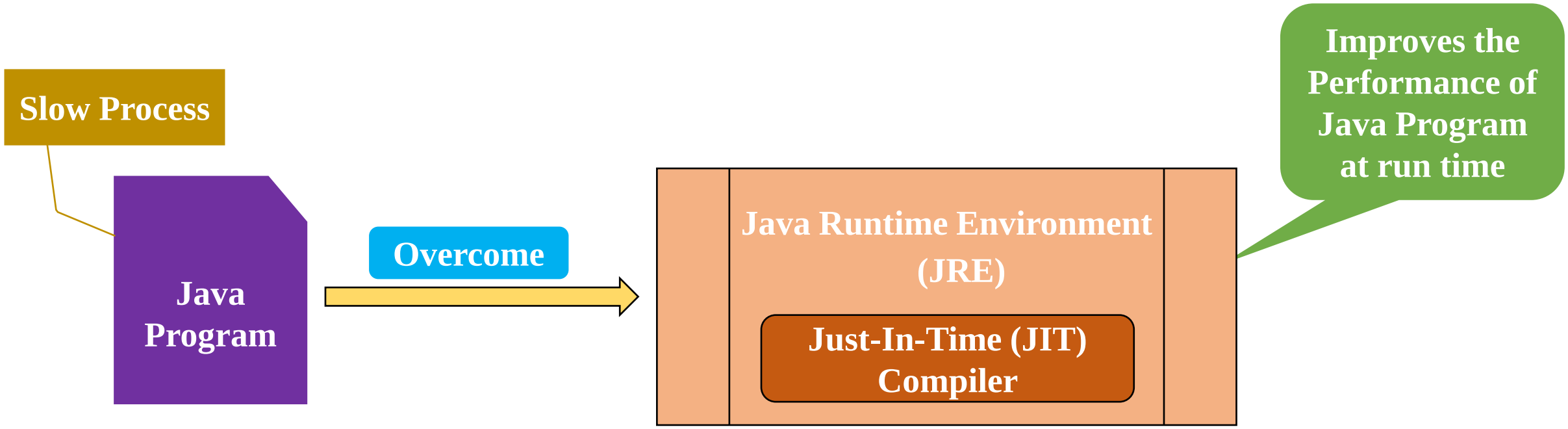
Introduction to Java



Introduction to Java

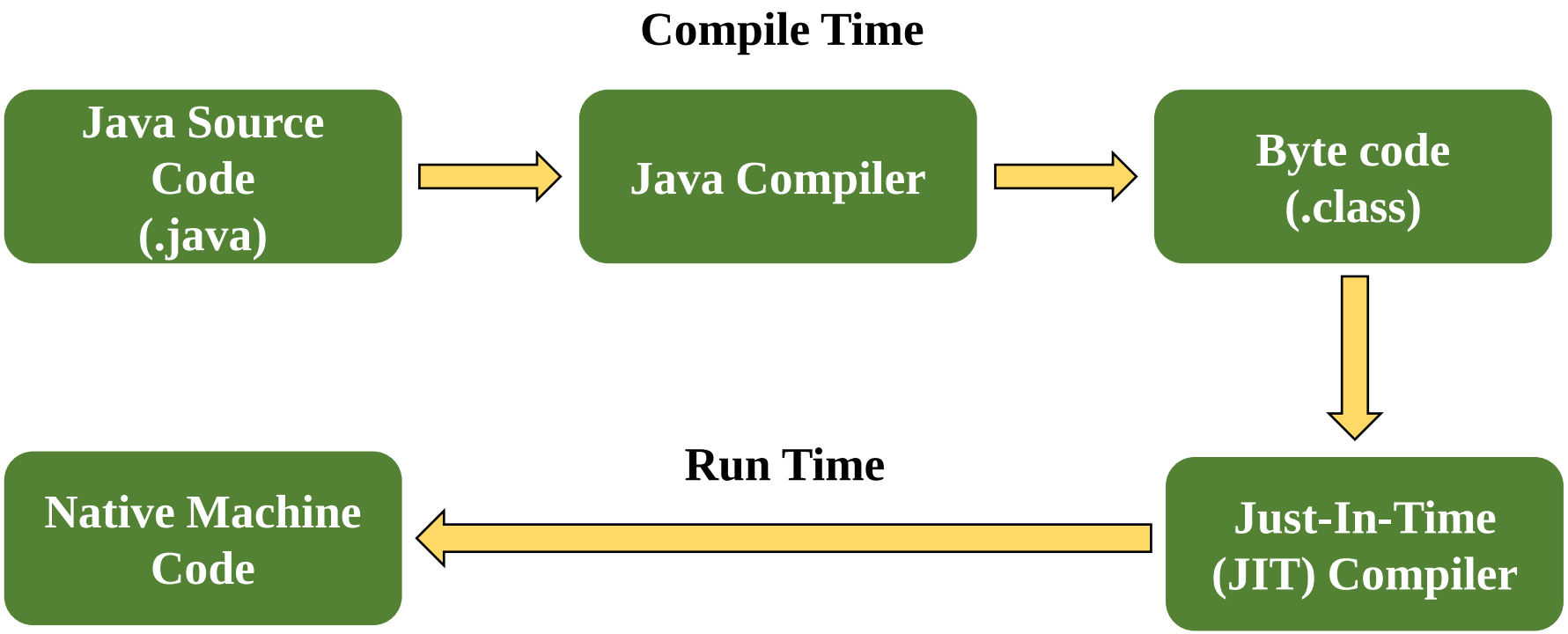


Introduction to Java



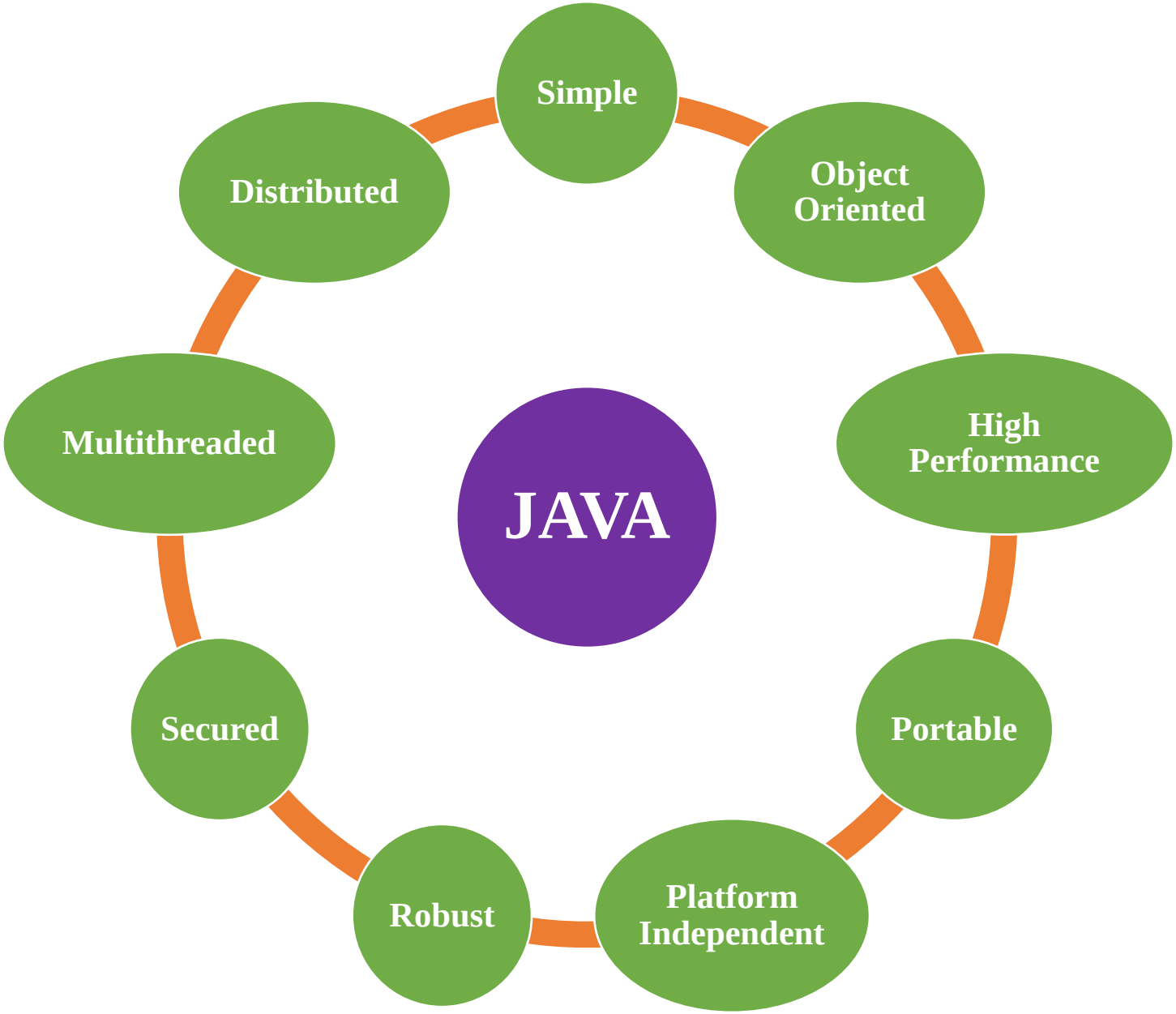
To overcome the slow process, Java introduced Just-In-Time (JIT) compiler, which is an integral component of the Java Runtime Environment (JRE) JIT compiler is responsible for improving the performance of Java applications at run time.

Introduction to Java



Compilation process of JRE

Features of Java



Features of Java

Simple	Java is very easy to learn programming language, as its syntax is very simple and clear in understanding.
Object Oriented	Java is an object-oriented programming language, were everything is an object, which has some data and behaviour.
High Performance	Java is faster as its code is compiled into byte code which is optimized by the Java compiler for Java Virtual Machine (JVM) to execute its applications faster with the help of Just-In-Time (JIT) compiler.

Features of Java

Portable	Java is portable as it provides the Java byte code to execute on any platform without any implementation.
Platform Independent	Java code is compiled into byte code format, which does not bound it to any specific operating system.
Robust	With features like strong memory management, automatic garbage collection and mechanism like exception handling makes Java a robust language.

Features of Java

Secured	**Java's secure features allow us to develop virus-free systems, as Java program runs in Java Runtime Environment (JRE) making it is more secure.**
Multithreaded	**Multithreading feature of Java makes it possible to write program that can perform many tasks simultaneously without occupying memory for each thread, instead it shares a common memory area, resulting in highly interactive and responsive applications.**
Distributed	**Java is a distributed language as it can create applications that can run over network, with the help of TCP/IP protocols used for communication.**

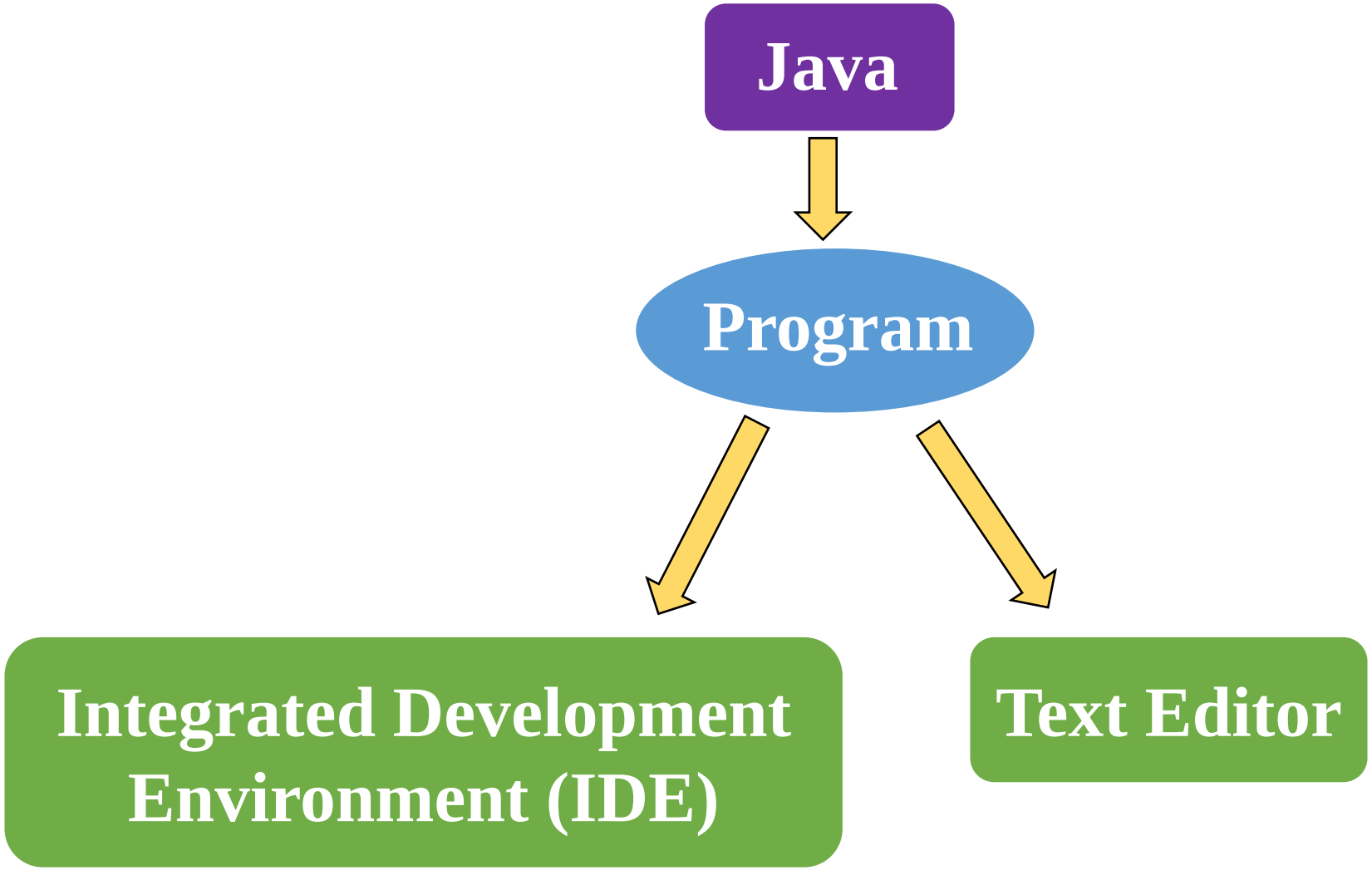
Any Questions… ???

Thank You

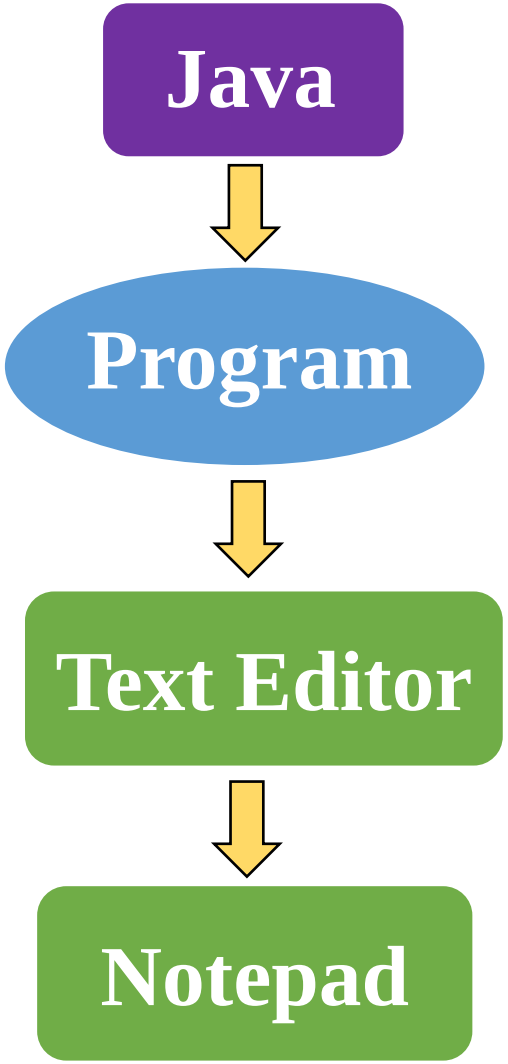
Sumeet Rathod

JAVA PROGRAMMING FORMAT

Java Programming Format



Java Programming Format



Java Programming Format

Program

```
class Demo
{
        public static void main(String [] args)
        {
                System.out.println(“Hello World”);
        }
}
```

Java Programming Format

Program Structure

```
class Demo
{

}
```

It is an object oriented construct.

Java Programming Format

Program Structure

```
class Demo
{

}
```

“class” is a keyword and declares a new class definition.

Java Programming Format

Program Structure

```
class Demo
{

}
```

“Demo” is a Java Identifier that specifies the name of the class to be defined.

Java Programming Format

Program Structure

```
class Demo
{
}
```

Java is a block structured language, so code blocks are always enclosed by curly braces “{” and “}”.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {

        }
}
```

"public" is an access modifier, whose access level is everywhere.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {

        }
}
```

"static" keyword declares the main() method as a global one and can be accessed without creating an object of the class.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {

        }
}
```

"void" keyword states that, the main() method does not return any value.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {

        }
}
```

"main()" method is the entry point of Java program, which is configured by JVM.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {

        }
}
```

The main() method accepts a single argument, which is an array of elements named as “args” of type “String”.

Java Programming Format

Program Structure

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

This statement is used to print an argument, which is passed to it.

Java Programming Format

Program Structure

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

"System" is a predefined class in "java.lang" package.

Java Programming Format

Program Structure

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

"out" is an instance of type PrintStream, which is a static member field of the System class.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {
                System.out.println(“Hello World”);
        }
}
```

“println()” is a method of PrintStream class, which prints the argument passed to the standard console and a newline.

Java Programming Format

Program Structure

```
class Demo
{
        public static void main(String [] args)
        {
                System.out.println(“Hello World”);
        }
}
```

Semicolon is used to end the statement.

Java Programming Format

Save the Program

Save the program with same name as the class name -

<ClassName>.java

Demo.java

Java Programming Format

Set Path

Method - 1

Java Programming Format

Set Path

Method - 1

1) Go to the following location and copy it **C:\Program Files\Java\jdk1.8.0_161\bin**

2) Press  + R → Run, type **cmd** and press OK.

Java Programming Format

Set Path

Method - 1

3) In cmd, type the following command, along with copied location in Step 1
set path="C:\Program Files\Java\jdk1.8.0_161\bin"
and type enter.

Java Programming Format

Set Path

Method - 1

4) Copy the location where your Java program is saved and type the following command followed by your copied location
cd C:\Users\SUMEET\Desktop\Java Programs and press enter.

Java Programming Format

Set Path

Method - 1

5) Compile the program - **javac <FileName>.java**
javac Demo.java

6) Execute the program - **java <FileName>**
java Demo

Java Programming Format

Set Path

Method - 2

Java Programming Format

Set Path

Method - 2

1) Go to the following location and copy it **C:\Program Files\Java\jdk1.8.0_161\bin**

Java Programming Format

Set Path

Method – 2

2) Right click on → This PC→ Go to properties → Advanced System Settings → **Environment Variables** → System Variables → Select Path and click on Edit → click on New → paste the copied location of Step 1 and click OK.

Java Programming Format

Set Path

Method - 2

3) Press  + R → Run, type **cmd** and press OK.

Java Programming Format

Set Path

Method - 2

4) Copy the location where your Java program is saved and type the following command followed by your copied location
cd C:\Users\SUMEET\Desktop\Java Programs and press enter.

Java Programming Format

Set Path

Method - 2

5) Compile the program - **javac <FileName>.java**
javac Demo.java

6) Execute the program - **java <FileName>**
java Demo

Java Programming Format

Output

```
Hello World
```

Java Environment

Java Development Kit (JDK)

java	It serves as Java interpreter.
javac	It serves as Java compiler.
javadoc	It creates HTML documentation for java source code files.

Java Environment

Java Development Kit (JDK)

javap	It serves as Java disassembler, which is used to convert byte code files into a java program description.
jdb	It serves as Java debugger.
jar	A jar (Java ARchive) serves as an archive used to package, related to class library into a single executable jar file.

Java Tokens

Java tokens are smallest elements of a program which are identified by the compiler.

In a Java program, all characters are grouped into symbols called tokens.

Java Tokens

The following are the Java tokens -

- Comments
- Identifiers
- Separators
- White Spaces
- Literals
- Operators
- Keywords

Java Tokens

Comments

There are two types of comments in Java -

- Single Line Comment
- Multi Line Comment

Java Tokens

Comments

There are two types of comments in Java -

Single Line Comment

```
class Demo
{
        //Statement
}
```

Java Tokens

Comments

There are two types of comments in Java -

Multi Line Comment

```
class Demo
{
        /*
        Statements
        */
}
```

Java Tokens

Identifiers

In programming languages, identifiers are used for identification purpose.

In Java, an identifier can be names of classes, methods, variables, packages and interface.

Java Tokens

Identifiers

```
class Demo
{
        public static void main(String [] args)
        {
                int x;
        }
}
```

1) Class name

Java Tokens

Identifiers

```
class Demo
{
        public static void main(String [] args)
        {
                int x;
        }
}
```

2) Method name

Java Tokens

Identifiers

3) Predefined Java class name

```
class Demo
{
        public static void main(String [] args)
        {
                int x;
        }
}
```

Java Tokens

Identifiers

```
class Demo
{
        public static void main(String [] args)
        {
                int x;
        }
}
```

4) Array name

Java Tokens

Identifiers

```
class Demo
{
        public static void main(String [] args)
        {
                int x;
        }
}
```

5) Variable name

Java Tokens

Separators

Symbol	Name	Purpose
()	Parentheses	Used to contain lists of parameters in method definition and invocation.Also used for defining precedence in expressions, containing expressions in control statements, and surrounding cast types.
{}	Braces	Used to contain the values of automatically initialized arrays. Also used to define a block of code, for classes, methods, and local scopes.
[]	Brackets	Used to declare array types. Also used when dereferencing array values.
;	Semicolon	Terminates statements.
,	Comma	Separates consecutive identifiers in a variable declaration. Also used to chain statements together inside a for statement.
.	Period	Used to separate package names from subpackages and classes. Also used to separate a variable or method from a reference variable.

Java Tokens

White Spaces

The java.lang.Character.isWhitespace() is an inbuilt method in a java that determines if the specified character (Unicode code point) is white space according to Java.

Unicode is an Information Technology (IT) standard for the consistent encoding, representation, and handling of text expressed in most of the world's writing systems. It provides a unique number for every character, no matter what platform, device, application or language.

Java Tokens

Literals

Any constant value, which can be assigned to a variable is called as Literal.

```
int x = 10;
```

Data type Variable Constant Value

Keyword Identifier Literal

Thank You