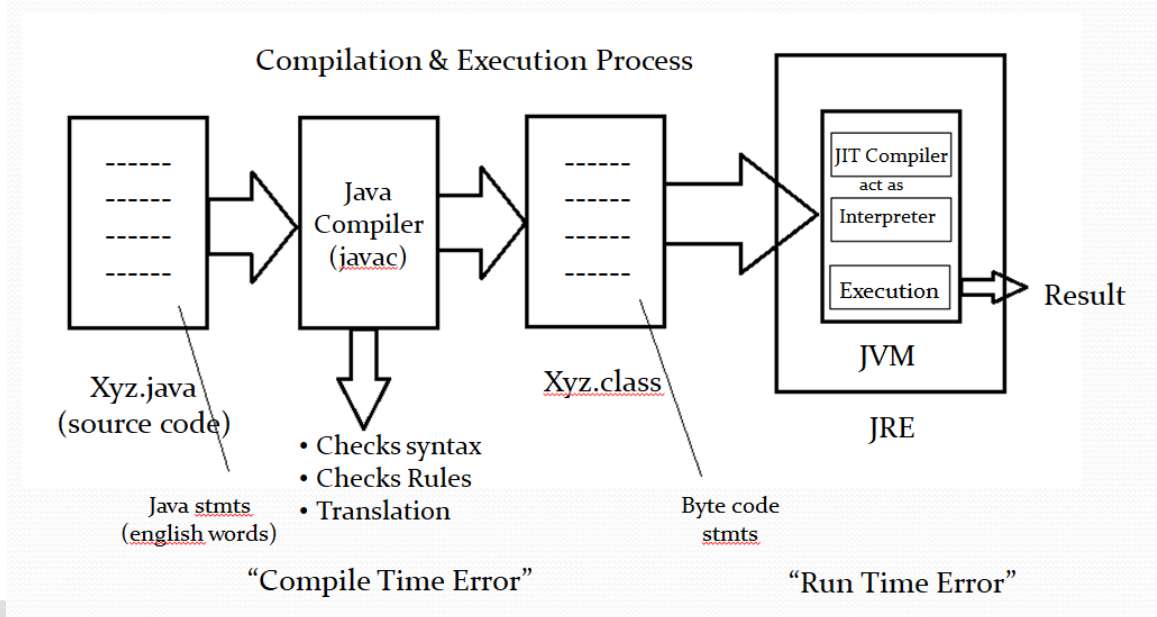## What is JAVA...?

- ✓ Java is a general purpose, **high-level programming language** developed by **Sun Microsystems.**

- ✓ A small team of engineers, known as the *Green Team*, initiated the language in 1991.

- ✓ Java was originally developed by **James Gosling.**

- ✓ Java was originally called *OAK,* and was designed for handheld devices and set-top boxes.

- ✓ Oak was unsuccessful, so in 1995 Sun changed the name to Java and modified the language.

JAVA Components

JIT – Just In Time
JVM – Java Virtual Machine
JRE – Java Runtime Environment

Compilation & Execution Process

Xyz.java (source code) → Java Compiler (javac) → Xyz.class → JIT Compiler act as Interpreter → Execution → Result

JVM
JRE

Java stmts (english words) — • Checks syntax • Checks Rules • Translation

Byte code stmts

"Compile Time Error"   "Run Time Error"

- ✓ Developing a pgm in Java language includes 3 steps

  - o Source code creation

  - o Source code compilation

  - o Execution

- ✓ The source code creation is done by writing Java pgm using the syntax of Java language.

- ✓ The source code should be saved with the extension of **.java**

- ✓ We can create the source code using any text editor or IDE tools.

- ✓ The source code is compiled to get executable format.

- ✓ Java compiler is used to do the compilation of source code.

- ✓ Java compiler checks the syntax & rules before compiling it.

- ✓ If any syntax or rules mistakes, then compiler throws an error called "**Compiler error**".

- ✓ The java compiler translates the Java stmts into **Byte Codes.** The byte codes are saved in the files with the extension **.class**

- ✓ The execution of the pgm is done by JVM, inside JVM the JIT compiler compiles the byte codes to the m/c level formats & it is executed by JVM.

- ✓ The JRE is responsible to provide the necessary environment to the JVM. So that the JVM can executes the byte codes.

- ✓ The .class file is OS independent but JRE dependent. We can run the .class files on any OS provided the JRE's available in that system. This is known as "**Platform Independent**".

- ✓ The Java s/w is released in "**Development Tool Kit**" known as **JDK** which contains the necessary development tools like **Java compiler, JRE & other libraries.**

- ✓ There are 2 types of JRE

  - o Public JRE – used whenever the java pgms run on **Server**.

  - o Private JRE – used whenever the java pgms run on **local m/c.**

Features of Java

- ✓ **Simple** - confusing features in C++ are removed in Java like pointers etc..

- ✓ **Secure** - provides data security through encapsulation.

  - Programmes run within the JVM which protects from unauthorized access to system resources.

- ✓ **Portable** - Bytecode helps Java to achieve portability.

- ✓ **Object – Oriented** - it supports all the features of object oriented model like: Encapsulation, Inheritance Polymorphism & Abstraction.

- ✓ **Robust** - Type checking & Exception handling helps to make the programs robust.

- ✓ **Multithreaded** - supports multithreading which is not supported by C and C++.

- ✓ **Architecture neutral** - Since Java applications can run on any kind of CPU, Java is architecture – neutral.

- ✓ **Interpreted & High Performance** - JIT compiler converts the byte code into machine code piece by piece and caches them for future use. This enhances the program performance means it executes rapidly.

✓ **Distributed** - supports distributed computation using Remote Method Invocation (RMI) concept.

✓ **Dynamic** - The Java Virtual Machine (JVM) maintains a lot of runtime information about the program and the objects in the program.
  - Libraries are dynamically linked during runtime.

Compilation & Execution

- Create source file -> MyFirstPgm.java

```
Class MyFirstPgm
{
        public static void main(String[] args)
        {
                System.out.println("Welcome to class");
        }
}
```
→ Source file

- Go to command prompt and open respective path
- Compilation - >javac MyFirstPgm.java
- Execution - >java MyFirstPgm
- Result – Welcome to class

→ Steps

Java Operators

Types	Operators
Arithmetic Operators	+, -, *, /, %, ++ , --
Relational Operators	==, !=, >, <, >=, <=
Bitwise Operators	&, |, ^, ~, <<, >>, >>>
Logical Operators	&&, ||, !
Assignment Operators	=, +=, -=, *=, /=, %=, <<=, >>=, &=, ^=, |=
Misc Operators	? :, instanceof, new, .(dot)
Unary Operators	++, --

"+" Operator

✓ Java supports operator overloading concept only for + Operator.

  o Addition of numbers

    ▪ 20 + 40 = 60

- o Concatenation of Strings

    - ▪ "java" + "developer" = javadeveloper

    - ▪ "java"+ 10 = java10
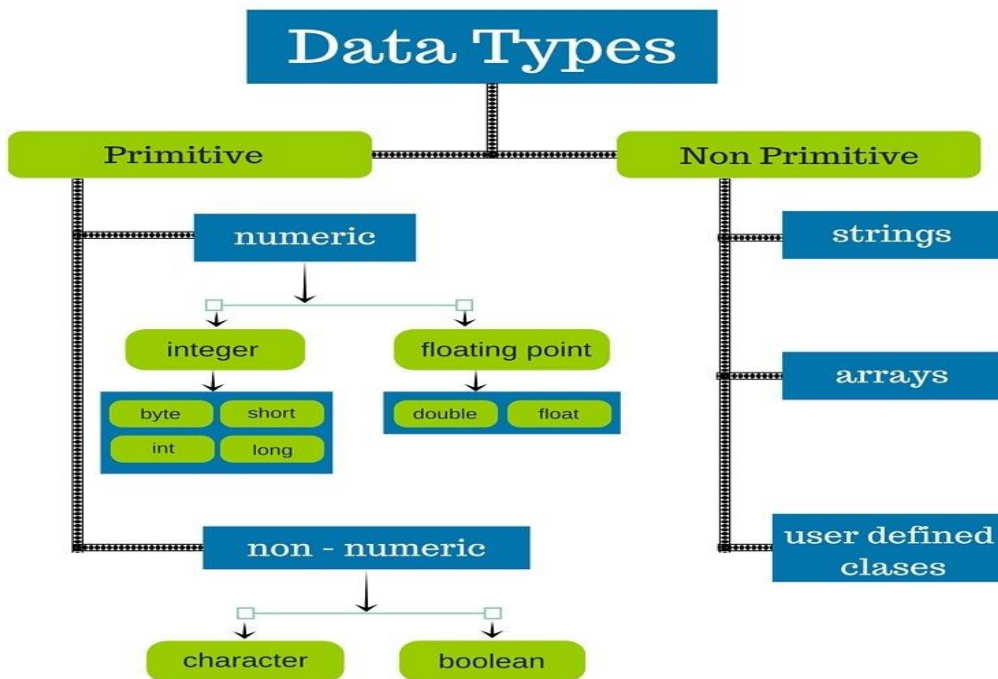
Keywords

- ✓ Keywords are predefined preserved word which is used for particular purpose.

- ✓ Each keyword has its own meaning & user cannot modify the meaning.

- ✓ The programmer can built the programme by using keywords.

- ✓ In java language all the keywords are represented in "lower case".

Identifiers

- ✓ Identifiers are used to represent a value in the programme.

- ✓ While using the identifiers we should follow the below rules

    - o An identifier can be Alpha-Numeric characters.

    - o All identifier should begin with Alphabets only. If it begins with numeric compiler throws error.

    - o Special character "_" and "$" can be used.

    - o It should not have any space.

Valid identifiers	Invalid identifiers
empid	123empid
empid123	emp id
emp_id	emp@id

Control Statements

- ✓ Control statements control the order of execution in a program, based on data values and conditional logic.

- ✓ Types

    - o If statement

    - o If – else statement

    - o If – else ladder statement

    - o Nested if statement

    - o Switch statement

- ✓ These are also known as **selection statements**.

Looping Statements

- ✓ Looping statements repeat a specified block of code until a given condition is met.

- ✓ **Types**

    - o While loop

    - o Do – while loop

    - o For loop

Functions/Methods

- ✓ Functions are used to defined the operation or the task in a program.
- ✓ By developing functions we can achieve modularity & code reusability.
- ✓ While developing a pgm each task is built or coded using the function
- ✓ The syntax of declaring & defining a function is :

<access specifiers> <modifiers> returntype function_name(<argument lists>)

{

   ----

   return value;

}

- ✓ The function arguments are used to pass values to the function body.
- ✓ The function arguments should be declared in the function declaration line.
- ✓ We can declare function without argument or with argument.
- ✓ We can declare function with multiple arguments, multiple arguments should be separated by comma(,)
- ✓ The function argument is local to the function body.

Function Returntype

- ✓ The function return type specifies the type of value returning by the functions.
- ✓ We should specify the data type in the return type field.
- ✓ A function can return a value by using "return" keyword.
- ✓ If a function doesn't want to return a value, then in the return type we should mention "void".
- ✓ A function can return only one value at a time.

How to read inputs from keyboard

- ✓ Step 1: improt java.util.Scanner;
- ✓ Step 2: create Scanner class object
  - ○ Scanner scn = new Scanner(System.in);
- ✓ Step 3: Use funtions to read inputs
  - ○ To read int value from keyboard
  - ○ int x = scn.nextInt();
  - ○ To read String value from keyboard
  - ○ String st = scn.next();
  - ○ To read double value from keyboard
  - ○ double y = scn.nextDouble();

Arrays

- ✓ Declaration & Initialisation
  - ○ Method 1: By specifying size

    datatype[] referenceVariable = new datatype[size];

referenceVariable[0] = value 1;

. . .

referenceVariable[size-1] = value n;

- o Method 2: Using array initializer

  datatype[] referenceVariable = {value 1, value 2, . . . , value n};