JAVA

1. WHAT IS APPLICATION ?

- APPLICATION IS A PIECE OF SOFTWARE OR LIST OF PROGRAM WHICH IS USED TO PERFORM SOME SPECIFIC TASK.

2. <u>TYPES OF APPLICATION</u> :

1. STAND ALONE APPLICATION.
2. WEB APPLICATION.
3. CLIENT SERVER/MOBILE APPLICATION.

<u>STAND ALONE APPLICATION</u> :

- SOFTWARE INSTALLED IN ONE C OMPUTER AND USED BY ONLY ONE PERSON.
- INTERNET NOT REQUIRED AND NO NEED TO INSTALL .

<u>EXAMPLE</u> :

CALCULATOR, ADOBE PHOTO SHOP, MS OFFICE, PAINT ETC..

<u>WEB APPLICATION</u> :

- ANY APPLICATION WHICH IS OPENED THROUGH A BROWSER IS KNOWN AS WEB APPLICATION.
- INTERNET REQUIRED AND NO NEED TO INSTALL.

<u>EXAMPLE :</u>

FACEBOOK , GMAIL , AMAZON. ETC ..

<u>SERVER :</u> IT IS NOTHING BUT A SUPER COMPUTER WHERE IN ALL THE APPLICATIONS ARE INSTALLED AND CAN BE ACCESSED BY ANYONE .

<u>CLIENT SERVER / MOBILE APPLICATION</u> :

- ANY SERVER APPLICATION WHICH IS ACCESSABLE THROUGH CLIENT APPLICATION IS CALLED CLIENT SERVER APPLICATION.
- INTERNET REQUIRED AND NEED TO INSTALL .

<u>EXAMPLE :</u>

WHATS APP , INSTAGRAM , YOUTUBE , FLIPKART ETC ….

3. **WHAT IS PROGRAMMING LANGUAGE ?**

- ONE PERSON COMMUNICATE WITH OTHER PERSON WE NEED A LANGUAGE, SO WE ARE USING TAMIL, ENGLISH ETC...
- SIMILARLY, WE NEED TO COMMUNICATE WITH COMPUTER WE NEED A PROGRAMMING LANGUAGE.
- PROGRAMMING LANGUAGE IS USED TO INSTRUCT THE COMPUTER TO PERFORM USER GIVEN TASKS.
- COMPUTER CAN UNDERSTAND ONLY MACHINE LANGUAGE LIKE 0'S AND 1'S .

4. **<u>TYPES OF LANGUAGES</u> :**

1.PROGRAMMING LANGUAGE - **C , C++ , JAVA** .
2.SCRIPTING LANGUAGE - **JAVA SCRIPT , PYTHON** .
3.MARKUP LANGUAGE - **HTML** .

- IT IS SPECIFIED INTO **LOW LEVEL** AND **HIGH LEVEL LANGUAGE.**

LOW-LEVEL LANGUAGE :

<u>MACHINE LANGUAGE</u> : IT IS A LOW-LEVEL LANGUAGE.

- HERE WE CAN WRITE THE INSTRUCTION BY BINARY CODE LIKE 0'S ANS 1'S .
- PROGRAMMER CANNOT UNDERSAND THE BINARY CODE EASILY BUT THE COMPUTER CAN UNDERSTAND THE BINARY CODE EASILY .
- HERE THE EXECUTION IS FAST BECAUSE WE WRITE THE CODE IN BINARY FORMAT.

<u>ASSEMBLY LANGUAGE</u> : IT IS A LOW-LEVEL LANGUAGE.

- ASSEMBLY LANGUAGE CONTAINS SOME PREDEFINED SYMBOLS AND NUMBERS THAT IS CALLED MNEMONIC.

EG(ADD 3,4 & MOV LOCATION,10.

- PROGRAMMER CAN UNDERSAND EASILY BUT THE COMPUTER CANNOT UNDERSTAND EASILY.
- HERE (COMPILER)ASSEMBLER WILL CONVER THE ASSEMBLY LANGUAGE CODE INTO MACHINE CODE .
- EXECUTION IS LESS THEN MACHINE LANGUAGE.

HIGH-LEVEL LANGUAGE :

- HIGH-LEVEL LANGUAGE IS EASY TO READ, WRITE AND MAINTAIN THE CODE FOR (USER OR HUMAN).
- HIGH-LEVEL LANGUAGE CONTAINS COMPILER OR INTERPRETER .

5. WHAT IS JAVA ?

- JAVA IS A OBJECT ORIENTED PROGRAMMING LANGUAGE .
- JAVA WAS DEVELOPED BY **JAMES GOSLING** AT **SUN MICROSYSTEM** IN **1995 MAY .**
- CURRENTLY JAVA WAS OWNED BY **ORACLE** .
- EVERY 6 MONTHS(**MARCH AND SEPTEMBER**) JAVA NEW VERSION WAS RELEASED.
- JAVA IS A PLATFORM INDEPENDENT BUT JVM IS DEPENDENT.
- CURRENT VERSION IS JDK22.

6. WHY JAVA IS POPULAR OR WHY WE USE JAVA WIDELY?

- IT IS A PLATFORM INDEPENDENT LANGUAGE , STRONG COMMUNITY SUPPORT , OPEN SOURCE , SECURE ETC.. (WORA) WRITE ONCE RUN ANYWHERE WIHICH CONTAINS JVM .

7. JAVA FEATURES ?

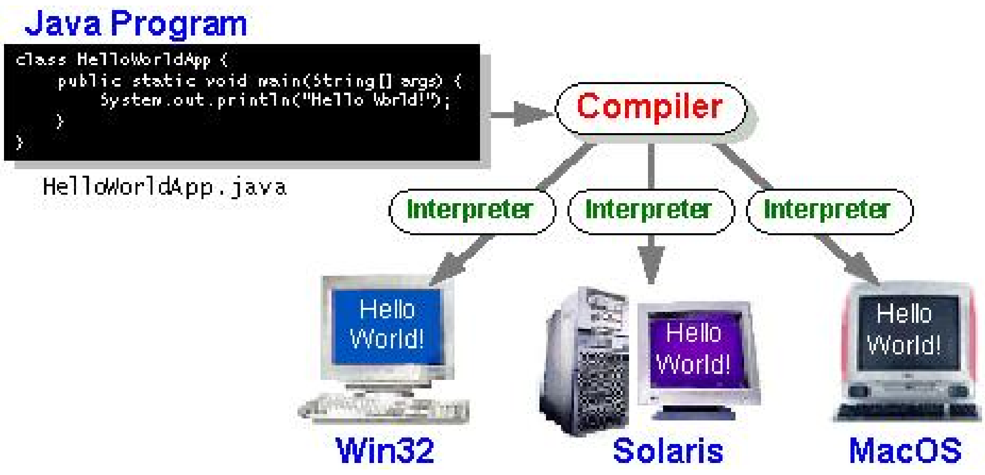
I. **SIMPLE** - JAVA IS SIMPLE AND EASY TO UNDERTAND, BECAUSE OF JAVA REMOVE POINTER,TEMPLATE AND OPERATOR OVERLOADING ETC..

- JAVA SYNTAX DESIGNED EASY TO UNDERTAND FOR USERS.

II. **OBJECT ORIENTED** - JAVA SUPPORTS OOPS CONCEPTS LIKE ENCAPSULATION , INHERITANCE , POLYMORPHISM AND ABSTRACTION .

- ALL CLASSES IN JAVA IS OBJECT TYPE EXCEPT PRIMITIVE DATA TYPES .

III. **PLATFORM INDEPENDENT** - JAVA CODE CAN COMPILED TO BYTE CODE AND RUN ON ANY DEVICE THAT HAS A JVM .TAKING THE COMPILED CODE(.CLASS)FILE AND RUN ON ANOTHER OS WITHOUT ANY RECOMPILATION .



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

IV. **SECURED** - JAVA DOESNOT SUPPORT POINTER CONCEPT.(POINTER MEANS ONE VARIABLE HOLD THE ANOTHER VARIABLE REFERENCE).

- JAVA CONTAINS AUTHENTICATION FUNCTIONALITIES AND AUTHORIZATION SERVICE.
- JAVA PROGRAM RUN ON SANDBOX (SEPERATE THE PATH TO RUN ALL PROGRAM) VIRTUAL MACHINE, BEFORE EXECUTE JAVA PROGRAM BYTE CODE VERIFIER VERIFIES THE BYTE CODE .

V. **ROBUST** - IT UTILIES STRONG MEMORY MANAGEMENT BECAUSE OF GARBAGE COLLECTOR CLEAR THE UNUSED MEMORY.

- JAVA CAN HANDLE THE EXCEPTION. SO WE CANNOT RUN THE PROGRAM UNTIL THE EXCEPTION HANDLE PROPERLY.

VI. **DYNAMIC** - JAVA SUPPORTS THE DYNAMIC BECAUSE OF LOADING THE CLASSES AND EXECUTE AT RUN TIME .

VII. **INTERPRETED** - JAVA CAN BE CONSIDERED BOTH COMPILED AND INTERPRETED LANGUAGE.

- INTERPRETER WILL READ AND EXECUTE THE CLASS FILE INTO OUTPUT.
- IT HELPS TO CONVERT A HIGH-LEVEL PROGRAM STATEMENTS INTO A MACHINE CODE .

VIII. **MULTITHREADED** - JAVA SUPPORTS MULTIPLE TASK TO PERFORM SIMULTANEOUSLY.(GAMING AND ANIMATION)

IX. **HIGH PERFORMANCE** - JAVA BYTECODE IS "CLOSE" TO NATIVE CODE(MACHINE CODE).

X. **PORTABLE** - JAVA IS PORTABLE BECAUSE OF JAVA BYTE CODE CARRY TO ANY OTHER PLATFORM.

• PORTABLE MEANS THAT YOU CAN RECOMPILE THE SOURCE CODE OF A PROGRAM TO A PLATFORM-SPECIFIC EXECUTABLE WITHOUT CHANGING THE SOURCE CODE ITSELF, YOU JUST NEED TO COMPILE IT.

X1. **ARCHITECTURAL NEUTRAL** - JAVA IS ARCHITECTURE NEUTRAL BECAUSE THERE ARE NO IMPLEMENTATION DEPENDENT FEATURES, FOR EXAMPLE, THE SIZE OF PRIMITIVE TYPE IS FIXED.

X12. **DISTRIBUTED** - BECAUSE IT FACILITATES USERS TO CREATE DISTRIBUTED APPLICATIONS IN JAVA.

• REMOTE METHOD INVOCATION (RMI) AND ENTERPRISE JAVA BEANS (EJB) ARE USED FOR CREATING DISTRIBUTED APPLICATIONS

8. **DIFFERENCE BETWEEN C, C++ AND JAVA ?**

C

PROCEDURAL LANGUAGE
DENNIS RITCHIE IN 1972
COMPILER ONLY
PLATFORM DEPENDENT
FILE EXTENSION .C
FILE GENERATED .EXE FILE
POINTER SUPPORT

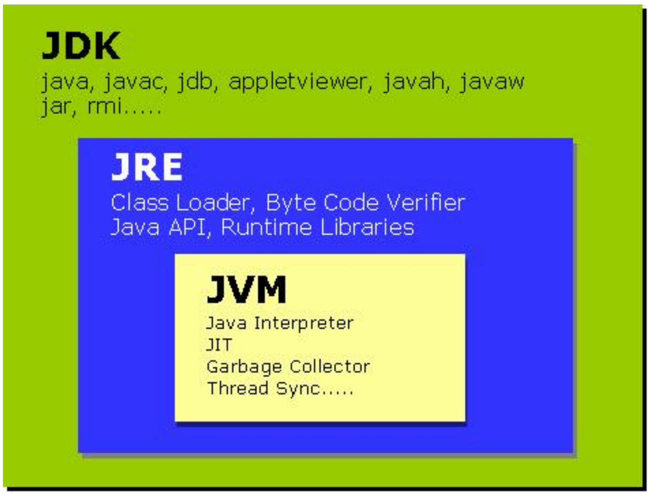
C++

BOTH PROCEDURAL AND OBJECT ORIENTED LANGUAGE
BJARNE STROUSTRUP IN 1979
COMPILER ONLY
PLATFORM DEPENDENT
FILE EXTENSION .CPP
FILE GENERATED .EXE FILE
POINTER SUPPORT

JAVA

OBJECT ORIENTED LANGUAGE
JAMES GOSLING IN 1995 MAY
BOTH COMPILER AND INTERPRETER
PLATFORM INDEPENDENT
FILE EXTENSION .JAVA
FILE GENERATED .CLASS FILE
POINTER NOT SUPPORT

9. JAVA ARCHITECTURE AND JDK ARCHITECTURE


JDK
java, javac, jdb, appletviewer, javah, javaw
jar, rmi.....
JRE
Class Loader, Byte Code Verifier
Java API, Runtime Libraries
JVM
Java Interpreter
JIT
Garbage Collector
Thread Sync.....

JDK :

- JDK STANDS FOR JAVA DEVELOPEMENT KIT.
- JDK CONSISTS OF JRE AND DEVELOPMENT TOOLS LIKE JAVA COMPILER, JAVA DOCUMENTS ETC…
- TO DEVELOPE JAVA APPLICATION OR EXECUTE JAVA PROGRAMS WE NEED JDK.

JRE :

- JRE STANDS FOR JAVA RUNTIME ENVIRONMENT .
- JRE CONSISTS OF JVM AND JAVA LIBRARIES LIKE CLASSES AND INTERFACE,THIS LIBRARY WILL PROVIDE TO JVM TO USE .
- JRE IS USED TO EXECUTE JAVA PROGRAMS .

JVM :

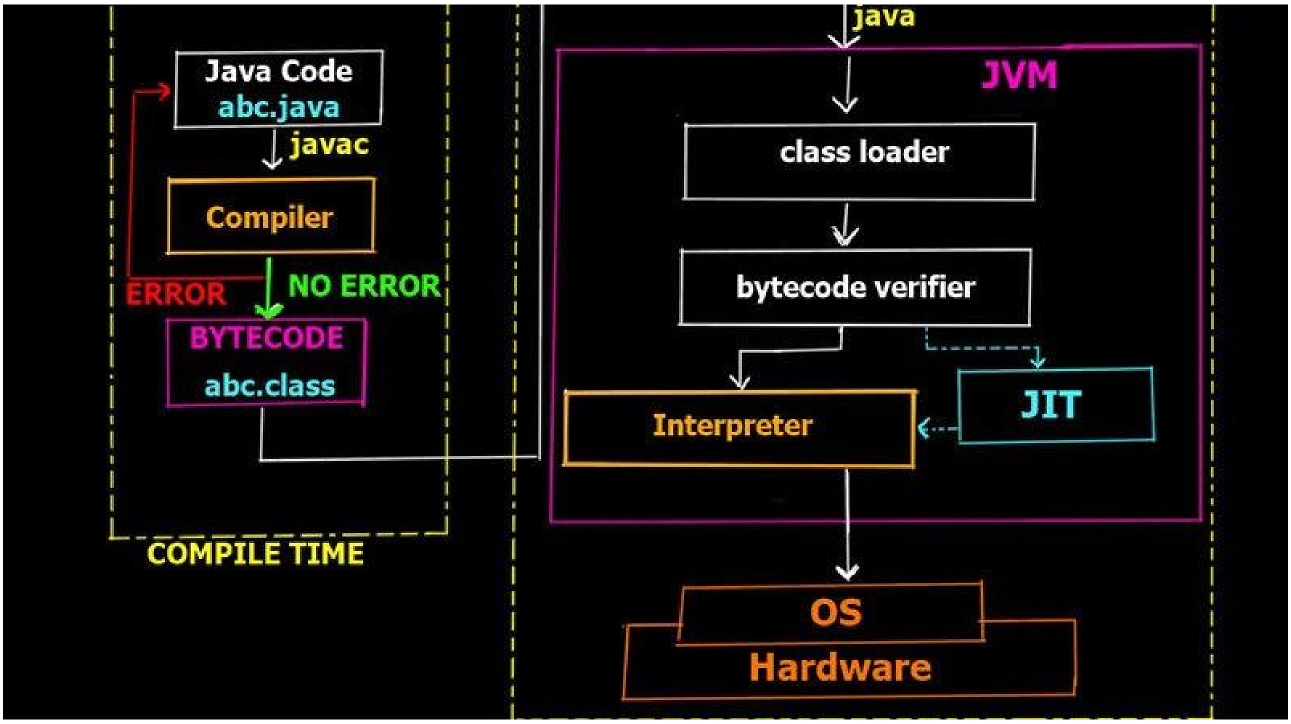
- JVM STANDS FOR JAVA VIRTUAL MACHINE .
- JVM READ THE INSTRUCTION LINE BY LINE UNDERSTAND AND EXEUTE IT .

- JVM CONSISTS OF JIT AND INTERPRETER . INTERPRETER HELPS TO CONVERT THE BYTECODE INTO MACHINE CODE .

JIT :

- JIT STANDS FOR JUST IN TIME
- IT WILL INCREASE THE PERFORMANCE. WHENEVER THE CODE IS REPEATED IT WILL CONVERT THE BYTE CODE INTO MACHINE CODE ONLY ONCE, JVM WILL USE THIS DIRECTLY.

10. JAVA EXECUTION FLOW

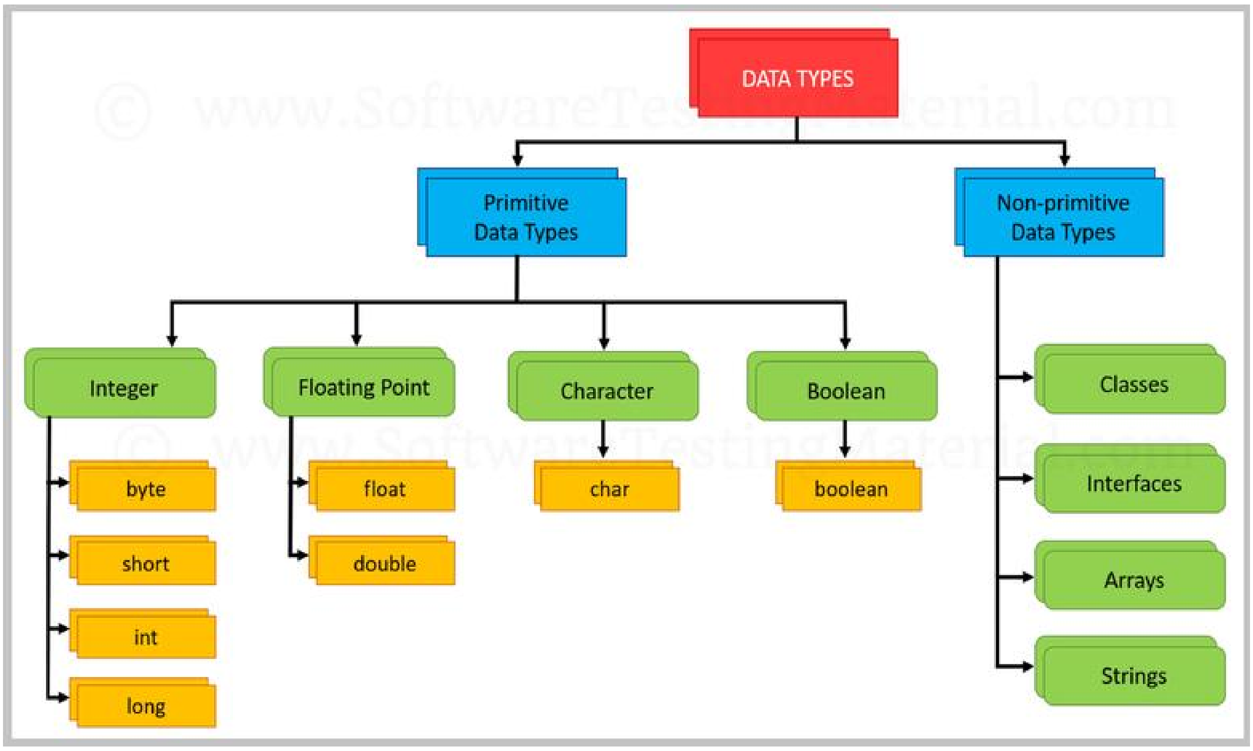


11. **WHAT IS DATA TYPE ?**

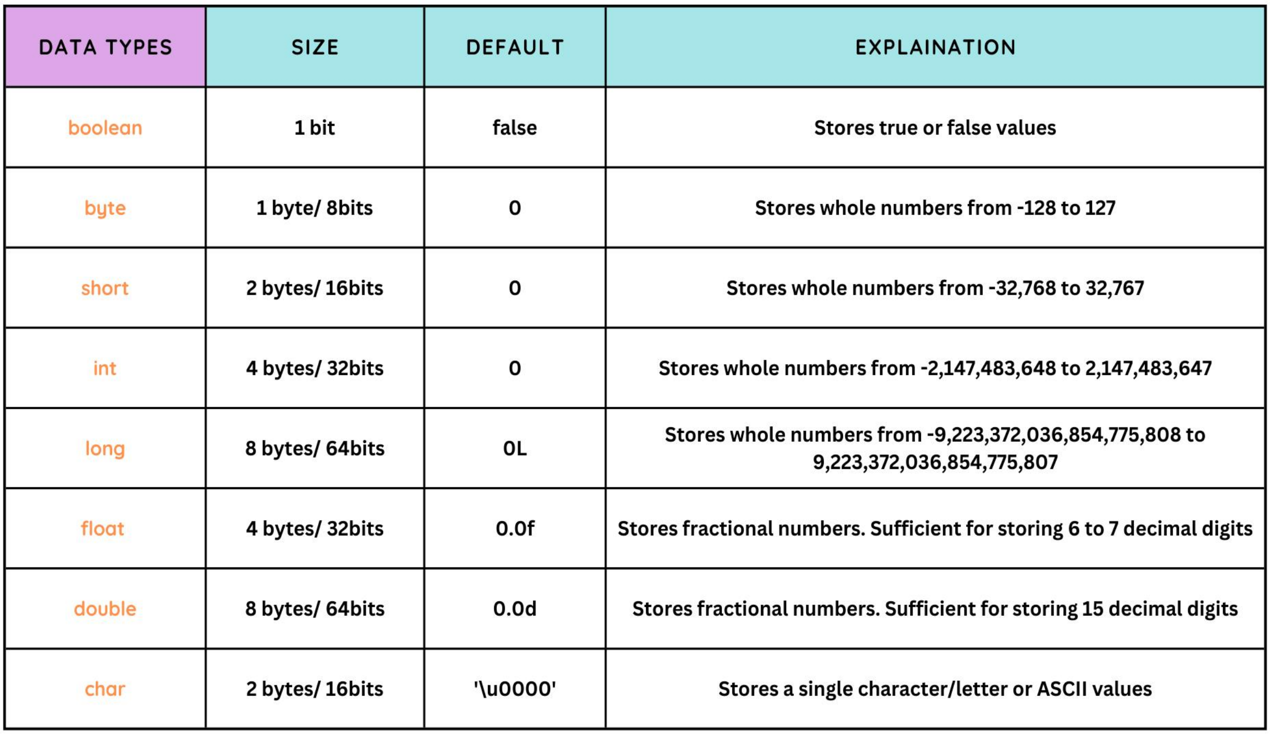
- DATA TYPE ARE USED TO SEPCIFY WHAT TYPE OF DATA WE CAN STORE .

THERE ARE TWO OF DATATYPES IN JAVA :

1. PRIMITIVE DATATYPE
2. NON-PRIMITIVE DATATYPE



© www.SoftwareTestingMaterial.com

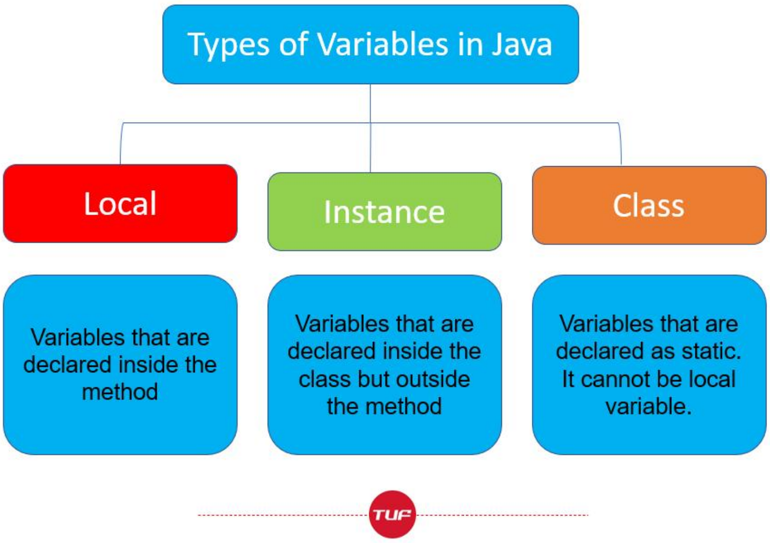


DATA TYPES	SIZE	DEFAULT	EXPLAINATION
boolean	1 bit	false	Stores true or false values
byte	1 byte/ 8bits	0	Stores whole numbers from -128 to 127
short	2 bytes/ 16bits	0	Stores whole numbers from -32,768 to 32,767
int	4 bytes/ 32bits	0	Stores whole numbers from -2,147,483,648 to 2,147,483,647
long	8 bytes/ 64bits	0L	Stores whole numbers from -9,223,372,036,854,775,808 to 9,223,372,036,854,775,807
float	4 bytes/ 32bits	0.0f	Stores fractional numbers. Sufficient for storing 6 to 7 decimal digits
double	8 bytes/ 64bits	0.0d	Stores fractional numbers. Sufficient for storing 15 decimal digits
char	2 bytes/ 16bits	'\u0000'	Stores a single character/letter or ASCII values

12. **WHAT IS VARIABLE ?**

- VARIABLE IS JUST A CONTAINER TO STORE THE DATA.

IN JAVA WE HAVE 3 TYPES


Types of Variables in Java
Local
Instance
Class
Variables that are declared inside the method
Variables that are declared inside the class but outside the method
Variables that are declared as static. It cannot be local variable.
TUF

1. **<u>LOCAL VARIABLE :</u>**

- A VARIABLE DECLARED INSIDE THE BODY OF THE METHOD IS CALLED LOCAL VARIABLE. YOU CAN USE THIS VARIABLE ONLY WITHIN THAT METHOD.
- IT CONSUMES LESS MEMORY AND LOCAL VARIABLE CANNOT BE STATIC .
- NAMES CAN BE REUSED FOR DIFFERENT METHODS .

2. **<u>STATIC VARIABLE :</u>**

- STATIC ARE SHARED RESOURCE FOR ALL THE INSTANCES OF THE CLASS.
- WE CAN USE ONLY IN CLASS AREA, WE CAN'T USE INSIDE THE METHOD AREA.
- IT WILL CREATE THE MEMORY AREA ONLY ONCE ON CLASS LOAD TIME.

3. **<u>INSTANCE VARIABLE :</u>**

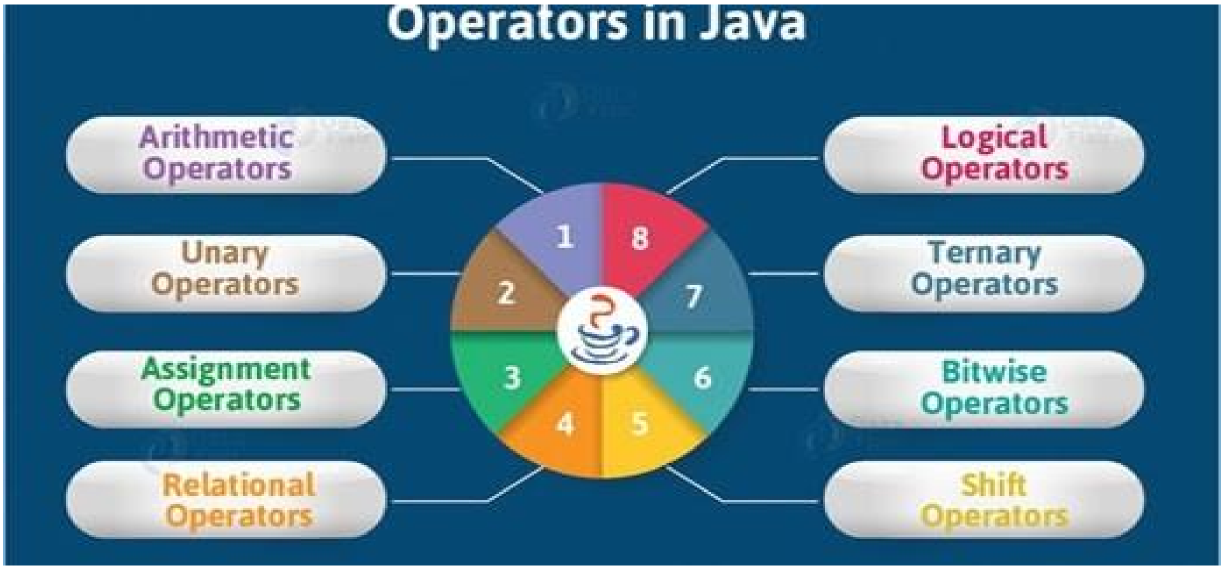
- EACH INSTANCE OF THE CLASS HAS ITS OWN COPY OF THE VARIABLES.
- WHEN THE OBJECT IS CREATED, MEMORY WILL CREATED IN HEAP AREA .

- WHEN THE OBJECT IS DESTROYED INSTANCE ARE DESTROYED IN HEAP AREA .

13. **WHAT IS OPERATORS ?**
    - OPERATORS ARE PRE - DEFINED SYMBOL WHICH IS USED TO PERFORM SOME SPECIFIC TASK .

WE HAVE UNARY,BINARY AND TERNARY OPERATOR.

1. ARITHMETIC OPERATOR
   (+ , - , * , /)
2. RELATIONAL OPERATOR
   (< , > , <= , >=)
3. COMPARISON OPERATOR
   (= , !=)
4. LOGICAL OPERATOR
   (AND(&&) , OR(||) , NOT(!))
5. ASSIGNMENT OPERATOR
   (+=   , -= , *= , /= , %=)
6. INC/DEC(POST/PRE) OPERATOR
   (VAR++ , ++VAR , VAR-- , --VAR)
7. TERNARY OR CONDITIONAL OPERATOR
   ((CONDITION) ? TRUE : FALSE)



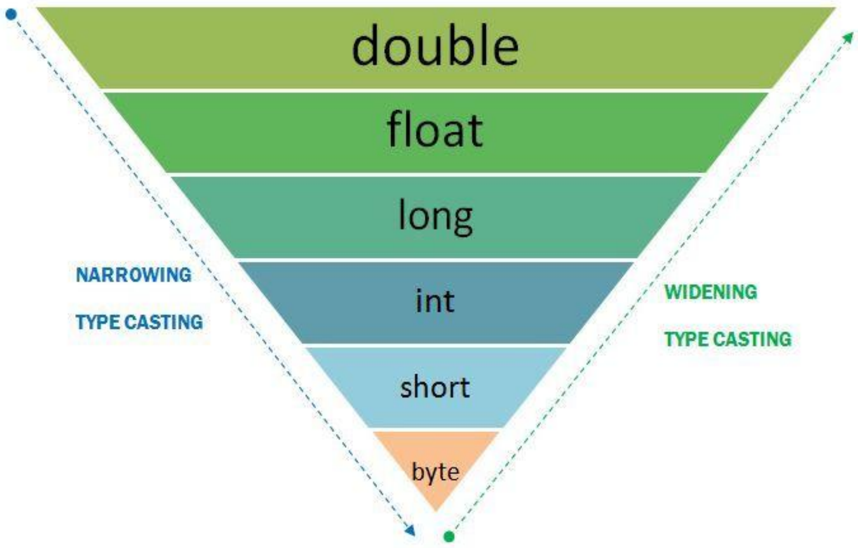
14. WHAT IS TYPE CASTING ?

• THE PROCESS OF CONVERTING ONE DATA TYPE INTO ANOTHER DATA TYPE IS KNOWN AS TYPE CASTING .

THERE ARE TWO TYPES OF TYPE CASTING IN JAVA :

1. **PRIMITIVE TYPE CASTING.**
2. **NON-PRIMITIVE TYPE CASTING.**

PRIMITIVE TYPE CASTING :



1. **WIDENING(IMPLICIT) :**

• THE PROCESS OF CONVERTING THE SMALLER RANGE DATATYPE INTO HIGHER RANGE DATA TYPE IS KNOWN AS WIDENING.

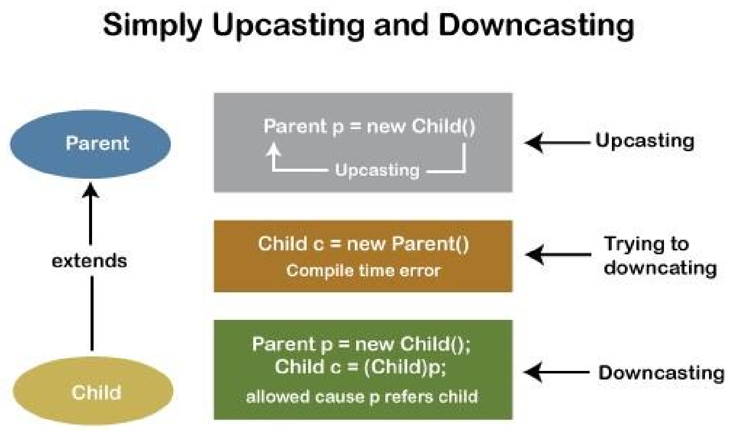
• HERE THERE IS NO DATA LOSS AND COMPILER CAN DO IMPLICIT TYPE CASTING.

2. **NARROWING(EXPLICIT) :**

• THE PROCESS OF CONVERTING THE HIGHER RANGE DATATYPE INTO LOWER RANGE DATA TYPE IS KNOWN AS NARROWING.

- HERE THERE IS POSSIBILITY OF DATA LOSS AND COMPILER CANNOT DO IMPLICIT TYPE CASTING INSTEAD OF THAT PROGRAMMER CAN DO EXPLICIT TYPE CASTING.

NON-PRIMITIVE TYPE CASTING :



1. UPCASTING :

- THE PROCESS OF STORING CHILD OBJECT REFERENCE IN PARENT TYPE OF CONTAINER IS CALLED AS UPCASTING.

2. DOWNCASTING :

- THE PROCESS OF STORING PARENT OBJECT REFERENCE IN CHILD TYPE OF CONTAINER IS CALLED AS DOWNCASTING.
- DOWN CASTING IS NOT POSSIBLE IMPLICIT INSTEAD OF THAT WE CAN DO IT EXPLICIT WITH THE HELP OF TYPE CASTING.

15. WHAT IS METHOD ?

- METHOD IS A BLOCK OF QUOTE OR LIST OF INSTRUCTION WHICH IS USED TO PERFORM SPECIFIC TASK.
- METHODS ARE USED TO REDUCE THE LINE OF CODE AND WE CAN ACHIEVE CODE REUSABILITY.
- WE CANNOT CREATE A METHOD INSIDE ANOTHER METHOD IN JAVA.

SYNTAX :

ACCESS MODIFIER	MODIFIER	RETURN TYPE	METHOD_NAME	([FORMAL ARGUMENTS])
PRIVATE	STATIC	VOID	METHOD_NAME +	
DEFAULT	ABSTRACT	PRIMITIVE D.T	FORMAL_ARGUMENT	
PROTECTED	SYNCHRONIZED	NON-PRIMITIVE D.T	=	
PUBLIC	FINAL		METHOD_SIGNATURE	

EXAMPLE :

```
PUBLIC STATIC VOID METHOD_NAME( ){
        //JAVA STATEMENTS…}
```

METHOD DECLARATION :

- IT CONSISTS OF ACCESS MODIFIER, MODIFIER, RETURN TYPE AND METHOD NAME.

ACCESS MODIFIER :

- IT IS USED TO CHANGE THE VISIBLITY OF THE METHOD. IT CONSISTS OF PRIVATE,DEFAULT, PROTECTED AND PUBLIC.

MODIFIER :

- IT IS USED TO CHANGE THE BEHAVIOUR OF THE METHOD. IT CONSISTS OF STATIC,TRANSIENT, VOLATILE, FINAL, ABSTRACT AND SYNCHRONIZED ETC…

RETURN TYPE :

- IT IS USED TO RETURN THE DATA TO THE CALLER METHOD . IT CONSISTS OF VOID, PRIMITIVE DATA TYPE AND NON - PRIMITIVE DATA TYPE .

METHOD SIGNATURE :

- IT CONSISTS OF METHOD NAME AND FORMAL ARGUMENTS.

TYPES OF MEHODS :

1. NON RETURN TYPE   WITH   NO ARGUMENT
2. NON RETURN TYPE   WITH   PARAMETERIZED
3. RETURN TYPE   WITH   NO ARGUMENT
4. RETURN TYPE   WITH   PARAMETERIZED

16. **WHAT IS CONTROL FLOW STATEMENT ?**

- IN JAVA WE CAN CONTROL THE EXECUTION OF THE PROGRAM BY USING TWO IMPORTANT STATEMENTS .

1. DECISION MAKING STATEMENT?

- IN JAVA WE USE DECISION MAKING STATEMENT TO DECIDE WHICH INSTRUCTION HAS TO BE EXECUTED AND WHICH INSTRUCTION NEED TO BE IGNORED .
- WE CAN ACHIEVE BY USING THIS STATEMENTS :

1. IF 2. IF ELSE 3. ELSE IF LADDER 4. SWITCH

2. LOOPING STATEMENT?

- LOOPING STATEMENT ARE USED TO REPEAT THE SET OF INSTRUCTIONS CONTINUOUSLY UNTIL THE CONDITION BECOME FALSE .
- WE CAN ACHIEVE BY USING THIS STATEMENTS :

1. WHILE 2. DO-WHILE 3. FOR 4. FOREACH

17. WHAT IS CONTROL TRANSFER STATEMENT ?

- **break** AND **return** KERYWORDS ARE CONTROL TRANSFER STATEMENT.

18. WHAT IS PACKAGE ?

- A SET OF CLASSES AND INTERFACES GROUPED TOGETHER IS KNOWN AS PACKAGE.
- IN JAVA WE HAVE : **BUILD-IN PACKAGE** AND **USER DEFINED PACKAGE.**

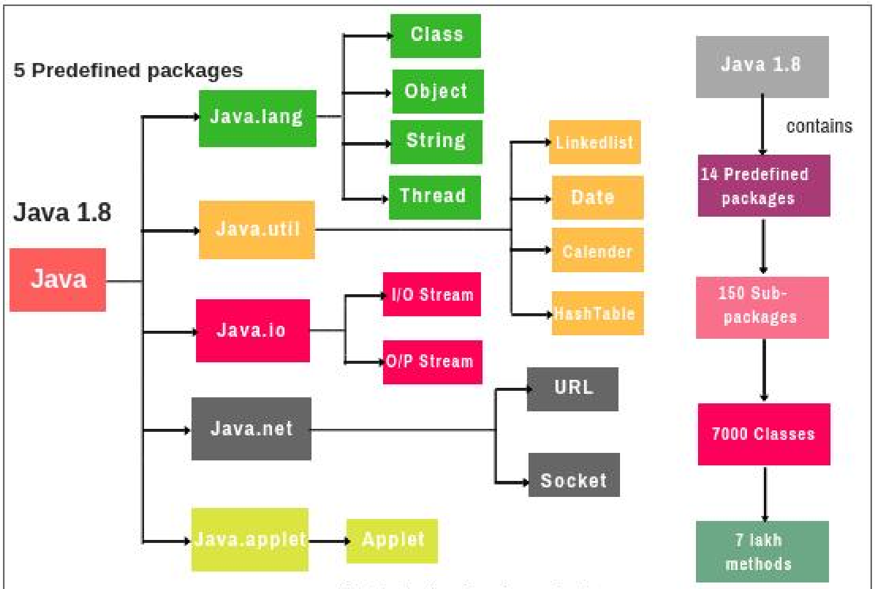


Fig: Predefined packages in Java

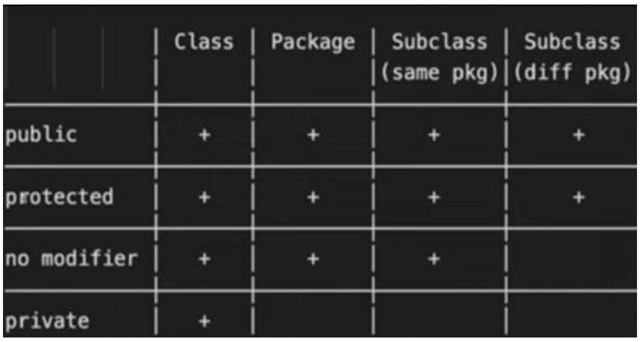
19. **WHAT IS ACCESS MODIFIER OR ACCESS SPECIFIER ?**

PRIVATE : IT IS USED TO ACCESS THE DATA WITHIN THE CLASS.

DEFAULT : IT IS USED TO ACCESS THE DATA WITHIN THE PACKAGE.

PROTECTED : IT IS USED TO ACCESS THE DATA WITHIN THE PACKAGE AND ALSO OUTSIDE THE PACKAGE BUT THE CLASS SHOULD BE CHILD .

PUBLIC : IT IS USED TO ACCESS THE DATA ANYWHERE.



	Class	Package	Subclass (same pkg)	Subclass (diff pkg)
public	+	+	+	+
protected	+	+	+	+
no modifier	+	+	+	
private	+			