Variables and Data Types

In any computer program, there is presence of the **variables** for either storing the input or output data required or generated in the memory during its execution. So, we can call **variable** as the name of a reserved area allocated in the memory.

In other words, it is a name of the memory location allocated by a particular operating system which can be called as the combination of "vary + able". This means its value can be changed when required.

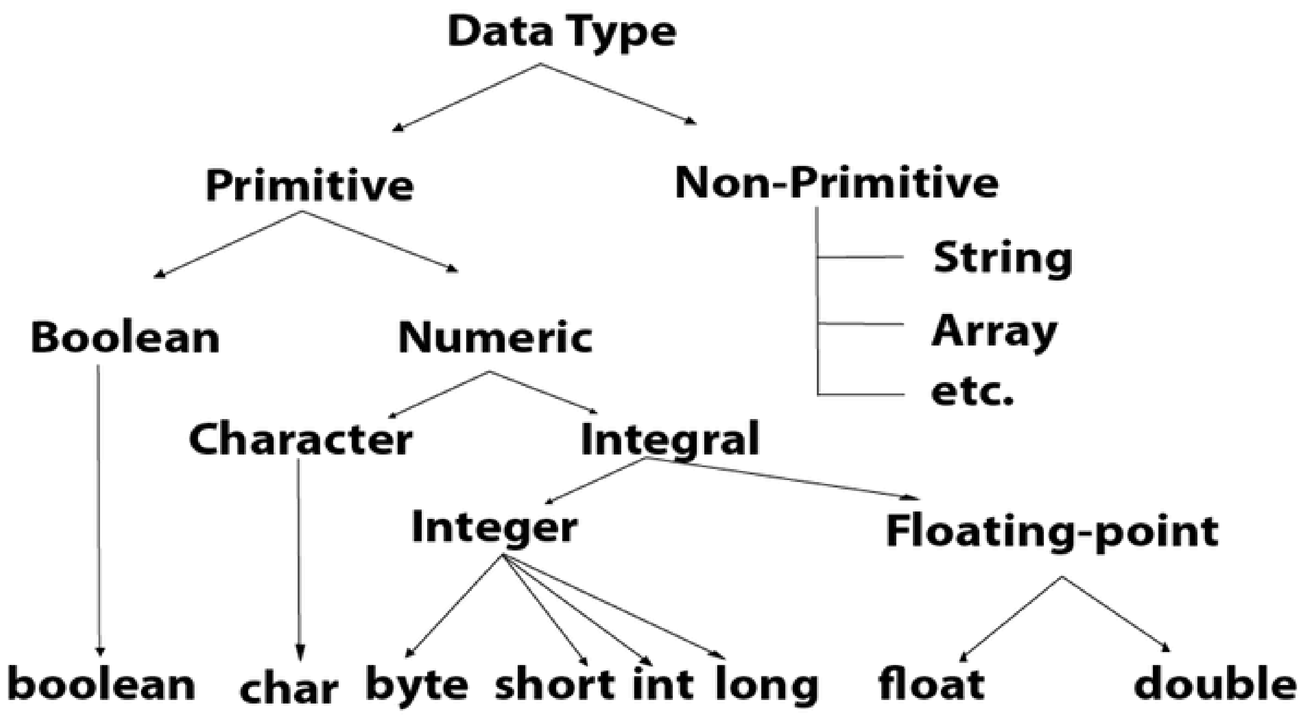
During the memory allocation to the variable, another important factor ***data type*** is taken into consideration which signifies what type of value has to be stored in that variable.

Thus based upon the data type, the operating system allocates the memory and decides what can be stored on to that space. In other words, we can store integers, decimals, characters, or other data in the variables by declaring their types.

Here in JAVA, variable is termed as a container which holds the value characterized via the data type while the developed program is executed.

While discussing about the data types, there are two types available in JAVA which are as;

1. **Primitive Data Types**
2. **Non-Primitive Data Types**



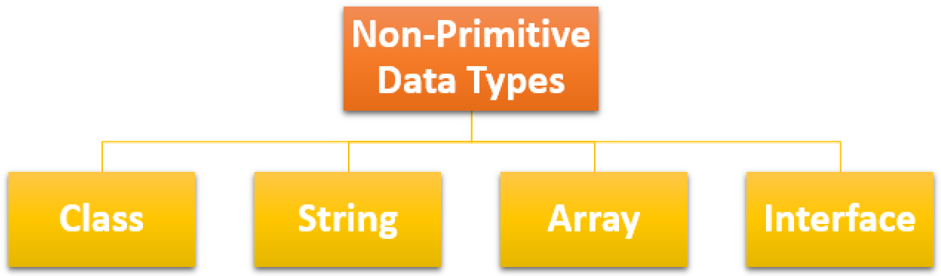
Primitive Data Types

It is the fundamental data type predefined by the programming language which is assigned to the variable for storing only that type of value in it. The optimum size and capacity to store the data of this data type is fixed. Some of the primitive data types include boolean, byte, short, int, long, float, double and char.

Type	Size (bits)	Minimum	Maximum	Example
byte	8	-2^{7}	$2^{7}– 1$	*byte b = 100;*
short	16	-2^{15}	$2^{15}– 1$	*short s = 30_000;*
int	32	-2^{31}	$2^{31}– 1$	*int i = 100_000_000;*
long	64	-2^{63}	$2^{63}– 1$	*long l = 100_000_000_000_000;*
float	32	-2^{-149}	$(2-2^{-23})\cdot2^{127}$	*float f = 1.456f;*
double	64	-2^{-1074}	$(2-2^{-52})\cdot2^{1023}$	*double f = 1.456789012345678;*
char	16	0	$2^{16}– 1$	*char c = ‘c';*
boolean	1	–	–	*boolean b = true;*

Non-Primitive Data Types

Similarly, non-primitive data types are those data types which are actually not defined by the programming language but are created by the programmers. So, they are also called as “reference variables or object references”. Along with this they cannot store the value of a variable directly in the memory instead they store a memory address of the variable which is referenced to hold the data. Variables created with these data types can be assigned with null and should always start with an upper-case letter.



Some of the non-primitive data types are presented in the figure above which are described as;

1. **Class**

A class is a user-defined data type from which objects are created. It describes the set of properties or methods that are common to all objects of the same type. A class contains fields and methods that represent the behavior of an object.

2. **String**

The String data type stores a sequence or array of characters. A string is a non-primitive data type but it is predefined in Java. String literals are enclosed in double-quotes.

Syntax

String <String_variable_name> = "<sequence_Of_Strings>" ;

3. **Arrays**

An Array is a single object that stores multiple variables of the same data type. The variables can be primitive or non-primitive data types.

Syntax

For declaring an array variable of primitive data type int:

int [ ] age;

4. **Interface**

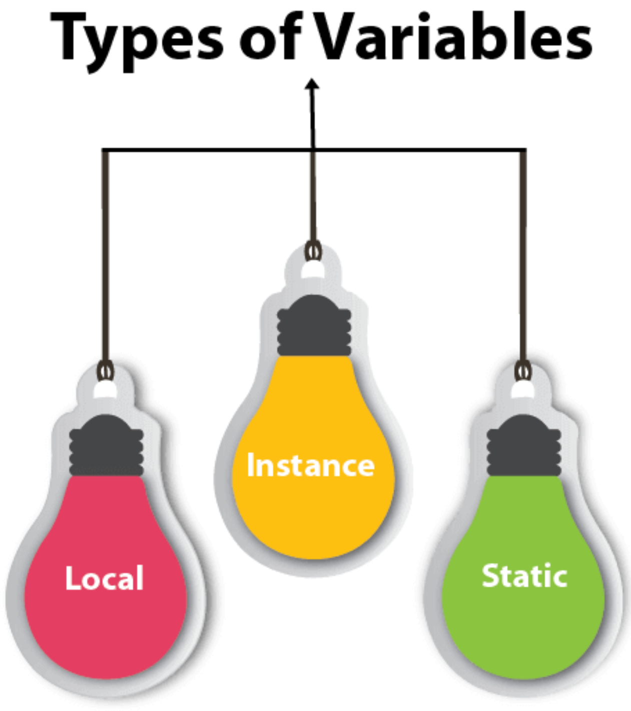
An interface is declared like a class. The key difference is that the interface contains methods that are abstract by default; they have no body.

Differences between Primitive and Non-Primitive Data Types:

- Primitive types are predefined (already defined) in Java. Non-primitive types are created by the programmer and is not defined by Java (except for String).
- Variables defined with non-primitive types can be used to call methods to perform certain operations, while primitive types cannot.
- A variable with primitive type has always a value, while non-primitive types can be null.

- A variable declared as primitive type starts with a lowercase letter, while non-primitive types starts with an uppercase letter.
- The size of the data having primitive type is fixed and depends on its capacity, while non-primitive types have all the same size.

Here, using either of the data types we can create variables and store data in it. In JAVA we can create variables in different ways based on their accessibility and usages. With reference to these cases, variables has been categorized into following types as;



a. Local Variable

Local variables are declared in methods, constructors, or blocks. Local variables are created when the method, constructor or block is entered, and the variable will be destroyed once it exits the method, constructor, or block. They must be initialized while declaring and can only be used within specific block scope inside which it has been declared.

b. Instance Variable

Similarly, a variable declared inside the class but outside the body of the method or any block scope is called an instance variable. It is not declared as static and initialization is not mandatory. Along with this it can also be created with the use of access specifiers, if not used then a default access specifier will be used.

It is called an instance variable because its value is instance-specific and is not shared among instances. This means when the new objects of a class are created and used to access the instance variables then each objects creates their own copy of instance variables rather than using the single one.

c. Static/Class Variable

Class variables also known as static variables are declared with the use of static keyword in a class, but outside a method, constructor or a block. There would only be one copy of each class variable per class, regardless of how many objects are created from it. It can be directly accessed without the use of any objects or instances and does not required initialization as well.

Demonstration of the types of the variables are as;

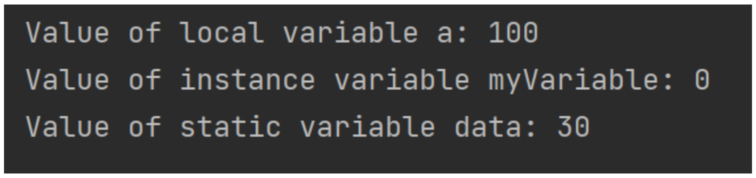
```java
class Demo
{
   static int m=100;//static variable
   void method()
   {
      int n=90;//local variable
   }
   public static void main(String args[])
   {
      int data=50;//instance variable
   }
}//end of class
```

Similarly, lets create a program to see the demo of the variable types in JAVA.

```java
class VariableExample{
   int myVariable;
   static int data = 30;

   void method(){
      int a = 100;
      System.out.println("Value of local variable a: "+a);
   }
   public static void main(String args[]){
      VariableExample obj = new VariableExample();
      obj.method();
      System.out.println("Value of instance variable myVariable: "+obj.myVariable);
      System.out.println("Value of static variable data: "+VariableExample.data);
   }
}
```

Output



```
Value of local variable a: 100
Value of instance variable myVariable: 0
Value of static variable data: 30
```