Lesson 3.1

Variable Types

By the end of this lesson you will learn:

- Different Types of Variables
- Identifying different types of variables
- Memory Allocation for instance and class variables
- Calculating memory size of an object

We need to use different variables to store data while coding. So, it is necessary to know about different types of variables, how they can be accessed and how memory is allocated for them. This memory allocation of variables ultimately leads to the size of memory of an object of a class. In this lesson we will be discussing about these things and as a bonus, I will be sharing a trick for output tracing so that your life gets a little bit easier.

Different Types of Variable

In Java, Variables can be divided into 3 types. They are:

- ✓ Local Variables
- ✓ Class Variables
- ✓ Instance Variables

Local Variables: Variables declared inside the scope of any method, constructor, loop, conditional statements are known as local variables. They do not have any existence outside the scope it is declared. The concept of default value is not applicable for local variables.

Class Variables: Variables declared inside the scope of a class but outside the scope of any method or constructor are known as Class Variables if and only if the keyword static is used while declaring the variable. It can be accessed from anywhere inside the class it is declared. It can also be accessed from outside of the class it is declared depending its access modifiers. We do not need any object to access it. It is accessed using the class name. All the objects of a class, share the same memory for a class variable.

Instance Variables: Variables declared inside the scope of a class but outside the scope of any method or constructor are known as Class Variables if and only if the keyword static is not used while declaring the variable. It can be accessed from anywhere inside the class it is declared. It can also be accessed from outside of the class it is declared depending its access modifiers. We have to use an object to access it. Each of the objects of a class, hold different memory for an instance variable.

The following code contains an example of different types of variables:

```
public class MyClass
{
    int m, n;
    static double o;
    public void changeValues(int a)
    {
        int x;
        for(int i=0; i<5; i++)
        {
            int y = i + a;
            if(y%2 ==0)
            {
                int z = a * y;
            }
        }
    }
}
```

Local variables declared in the example are: *int a, int x, int i, int y* and *int z*. These variables do not have any existence outside the block they are declared. Variables *x* and *a* are accessible/ have existence throughout the whole *method*. Variables *y* and *i* are accessible/have existence only within the *for* loop. The variable z is only accessible/exists inside the *if* block.

The variable *double o* is a class variable. Because it is declared outside the scope of a method and the *static* keyword is used while declaring. As, it is a class variable, we do not need object to access it. It can be accessed using the name of the class *MyClass*.

The variables *int m* and *int n* are instance variables. Because they are declared outside the scope of a method and the *static* keyword is not used while declaring. As, they are instance variables, we need to use object of *MyClass* to access it.

Identifying Different Types of Variables

Let us consider the following code to learn to identify different types of variables:

```
class Account
{
    private int accountNo;
    private double balance;
    public static double perDayTransactionLimit = 500;
    public Account( ){ }
    public Account(int an, double b)
    {
        accountNo = an;
        balance = b;
    }
    public void addInterest(double rate)
    {
        balance  = balance + (balance * rate / 100);
    }
    public void show( )
    {
        String msg = “Information”;
        System.out.println(msg);
        System.out.println(“AccountNo: ”+ accountNo);
        System.out.println(“Balance: ”+ balance);
    }
}
```

Step 1: First of all, list down all the local variables. As we have mentioned earlier that locals variables are declared inside the scope of ant method or any constructors, so we need to look inside each of the methods and constructors. There are two constructors in this class. One of them does not have any variables declared in it and the other takes two variables as arguments of the parameters. So, this *int an* and *double b* are local variables. Again there are two methods in this class. One of them takes a variable in its parameter and the other declares a variable in it. So, *double rate* and *String msg* are also local variables.

Step 2: Now, list down the class variables. As we have crossed off all the local variables, class variables and instance variables are remaining. Class variables has the keyword *static* in its declaration. In the above code, *double perDayTransactionLimit* is the class variable.

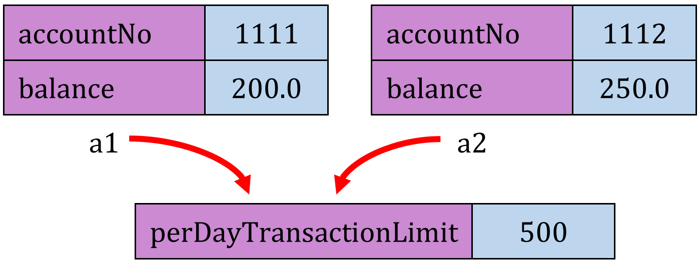
Step 3: Lastly, only the instance variables are remaining. In the above code, *int accountNo* and *double balance* are instance variables.

Memory Allocation for Instance and Class Variables

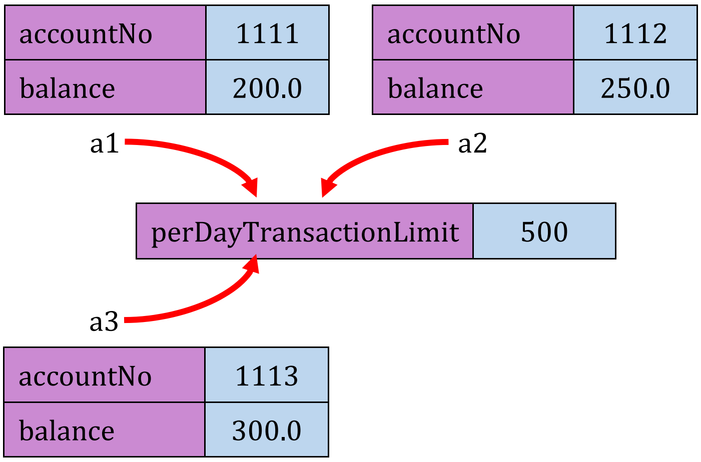
As we stated earlier, all the objects of a class, share the same memory for a class variable and each of the objects of a class, hold different memory for an instance variable. So, if we create two objects of Account class the variables accountNumber and balance will have two different memory locations for two different objects and the memory location for the variable perDayTransactionLimit will be shared by both of the objects.

Let’s assume that we are creating two objects of this account class:
Account a1 = new Account(1111, 200.0);
Account a2 = new Account(1112, 250.0);

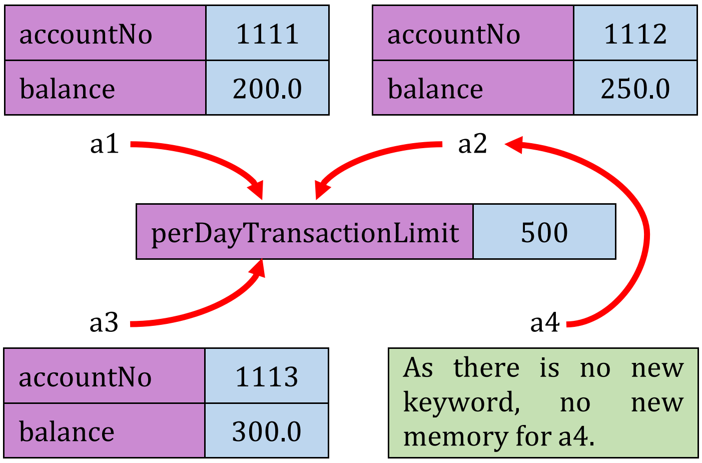
The memory representation will be:



Now, if we create another object:
Account a3 = new Account(1113, 300.0);



Again, if we write:
Account a4 = a2;



As there is no new keyword, no new memory for a4.

Calculating Memory Size of an Object

Previously, in lesson 2.1 we learned the memory size for the variables of the primitive datatypes. In lesson 2.2 we learned that objects are simply a variable of a non-primitive datatype. Now, we will learn to calculate the memory size of a variable of non-primitive datatype, hence an object.

The memory occupied by an object of a class is the summation of the memories occupied by the instance variables of that class and if the class has any class variables, all the objects of that class will share an additional memory of the total memories occupied by the class variables.

Let us assume that we have a class named *Book.* The *Book* class has four instance variables. *String title, String authorName, double price* and *int publicationYear.* So, there are two Strings, one double and one integer. The double variable occupies 8Bytes of memory and the integer variable occupies 4Bytes of memory. But the memory occupied by the String variables depends on the length of their values. So, if the maximum length for title is 30 and the maximum length for authorName is 50, the maximum memory occupied by title will be (30 x 2Bytes) 60Bytes and the maximum memory occupied by authorName will be (50 x 2Bytes) 100Bytes. So, the maximum memory occupied by an object of the Book class will be *(60+100+8+4)* 172Bytes.

Now, let us assume the *Account* class used above. The *Account* class has two instance variables. One is *int accountNo* and the other is *double balance.* The double variable occupies 8Bytes of memory and the integer variable occupies 4Bytes of memory. So, an object of *Account* class will occupy 12Bytes of memory. Also, as there is also a class variable *double perDayTransactionLimit*, so all the objects of *Account* class will share an additional 8Bytes of memory.

Practice

- ✓ Identifying different types of variables.
- ✓ Calculating memory size of an object.