Java Variables

|  |
| --- |
| In Java, Variables are the data containers that save the data values during Java program execution. Every Variable in Java is assigned a data type that designates the type and quantity of value it can hold. A variable is a memory location name for the data.  **Variables in Java**  Java variable is a name given to a memory location. It is the basic unit of storage in a program.  The value stored in a variable can be changed during program execution.  Variables in Java are only a name given to a memory location. All the operations done on the variable affect that memory location.  In Java, all variables must be declared before use. |

How to Declare Variables in Java?

|  |
| --- |
| We can declare variables in Java as pictorially depicted below as a visual aid. |
| 1. datatype: Type of data that can be stored in this variable. 2. data\_name: Name was given to the variable. |

|  |
| --- |
| In this way, a name can only be given to a memory location. It can be assigned values in two ways:   * Variable Initialization * Assigning value by taking input |
| How to Initialize Variables in Java?   * **datatype:** Type of data that can be stored in this variable. * **variable\_name:** Name given to the variable. * **value:** It is the initial value stored in the variable. |
|  |
| // Declaring float variable float simpleInterest;  // Declaring and initializing integer variable int time = 10, speed = 20;  // Declaring and initializing character variable char var = 'h'; |
| Types of Variables in Java   1. Local Variables 2. Instance Variables 3. Static Variables |
|  |
| 1. Local Variables   1. A variable defined within a **block or method or constructor** is called a local variable. 2. These variables are created when the block is entered, or the function is called and destroyed after exiting from the block or when the call returns from the function. 3. The scope of these variables exists only within the **block** in which the variables are declared, i.e., we can access these variables only within that block. 4. **Initialization of the local variable is mandatory before using it in the defined scope.** |
| // Java Program to implement  // Local Variables  import java.io.\*;  class GFG {  public static void main(String[] args)  {  **// Declared a Local Variable**  int var = 10;  // This variable is local to this main method only  System.out.println("Local Variable: " + var);  }  } |
| Local Variable: 10 |

|  |
| --- |
| **2. Instance Variables**   1. As instance variables are declared in a **class**, these variables are created when an object of the class is **created and destroyed** when the object is destroyed. 2. **Unlike local variables**, we may use access specifiers **for instance variables**. If we do not specify any access specifier, then the **default access specifier** will be used. 3. Initialization of an instance **variable is not mandatory**. Its default value is dependent on the data type of variable. **For String it is null, for float it is 0.0f, for int it is 0, for Wrapper classes like Integer it is null, etc.** 4. **Instance variables** can be accessed only by **creating objects**. 5. We initialize instance variables using [constructors](https://www.geeksforgeeks.org/constructors-in-java/) while creating an object. We can also use [instance blocks](https://www.geeksforgeeks.org/using-instance-blocks-in-java/) to initialize the instance variables. |
| class InstanceVariable {  // This is an instance variable  int num;  public static void main(String[] args) {  // Create an object of the InstanceVariable class  InstanceVariable obj = new InstanceVariable();  // Access and modify the instance variable 'num' using the object  obj.num = 10; // Assign a value to the instance variable  System.out.println("Value of num: " + obj.num); // Print the value of 'num'  }  } |

|  |
| --- |
| **3. Static Variables**  Static variables are also known as class variables.   1. These variables are declared similarly to instance variables. The difference is that static variables are declared using the static keyword within a class outside of any method, constructor, or block. 2. Unlike instance variables, we can only have one copy of a static variable per class, irrespective of how many objects we create. 3. Static variables are created at the start of program execution and destroyed automatically when execution ends. 4. Initialization of a static variable is not mandatory. Its default value is dependent on the data type of variable. For String it is null, for float it is 0.0f, for int it is 0, for Wrapper classes like Integer it is null, etc. 5. If we access a static variable like an instance variable (through an object), the compiler will show a warning message, which won’t halt the program. The compiler will replace the object name with the class name automatically. 6. If we access a static variable without the class name, the compiler will automatically append the class name. But for accessing the static variable of a different class, we must mention the class name as 2 different classes might have a static variable with the same name. 7. Static variables cannot be declared locally inside an instance method. 8. [Static blocks](https://www.geeksforgeeks.org/static-blocks-in-java/) can be used to initialize static variables. |
|  |

|  |
| --- |
| **Differences Between the Instance Variables and the Static Variables**   * Each **object** will have its own copy of an **instance variable**, whereas we can only have one copy of a **static variable** **per class**, irrespective of how many objects we create. Thus, **static variables are good for memory management.** * Changes made in an instance variable using one object will not be reflected in other objects as each object has its own copy of the instance variable. In the case of a static variable, changes will be reflected in other objects as static variables are common to all objects of a class. * We can access instance variables through object references, and static variables can be accessed directly using the class name. * Instance variables are created when an object is created with the use of the keyword ‘new’ and destroyed when the object is destroyed. Static variables are created when the program starts and destroyed when the program stops. |
|  |