VARIABLES IN JAVA

www.ravindrababuravula.com

What is a Variable?

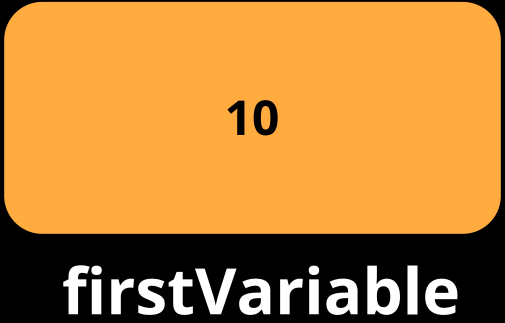
- A variable can be thought of as a container which holds value for you during the life of your program.
- Every variable is assigned a data type which designates the type and quantity of a value it can hold.

Syntax:

```
variable_name  = value;
```

What is Variable?

Physical Representation of a variable



Variable name: firstVariable
Value: 10
Data type: int

Rules to declare a Variable

- Variable names are case-sensitive. Every variable name should start with either alphabets or underscore (_) or dollar ($) symbol.
- Additionally, the dollar sign character, by convention, is never used at all. You may find some situations where auto-generated names will contain the dollar sign, but your variable names should always avoid using it.

Rules to declare a Variable

- ➢ A similar convention exists for the underscore character; while it's technically legal to begin your variable's name with "_", this practice is discouraged.
- ➢ No space are allowed in the variable declarations.
- ➢ Except underscore (_) no special symbol are allowed in the middle of variable declaration.

Rules to declare a Variable

- Variable name must not be a keyword or reserved word.
- Capitalization of First Character of Second Word: If variable name contain two words then write first letter of second word in Capital Case. If variable name contain single word then write that word in small case.

Java Variable Declaration

➢ In Java we declare a variable like this:

```
datatype name;
```

Example: Declaring a variable named myVariable of type int.

```
int myVariable;
int myVariable1, myVariable2, ...., myVariableN;
```

www.ravindrababuravula.com

Java Variable Assignment

- Assigning a value to a variable in Java follows this pattern:

```
variableName = value ;
```

- Here are three examples which assign values to three different variables with different data types

```
 myByte   = 127;
myFloat  = 199.99;
myString = "This is a text";
```

Scope and Lifetime of a Variables

- The scope of a variable defines the section of the code in which the variable is visible. As a general rule, variables that are defined within a block are not accessible outside that block.
- The lifetime of a variable refers to how long the variable exists before it is destroyed. Destroying variables refers to deallocating the memory that was allotted to the variables when declaring it.

Types of Variable in Java

- ➢ Java Programming language defines mainly three kind of variables.

1. Local Variable
2. Instance Variable
3. Static Variable

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.
- Access modifiers cannot be used for local variables

Local Variable

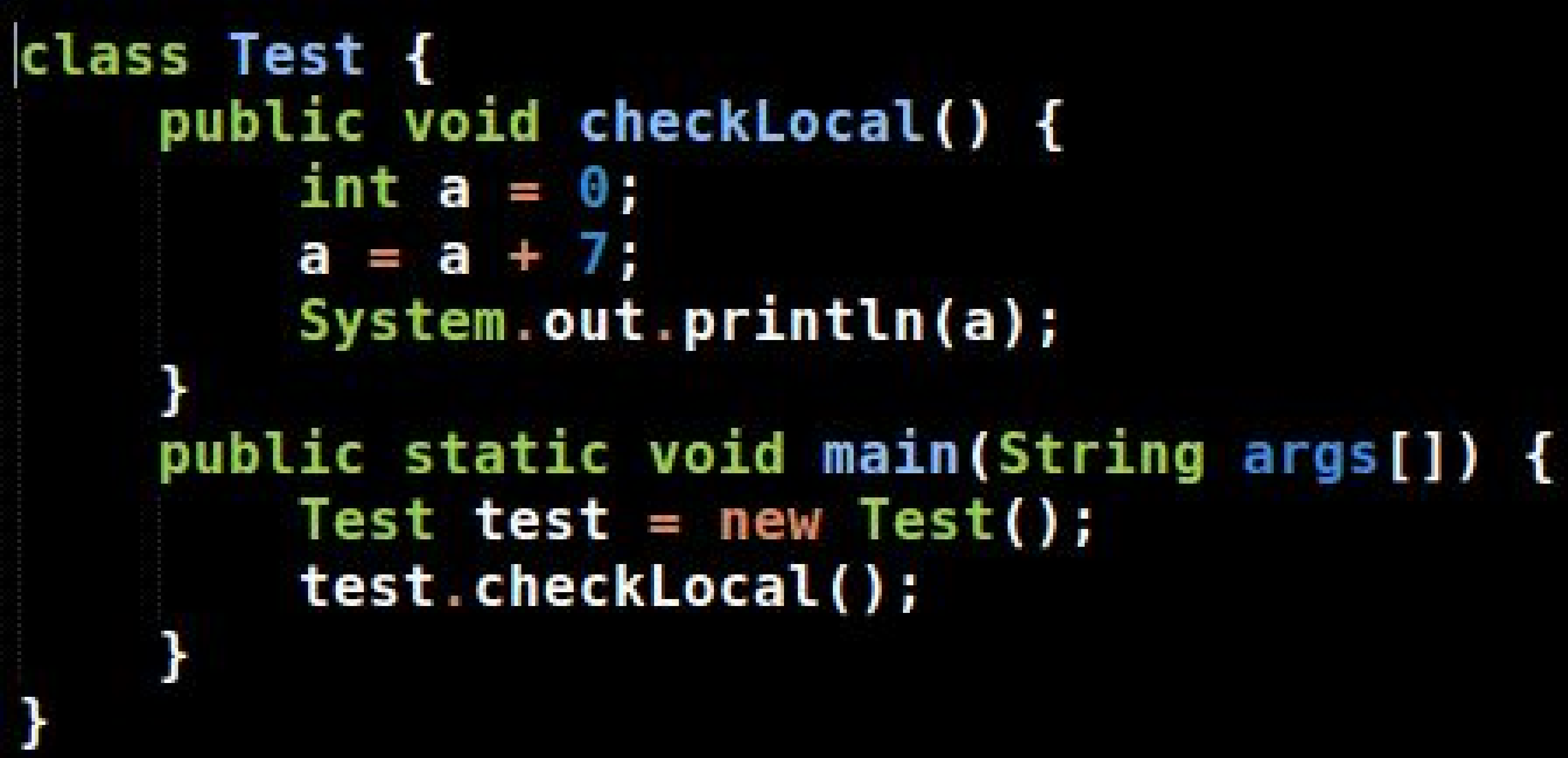
- Local variables are visible only within the declared method, constructor or block.
- Local variables are implemented at stack level internally.
- There is no default value for local variables so local variables should be declared and an initial value should be assigned before the first use

Syntax

```
class {
    constructor {
        // Use Local Variables within Constructor
        }
    method {
        // Use Local Variables within Method
        block {
            // Use Local Variables within Block
         }
      }
 }
```

www.ravindrababuravula.com

Example1



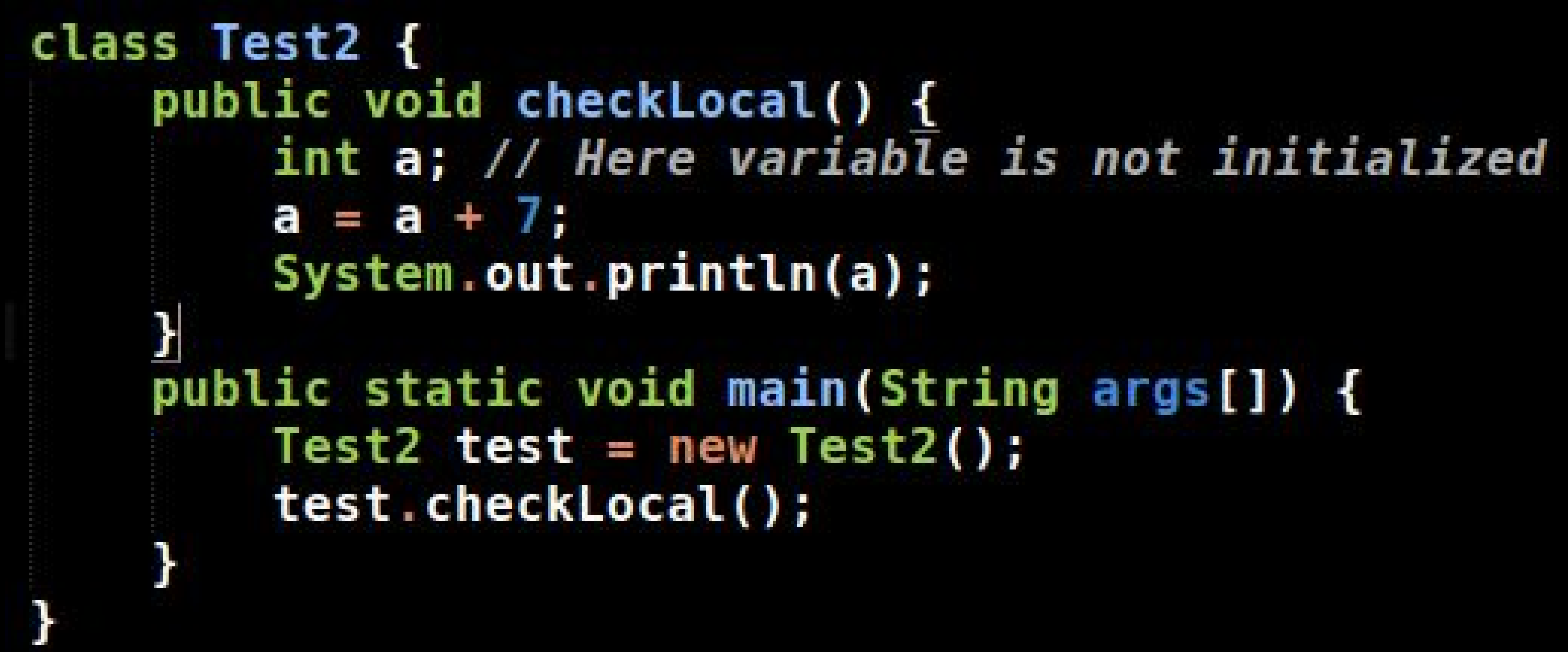
```
class Test {
    public void checkLocal() {
        int a = 0;
        a = a + 7;
        System.out.println(a);
    }
    public static void main(String args[]) {
        Test test = new Test();
        test.checkLocal();
    }
}
```

Example

In the given example, a is a local variable. This is defined inside checkLocal() method and its scope is limited to only this method.

Output: 7

Example2

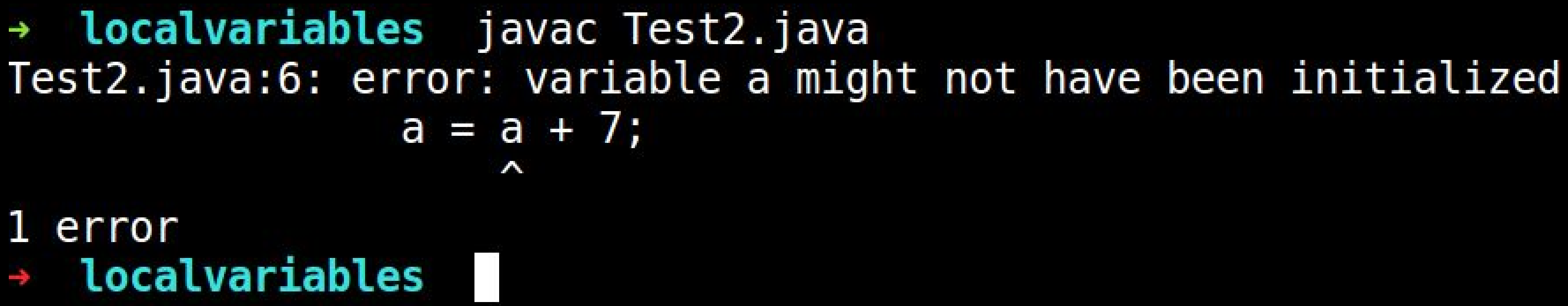


```
class Test2 {
    public void checkLocal() {
        int a; // Here variable is not initialized
        a = a + 7;
        System.out.println(a);
    }
    public static void main(String args[]) {
        Test2 test = new Test2();
        test.checkLocal();
    }
}
```

Example2

In this example uses a without initializing it, so it would give an error at the time of compilation.

Output:



```
→  localvariables  javac Test2.java
Test2.java:6: error: variable a might not have been initialized
                a = a + 7;
                    ^
1 error
→  localvariables  
```

Instance Variable

- ➢ Instance variables are declared in a class, but outside a method, constructor or any block.
- ➢ When a space is allocated for an object in the heap a slot for each instance variable value is created.
- ➢ Instance variables are created when an object is created with the use of the keyword 'new' and destroyed when the object is destroyed.

Instance Variable

- Instance variables hold values that can be referenced by more than one method, constructor or block.
- Instance variables can be declared in body of a class before or after use.
- Access modifiers can be given for instance variables.

Instance Variable

- The instance variables are visible for all methods, constructors and block in the class. Normally it is recommended to make these variables private (access level).
- However visibility for subclasses can be given for these variables with the use of access modifiers.
- Instance variables have default values. For integers the default value is 0, for Booleans it is false and for object references it is null.

Instance Variable

- Values can be assigned during the declaration or within the constructor
- Instance variables can be accessed directly by calling the variable name inside the class. However within static methods and different classes (when instance variables are given accessibility) these should be called using the fully qualified name .
- ObjectReference.VariableName

www.ravindrababuravula.com

Static Variable

- Static variables are declared with the 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

Static Variable

- Static variables are stored in static memory. It is rare to use static variables other than declared final and used as either public or private constants

Static Variable

- Static variables are created when the class containing static variables is loaded and destroyed when the program stops
- Visibility is similar to instance variables. However, most static variables are declared public since they must be available for users of the other classes.
- Static variables can be accessed by calling with the class name .

  ClassName.VariableName

Static Variable

- ➢ Default values are same as instance variables. For integers the default value is 0, for Booleans it is false and for object references it is null.

- ➢ Values can be assigned during the declaration or within the constructor. Additionally values can be assigned in special static initializer blocks

Java is Strongly Typed Language

- Java is a strongly typed programming language because every variable must be declared with a data type.
- A variable cannot start off life without knowing the range of values it can hold, and once it is declared, the datatype of the variable cannot change.

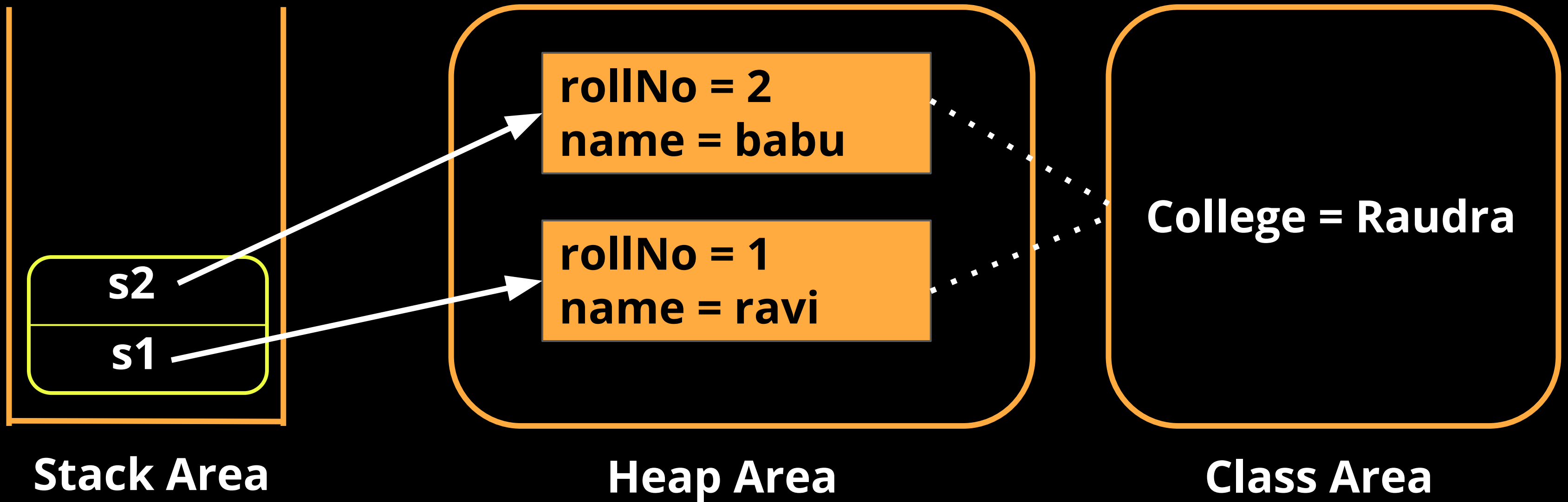
Java is Strongly Typed Language

- Example: The following declaration is allowed because the variable has "hasDataType" is declared to be a boolean data type:

```
boolean hasDataType;
```

- For the rest of its life, hasDataType can only ever have a value of true or false.

Example


rollNo = 2
name = babu
rollNo = 1
name = ravi
College = Raudra
s2
s1
Stack Area
Heap Area
Class Area