Non-Primitive Data Types

Non-primitive data types are called reference types because they refer to objects.

The main difference between primitive and non-primitive data types are:

- Primitive types are predefined (already defined) in Java. Non-primitive types are created by the programmer and are not defined by Java (except for String).
- Non-primitive types can be used to call methods to perform certain operations, while primitive types cannot.
- A primitive type has always a value, while non-primitive types can be null.
- A primitive type starts with a lowercase letter, while non-primitive types start with an uppercase letter.
- The size of a primitive type depends on the data type, while non-primitive types all have the same size.

Examples of non-primitive types are Strings,Arrays,class,interface, etc. You will learn more about these in a later chapter**.**

Declaration: Declaration is when you declare a variable with a name, and a variable can be declared only once.

Example: int x;

String myName;

boolean myCondition;

Initialization: Initialization is when we put a value in a variable, this happens while we declare a variable.

Example: int x = 7;

String myName = "hi";

boolean myCondition = false;

Assignment: Assignment is when we already declared or initialized a variable, and we are changing the value. You can change the value of the variable as many times you want or you need.

Example:

int x = 7;

x = 12; .....We just changed the value.

String myName = "hi";

myName = "bye" .....We just changed the value.

Boolean myCondition = false;

myCondition = true; .....We just changed the value.

Note: In memory will be saved the last value that we put.

Note:

Declaration is not to declare "value" to a variable; it's to declare the *type* of the variable.

Initialization is the assignment of a value to a variable *at the time of declaration* or for the first time.

Assignment is simply the storing of a value to a variable for the second time.

Java Identifiers:

All Java variables must be identified with unique names.

These unique names are called identifiers.

Identifiers can be short names (like x and y) or more descriptive names (age, sum, totalVolume).

Note: It is recommended to use descriptive names in order to create understandable and maintainable code:

Example:

```
public class Main {
  public static void main(String[] args) {
    // Good
    int minutesPerHour = 60;

    // OK, but not so easy to understand what m actually is
    int m = 60;

    System.out.println(minutesPerHour);
    System.out.println(m);
  }
}
```

In programming languages, identifiers are used for identification purposes. In Java, an identifier can be a class name, method name, variable name, or label. For example :

```
public class Test
{
    public static void main(String[] args)
    {
        int a = 20;
    }
}
```

In the above java code, we have 5 identifiers namely :

- **Test** : class name.
- **main** : method name.
- **String** : predefined class name.
- **args** : variable name.
- **a** : variable name.

Rules for defining Java Identifiers :There are certain rules for defining a valid java identifier. These rules must be followed, otherwise we get compile-time errors. These rules are also valid for other languages like C,C++.

- The only allowed characters for identifiers are all alphanumeric characters([**A-Z**],[**a-z**],[**0-9**]), '**$**'(dollar sign) and '**_**' (underscore).For example "ashokIt@" is not a valid java identifier as it contain '@' special character.
- Identifiers should **not** start with digits(**[0-9]**). For example "123geeks" is not a valid java identifier.
- Java identifiers are **case-sensitive**.
- There is no limit on the length of the identifier but it is advisable to use an optimum length of 4 – 15 letters only.
- **Reserved Words** can't be used as an identifier. For example "int while = 20;" is an invalid statement as while is a reserved word. There are **53** reserved words in Java.

Examples of valid identifiers :

MyVariable

MYVARIABLE

myvariable

x

i

x1

i1

_myvariable

$myvariable

sum_of_array

ashokIT123

Examples of invalid identifiers :

My Variable // contains a space

123geeks // Begins with a digit

a+c // plus sign is not an alphanumeric character

variable-2 // hyphen is not an alphanumeric character

sum_&_difference // ampersand is not an alphanumeric character

Reserved Words:

In Java, there are keywords that are reserved for the use of Java functions or other uses that cannot be identifiers like variables, classes and function names. When a reserved word is used as a variable, we will get an error or some other unexpected result.

Please notice that the Java reserved word must be written in the exact same way as Java states, including the case of the word.

The following are more examples of reserved words like, **throw, boolean, else, import, public, throws, break, return, byte, extends, int, short, true, false, case, interface, static, try, catch, final, long, void.**