String , StringBuffer & StringBuilder

Variables Scope, For each loop,

Arrays, Var-args,

Exception Handling.

String is a special data type, String is a built-in class defined in java.lang package.

Java.lang == the default package in JAVA. (We no need to import this package, JVM will automatically import this package)

Java API Documentation -- <https://docs.oracle.com/javase/8/docs/api/>

Generally package is a container which contains many classes, interfaces, enums, exceptions, annotations & errors.

Package name should be in lower case only.

Package naming convention == reverse of the company url.

<http://Google.com> package name com.google

com.google.project1.module1

com.google.project1.module2

com.google.project2.module == Com (folder) – google (sub folder) -project2(sub-folder)

src = default input folder (Source folder) All the .java files will be stored here

bin = default output folder (Output folder) All the .class files will be stored here

java.lang package will have all the Wrapper classes (8 Wrapper classes)

|  |  |  |
| --- | --- | --- |
| **Sl.No** | **Primitive Data type** | **Wrapper Class** |
| 1 | boolean | Boolean |
| 2 | byte | Byte |
| 3 | char | Character |
| 4 | short | Short |
| 5 | int | Integer |
| 6 | float | Float |
| 7 | double | Double |
| 8 | long | Long |

It also contains importance classes like

* System
* Package
* Object
* Thread
* Throwable
* String
* StringBuffer
* StringBuilder
* Process
* Runtime

String is a special class & Special Data type in JAVA.

Types of Classes

* Built-in Class (Pre-defined /System-defined/ developer defined) & Custom class (User defined class)
* Simple concrete class / abstract or in-complete or non-concrete class
* Bean Class/ Entity Bean Class/ Wrapper Class /POJO class
* DAO (Data Access Object)/ DTO (Data Transfer Objects) / Controller/ Service Class
* Thread/ Servlet Class
* Base/Parent/Super Class & Sub/Child/Derived Class

String is a group of characters in sequence.

String is a built-in class defined in java.lang package.

When we use String, we are creating objects using String class.

String objects will be saved In a different location of JVM (String Pool)

Strings are immutable in nature. (It can’t be modified after creation)

If we try to change/alter/update it, then a new copy will be created in the memory.

StringBuffer & StringBuilder (Mutable String)

+ is a concatenation operator (while using a string as one of the operand)

Equals method will check the hashcode values & == will check hashcode & type.

StringBuffer (Is a Thread-Safe implementation) – So we can use it in multi-threading environment ( All the methods are synchronized)

StringBuilder (Is not Thread Safe) – It’s not safe to use in multi-threaded environment.

Non-access Modifier (Behavior modifiers)

* Abstract
* Final
* Static (for methods/ variables/ block of code)
* Transient / Volatile

Arrays = derived data type

Class = derived data type

<class\_name> <ref\_variable\_name> = new Constructor(); // derived data type declaration & initialization

<data\_type> <variable\_name> [= initial\_value]; // primitive data type declaration & initialization

Int a[] = new int[5];

Varargs = variable arguments [ 0 or more arguments can be passed instead of varargs]

Varargs uses … {triple dot}

For each loop --- Enhanced for loop

Variable Scope

1. Instance variable = depending on access modifier (Accessed using object reference)
2. Local variable = accessed within the block only [ variable declared inside methods or blocks, variable passed as an arguments to the methods]
3. Static variable = depending on access modifier (Accessed using class name)

Exception Handling in JAVA.

Exception – Is a Road block while running the code.

Exception Hierarchy

Throwable (Class) java.lang [ Root of Exception Hierarchy] - Super class for Error & Exception

Exception – Un-avoidable condition which can be caught by proper codes.

Types of Exception WRT the developers

1. Built-in Exceptions or System-defined or pre-defined Exception
2. Custom Exception or User-defined Exception

Types of Exception WRT the behavior

1. Compile Time Exception ( Checked Exceptions)
2. Run-Time Exception (All sub-classes of Run-time Exception class) [Un-checked Exceptions]

Exceptions will end the program immediately if it is not properly handled.

Handling Exceptions

1. Using try/catch block (This is recommended method)
2. Using throws keyword

Arithmetic Exception. All the classes extending RunTimeException class are called Run-time Exceptions. Compiler will not throw any error message for not handling this type of exception.

* Checked exceptions needs to be handled properly
* Compiler will not allow to run the code without handling checked exceptions
* Un-checked exceptions or run-time exceptions will allow the code to run without handling it
* Try/catch method is the recommended approach to handle exceptions
* When using try/catch, try block needs to be followed by either catch block or by finally block and both.
* A try block with out catch or finally block is invalid.
* Throws keyword will pass the exception object to JVM which will not avoid pre-mature termination of Java program
* Try/catch block will help you to avoid pre-mature exit/closure of java program
* A try block can have multiple catch blocks and only one finally block
* The code in finally block will gets executed all the time irrespective of the exception status. This is the recommended place to close the resources, close the db connection & other objects.