WHAT IS OBJECT CLASS?

- OBJECT CLASS IS A SUPER CLASS FOR ALL THE CLASSES IN JAVA.
- OBJECT CLASS CONTAINS 11 PREDEFINED INSTANCE METHODS LIKE **toString(), equals(), hashcode(), getClass(), clone() finalize(), wait() and notify()** etc..
- IN JAVA ALL CLASS ARE INHERITED THIS METHODS.

WHAT IS STRING?

- STRING IS A SEQUENCE OF CHARACTER.
- STRING IS ENCLOSED WITHIN THE DOUBLE QUOTE.
- STRING IS A FINAL CLASS, SO SUB-CLASSES CAN'T EXTENDS THE STRING CLASS.
- STRING IS PRESENT IN JAVA. LANG PACKAGE.
- STRING IS **IMMUTABLE**.
- BY USING STRING CLASS WE CREATE A OBJECT THEN WE CAN'T CHANGE THE OBJECT INSTEAD OF THAT IT WILL CREATE A NEW OBJECT.

NOTE : STRING OBJECTS ARE STORED IN HEAP AREA AND STRING LITERALS ARE STORED IN SCP(STRING CONSTANT POOL).INSIDE HEAP AREA SCP IS PRESENT.

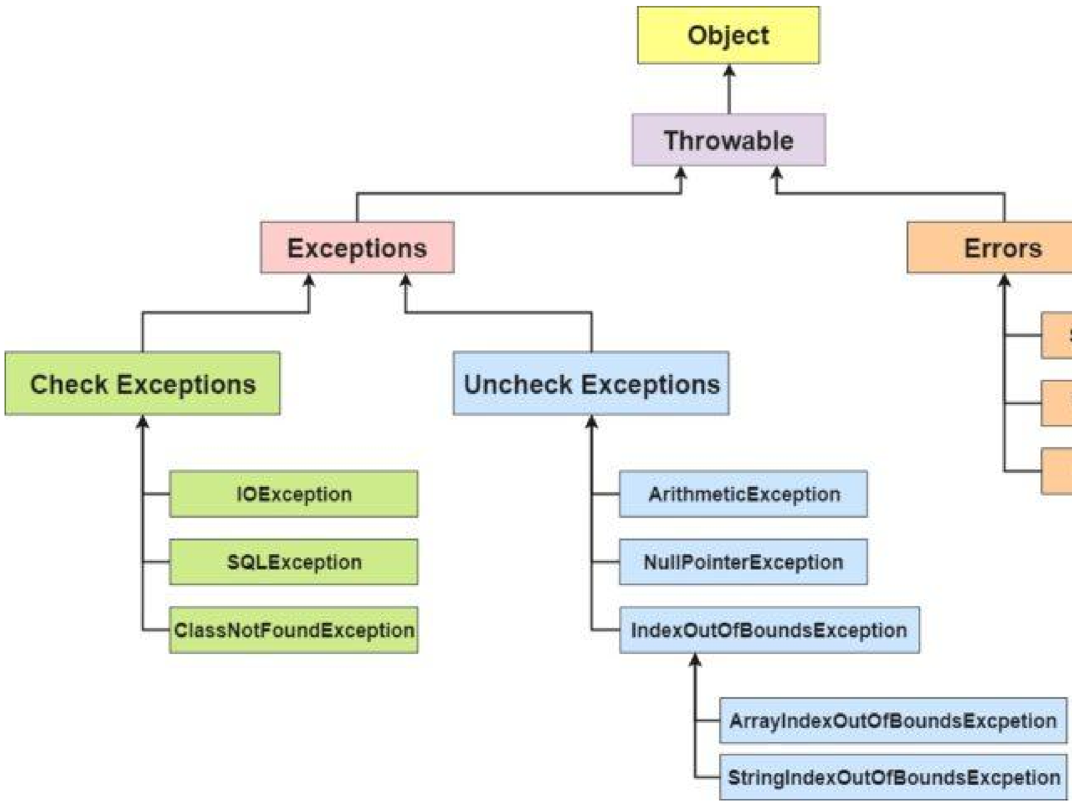
WHAT IS STRINGBUFFER?

- STRINGBUFFER IS SAME AS STRING.
- BUT STRINGBUFFER IS MUTABLE.
- BY USING STRINGBUFFER WE CAN CHANGE THE ORIGINAL OBJECT.
- STRINGBUFFER IS INTRODUCED IN 1.0 V.
- STRINGBUFFER IS SYNCHRONIZED AND IT IS THREAD SAFE.
- STRINGBUFFER IS LESS EFFICIENCY THAN STRINGBUILDER .

WHAT IS STRINGBUILDER?

- STRINGBUILDER IS SAME AS STRINGBUFFER.
- STRINGBUILDER IS INTRODUCED IN 1.5 V.
- STRINGBUILDER IS NON-SYNCHRONIZED AND IT IS NOT A THREAD SAFE.
- STRINGBUILDER IS MORE EFFICIENCY THAN STRINGBUFFER.

EXCEPTION HANDLING



WHAT IS EXCEPTION?

• AN EXCEPTION IS AN SUDDEN STOP OR UNEXPECTED EVENT, WHICH OCCURS DURING THE EXECUTION OF THE PROGRAM AT RUN TIME, IT AFFECT THE NORMAL FLOW OF THE PROGRAM'S.

• WHEN THE EXCEPTION IS HAPPEN REMAINING CODE WILL BE TERMINATED.

WHAT IS EXCEPTION HANDLING?

• EXCEPTION HANDLING IS A MECHANISM TO HANDLE THE RUNTIME EXCEPTION.

• THE MAIN ADVANTAGE OF EXCEPTION HANDLING IS TO MAINTAIN THE NORMAL FLOW OF THE APPLICATION BY USING TRY AND CATCH BLOCK.

CHECKED EXCEPTION

• CHECKED EXCEPTIONS ARE CHECKED AT COMPILE-TIME.

UNCHECKED EXCEPTION

• UNCHECKED EXCEPTIONS ARE CHECKED AT RUN TIME.

ERROR • ERROR IS IRRECOVERABLE.

TRY

- WE SHOULD WRITE THE ABNORMAL STATEMENTS IN TRY BLOCK.
- WE CANNOT USE TRY BLOCK ALONE, WE USE EITHER TRY WITH CATCH BLOCK OR TRY WITH FINALLY BLOCK.
- ONE TRY BLOCK CAN HAVE MULTIPLE CATCH BLOCK.

CATCH

- WE CANNOT USE CATCH ALONE, ALWAYS WE SHOULD USE CATCH WITH TRY BLOCK.
- WHICH CATCH BLOCK IS EXECUTED IS DEPENDS ON THE TYPE OF EXCEPTION THROWN.
- IF WE HANDLE THE EXCEPTION WE CAN GET A NORMAL FLOW OF THE PROGRAM, OTHERWISE WE CANNOT GET NORMAL FLOW OF THE PROGRAM.

FINALLY

- INSIDE THE FINALLY BLOCK WE SHOULD WRITE THE IMPORTANT INSTRUCTIONS THAT WILL BE EXECUTED EVEN EXCEPTION MAY OR MAY NOT HAPPEN.
- WE USE FINALLY BLOCK FOR CLOSING THE RESOURCES (FILES, DATABASE CONNECTION..),

THROW

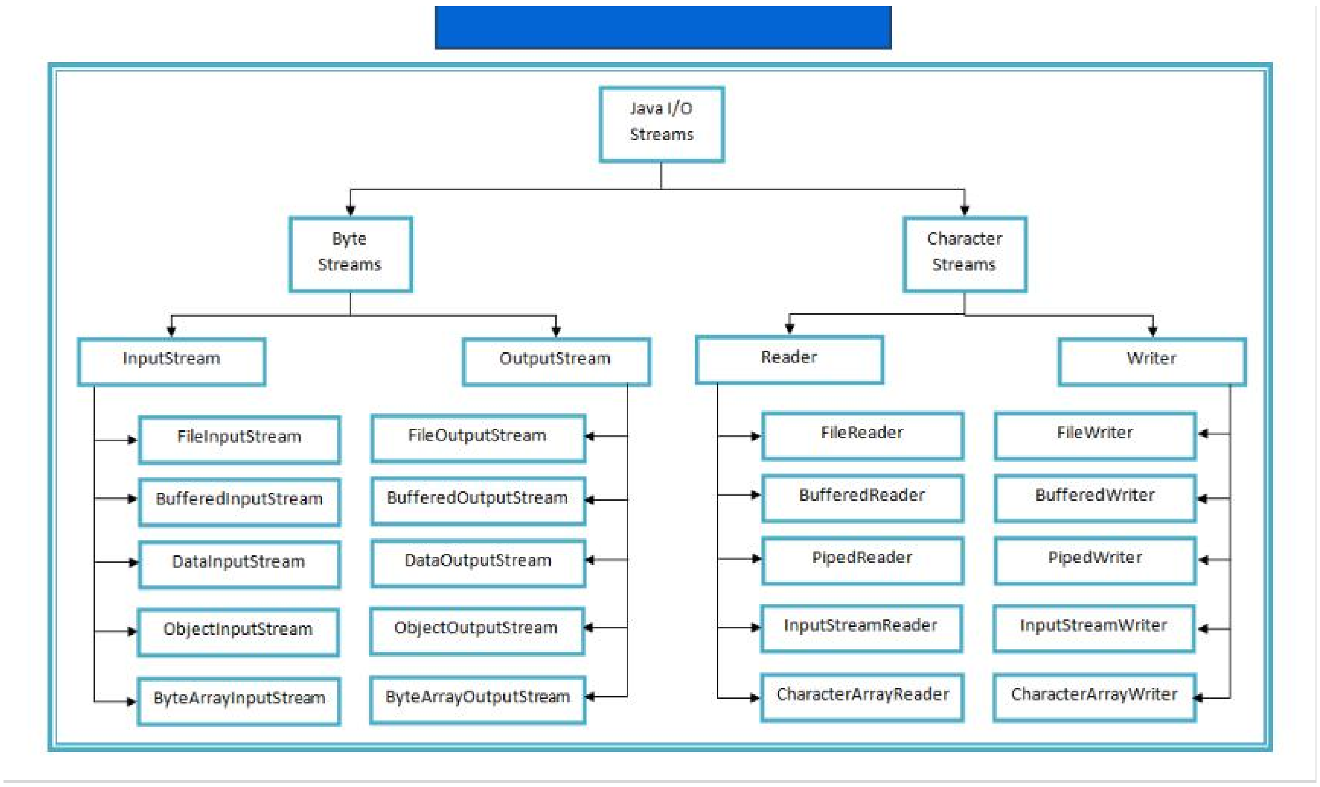
- THROW IS USED TO EXPLICITLY THROW AN EXCEPTION,WE CAN THROW ONLY ONE EXCEPTION.
- WE WANT TO STOP THE EXECUTION EXPLICITLY WE THROW AN EXCEPTION.
- WE USE THROW INSIDE THE METHOD BODY.

THROWS

- THROWS IS USED TO DECLARE AN EXCEPTION,WE CAN DECLARE MULTIPLE EXCEPTION.
- WE USE THROWS IN METHOD DECLARATION.

M.MUKESH

FILE HANDLING



STREAM : IT IS A INTERFACE.

BYTE STREAM : BYTE STREAMS SHOULD ONLY BE USED FOR THE MOST PRIMITIVE I/O.

CHARACTER STREAM : THE PRIMARY ADVANTAGE OF CHARACTER STREAMS IS THAT THEY MAKE IT EASY TO WRITE PROGRAMS THAT ARE NOT DEPENDENT UPON A SPECIFIC CHARACTER ENCODING, AND ARE THEREFORE EASY TO INTERNATIONALIZE.
JAVA STORES STRINGS IN UNICODE.

A SECOND ADVANTAGE OF CHARACTER STREAMS IS THAT THEY ARE POTENTIALLY MUCH MORE EFFICIENT THAN BYTE STREAMS.

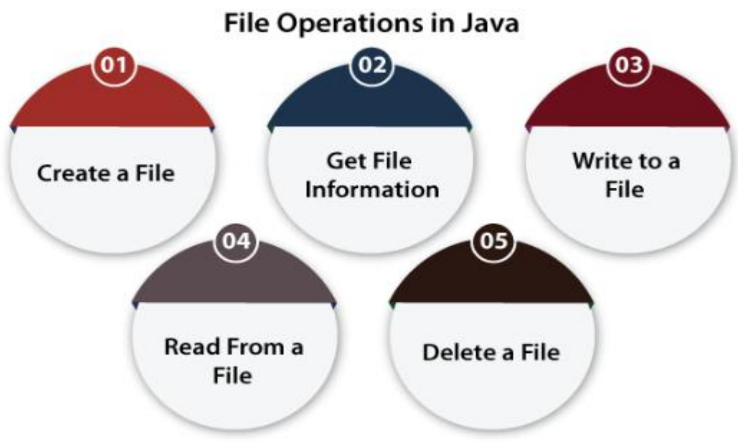
INPUTSTREAM AND OUTPUTSTREAM ARE ABSTRACT CLASS.

WHAT IS FILE HANDLING ?

- FILE HANDLING IS A PROCESS OF CREATING , READING, UPDATING AND DELETING THE FILE .
- IN FILE HANDLING WE CAN PERFORM CRUD OPERATIONS .

IN FILE HANDLING WE HAVE SOME METHODS

CREATENEWFILE() , WRITE(STR) , WRITE(INT) , DELETE() , WRITEOBJECT() , READOBJECT() , FLUSH() , CLOSE() , READY() , READ() , READLINE() ETC.



WHAT IS DATA STREAM?

- DATA STREAM SUPPORTS I/O OF PRIMITIVE DATATYPES.

WHAT IS OBJECT STREAM?

- OBJECT STREAM SUPPORTS I/O OF OBJECTS.

SERIALIZATION AND DESERIALIZATION

WHAT IS SERIALIZABLE ?

- SERIALIZABLE IS A MARKER INTERFACE .
- SERIALIZATION IS A PROCESS OF CONVERTING THE STATE OF AN OBJECT INTO BYTE STREAM .

CHARACTERISTICS OF SERIALIZABLE :

- IN SERIALIZATION WE CAN STORE TOTAL OBJECT TO THE FILE AND WE CANNOT STORE PART OF THE OBJECT .
- IN SERIALIZATION PERFORMANCE WAS LOW .
- IN SERIALIZATION CONSTRUCTOR IS NOT MANDATORY AND HERE WE USE TRANSIENT KEYWORD .
- IF WE WANT TO STORE TOTAL OBJECT GO WITH SERIALIZATION BECAUSE OF JVM WILL DO ALL THE THINGS .

WHAT IS EXTERNALIZABLE ?

• EXTERNALIZABLE IS A INTERFACE AND IT CONTAINS WRITEEXTERNAL() AND READEXTERNAL() .

CHARACTERISTICS OF EXTERNALIZABLE :

• THE ADVANTAGE OF EXTERNALIZABLE IS TO RECOVER LOSS OF INFORMATION BECAUSE OF TRANSIENT VARIABLE .
• HIGHLY RECOMMENDED TO USE SERIALVERSIONUID .
• IN EXTERNALIZATION WE CAN STORE TOTAL OBJECT OR PART OF AN OBJECT BASED ON OUR REQUIREMENT .
• IN EXTERNALIZATION PERFORMANCE WAS HIGH .
• IN EXTERNALIZATION CONSTRUCTOR IS MANDATORY AND HERE TRANSIENT KEYWORD IS NOT REQUIRED .
• IF WE WANT TO STORE PART OF AN OBJECT GO WITH EXTERNALIZATION BECAUSE WE CAN DO ALL THE THINGS .

TRANSIENT :

• IF YOU DON'T WANT TO SERIALIZE ANY DATA MEMBER OF A CLASS, YOU CAN MARK IT AS TRANSIENT.
• AFTER SERIALIZATION ANY UDATE WE PERFORM MEANS TRANSIENT MAKE IT AS DEFAULT VALUE .

SERIALVERSIONUID :

• SERIALIZABLE CLASS A VERSION NUMBER, CALLED A SERIALVERSIONUID .
IT IS USED TO VERIFY THE SENDER AND RECEIVER OF THE SERIALIZED OBJECT .
• TO VERIFY IT, SERIALVERSIONUID IS USED. THE SENDER AND RECEIVER MUST HAVE THE SAME SERIALVERSIONUID, OTHERWISE, INVALIDCLASSEXCEPTION WILL BE THROWN WHEN YOU DESERIALIZE THE OBJECT.

WHAT IS DESERIALIZATION ?

• DESERIALIZATION IS A PROCESS OF CONVERTING THE BYTE STREAM INTO STATE OF AN OBJECT.

WHAT IS ARRAY?

- ARRAY IS A LINEAR DATA STRUCTURE WHICH IS USED TO STORE HOMOGENEOUS(SAME TYPE) ELEMENTS.
- ARRAY SIZE IS FIXED AND ARRAY HAVING INDEX CONCEPT.
- ARRAY IS PRESENT IN JAVA.UTIL PACKAGE.
- ARRAY IS MUTABLE.
- IN ARRAY ALL THE DATA SHOULD BE OBJECT TYPE .

WE HAVE TWO TYPES OF ARRAY :

SINGLE DIMENTIONAL ARRAY	- int[] arr = {1,2,3,4,5};
MULTI DIMENTIONAL ARRAY	- int[][] arr = {{1,2}, {3,4}, {5,6}};

TWO WAYS TO CREATE A ARRAY :

STATIC TYPE	- datatype[] reference = {v1,v2,v3,v4,v5,…};
DYNAMIC TYPE	- datatype[] reference = new datatype[size];

WHY WE USE ARRAY ?

- WE CAN STORE (N) NUMBER OF DATA IN SINGLE VARIABLE.

WHAT IS THREAD?

- THREAD IS A LIGHT WEIGHT SUB-PROCESS OR SMALLEST UNIF OF PROCESS.
- GENERALLY ALL THE PROGRAMS HAVE AT LEAST ONE THREAD, THAT IS MAIN THREAD, THAT IS PROVIDED BY THE JVM AT THE EXECUTION OF THE PROGRAM .
- EVERY THREAD HAS A PRIORITY.
- THE ADVANTAGE IS IF WE USE MULTIPLE THREAD IN ONE PROCESS, ONE THREAD GET ERROR AND OTHER ALL THREADS ARE CONTINUE TO BE EXECUTE.

WHAT IS MULTITASKING IN JAVA?

- THE PROCESS OF EXECUTING MULTIPLE TASKS CONCURRENTLY IS KNOWN AS MULTITASKING .

THEY ARE :

1.PROCESS-BASED MULTITASKING (**MULTIPROCESSING**).
2.THREAD-BASED MULTITASKING (**MULTITHREADING**).

- EACH PROCESS ALLOCATES A SEPARATE MEMORY AREA AND PROCESS IS HEAVYWEIGHT.
- THREADS SHARE THE SAME ADDRESS AND THREADS ARE LIGHTWEIGHT.

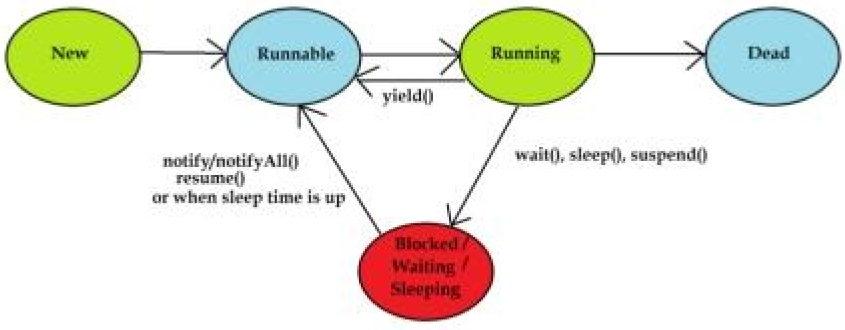
WHAT IS MULTI THREADING?

• THE PROCESS OF EXECUTING TWO OR MORE THREAD CONCURRENTLY IS KNOW AS MULTITHREADING.

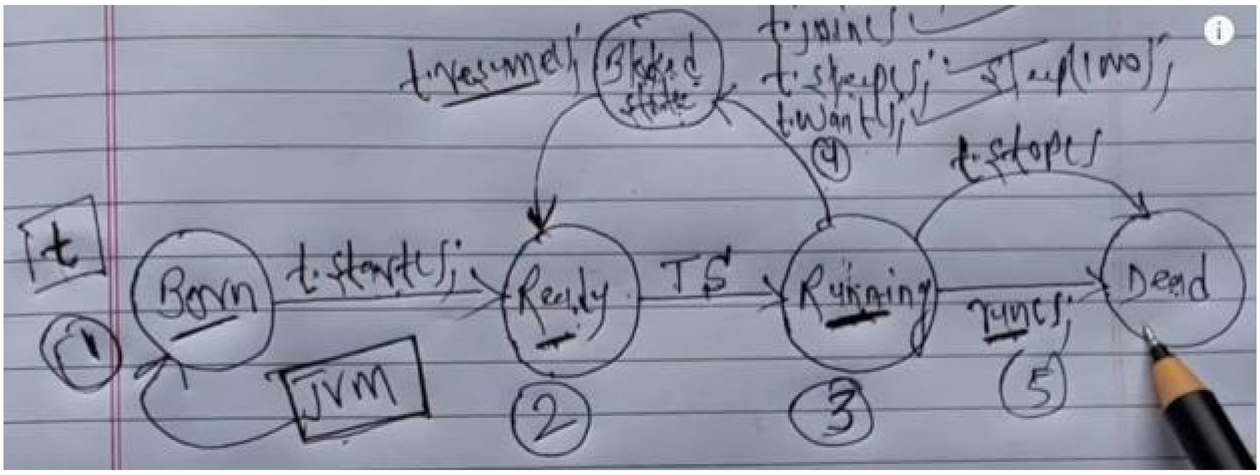
• MULTITHREADING CONCEPTS ARE USED IN GAMING AND ANIMATION.

• WE CAN ACHIEVE MULTITHREADING BY EXRENDS **THREAD** CLASS OR IMPLEMENTS **RUNNABLE** INTERFACE.

LIFE CYCLE OF THREAD



Thread Lifecycle using Thread states



NEW->**T1.START()**->RUNNABLE->**T1.RUN()**->RUNNING->
T1.SLEEP() /T1.WAIT()/T1.JOIN()->NON RUNNABLE->
T1.STOP()->STOP/DEAD

NEW STATE : WHEN A THREAD OBJECT IS CREATED USING NEW KEYWORD, IT IS IN A STATE THAT IS CALLED NEW STATE .

RUNNABLE STATE : THREAD IS RUNNABLE, MEANING IT CAN BE PICKED UP BY THE THREAD SCHEDULER TO RUN ANY TIME. BUT THE THREAD IS STILL NOT RUNNING(**T1.START()**).

RUNNING STATE : THE THREAD IS ACTUALLY RUN THE RUN().

• IN JAVA RESTART THE SAME THREAD IS NOT POSSIBLE
IF WE TRY TO RE-START THE SAME THREAD THEN WE GET **ILLEGALTHREADSTATEEXCEPTION.**

THREAD METHODS :

THREAD() ,THREAD(RUNNABLE TARGET) ,
THREAD(STRING NAME) ,
THREAD(THREADGROUP , RUNNABLE , STRING)

RUN() , GETID() , GETNAME() , GETPRIORITY() , GETSTATE()
GETTHREADGROUP() , INTERRUPT() , ISALIVE() , ISDAEMON()
JOIN() , SETDAEMON(BOOLEAN ON) , SETNAME(STRING NAME)
SETPRIORITY(INT NEWPRIORITY) , SLEEP(LONG MILLIS),YIELD() ,
CURRENTTHREAD() , WAIT() , NOTIFY() …

SUSPEND(), STOP() AND RESUME() ARE DEPRECATED.

WHAT IS VOLATILE?

• IF ANY ONE THREAD CAN CHANGE THE VARIABLE VALUE, IMMEDIATELY UPDATED VALUE WILL BE REFLECT TO OTHER THREADS.

WHAT IS DAEMON THREAD?

• DAEMON THREADS ARE LOW-PRIORITY THREADS.
• IT RUN IN THE BACKGROUND TO PERFORM TASKS SUCH AS GARBAGE COLLECTION OR PROVIDE SERVICES TO USER THREADS.
• THE LIFE OF A DAEMON THREAD DEPENDS ON USER THREADS, MEANING THAT WHEN ALL USER THREADS FINISH THEIR EXECUTION, THE JAVA VIRTUAL MACHINE (JVM) AUTOMATICALLY TERMINATES THE DAEMON THREAD.

WHAT IS DEADLOCK ?

• DEADLOCK DESCRIBES A CONDITION IN WHICH TWO OR MORE THREADS ARE BLOCKED (HUNG) FOREVER BECAUSE THEY ARE WAITING FOR EACH OTHER.

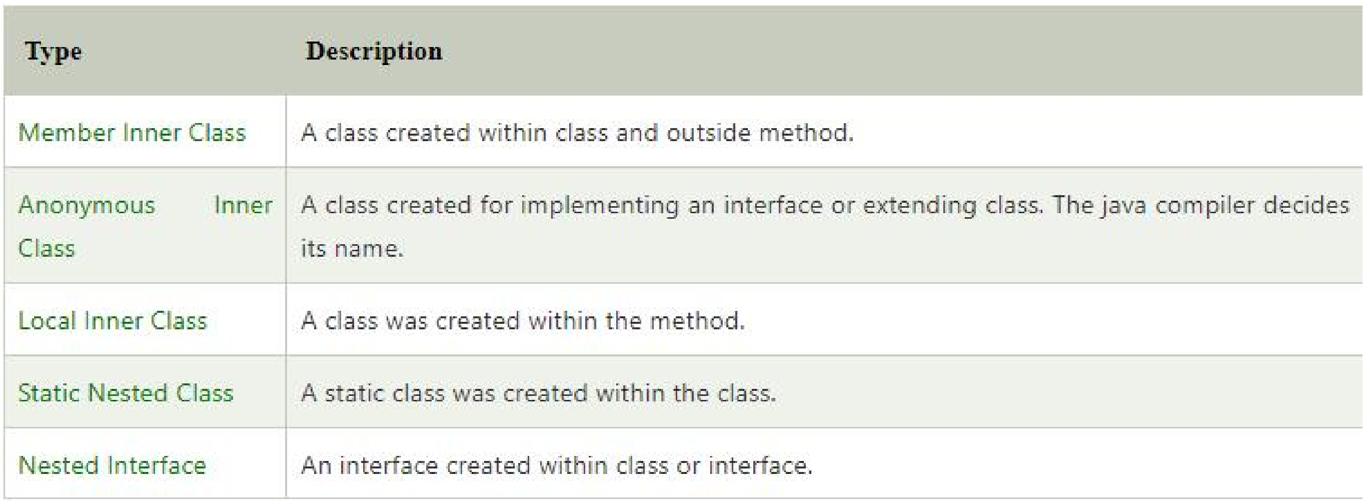
WHAT IS INNER CLASS?

• THE PROCESS OF CREATING A CLASS INSIDE THE ANOTHER CLASS IS KNOWN AS INNERCLASS.

• OUTER CLASS CANNOT BE STATIC BUT INNER CLASS CAN BE STATIC.

<u>IN THAT WE HAVE 4 TYPES OF INNER CLASS :</u>

1.STATIC INNER CLASS
2.INSTANCE INNER CLASS
3.ANONYMOUS INNER CLASS
4.METHOD INNER CLASS



Type	Description
Member Inner Class	A class created within class and outside method.
Anonymous Inner Class	A class created for implementing an interface or extending class. The java compiler decides its name.
Local Inner Class	A class was created within the method.
Static Nested Class	A static class was created within the class.
Nested Interface	An interface created within class or interface.

WHAT IS Comparable AND Comparator IN JAVA ?

• COMPARABLE IS A INTERFACE (because it predates the introduction of functional interfaces in Java 8).

• COMPARABLE PROVIDES A **SINGLE(NATURAL ORDERING) SORTING SEQUENCE**.

• COMPARABLE PROVIDES PUBLIC INT COMPARETO(T O); METHOD TO SORT ELEMENTS.

• COMPARABLE IS PRESENT IN **JAVA.LANG** PACKAGE.

• WE CAN SORT THE LIST ELEMENTS OF COMPARABLE TYPE BY COLLECTIONS.SORT(LIST) METHOD.

```
class T implements Comparable<T> {
    @Override
    public int compareTo(T t) {
        // comparasion logic
    }
}
```
public int compareTo(T o);

- COMPARATOR IS A FUNCTIONAL INTERFACE
- A COMPARATOR IS PRESENT IN THE **JAVA.UTIL** PACKAGE.
- COMPARATOR PROVIDES **MULTIPLE(CUSTOM) SORTING SEQUENCES**.
- COMPARATOR PROVIDES INT COMPARE(T O1, T O2); METHOD TO SORT ELEMENTS.
- WE CAN SORT THE LIST ELEMENTS OF COMPARATOR TYPE BY COLLECTIONS.SORT(LIST, COMPARATOR) METHOD.

int compare(T o1, T o2);

WHY WE USE MARKER INTERFACE?

- IT PROVIDES SOME INFORMATION ABOUT AN OBJECT TO THE JVM AT THE RUN TIME.
- WITHOUT IMPLEMENT THE SERIALIZABLE INTERFACE WE GET EXCEPTION.ONCE IMPLEMENT THE SERIALIZABLE INTERFACE THEN WE CAN'T GET ANY ERROR .