**JAVA.LANG.THROWABLE(ROOT CLASS)**

**!!!!!!!!!!!!!!!!!!!!EXCEPTION HANDLING!!!!!!!!!!!!!!!!!!!!!!!!!!**

* **Exception Handling:**(abnormal termination avoiding method)-

The Exception Handling in Java is one of the powerful mechanism to handle the runtime errors

So that normal flow of the application can be maintained.

OR

-An unpredictable, unexpected, unwanted event that disturb the normal flow of execution.

* Dictionary Meaning: Exception is an abnormal condition.

-In Java, an exception is an event that disrupts the normal flow of the program. It is an object

Which is thrown at runtime.

* Exception Handling is a mechanism to handle Exception such as ClassNotFoundException,

IOException, SQLException, RemoteException, Interrupted Exception etc. this are the compile time exception.

* THE java.lang.Throwable is root class for Exception which is divided into two types

1) Exception

2) Errors

1. **Exception type**

**1)Checked /Compile time Exception**

i)IOException.

ii)SQLException.

iii)ClassNotFoundException. Iv)IE V)Remote Exception

2)**Runtime Exception /Unchecked Exception**

i)ArithmeticException

ii)NULLPointerException

iii)NumberFormatException

iv)IndexOutOfBoundException

2)Errors type

1) VirtualMachineError

2) Stack overflow error

3) OUTOFMEMORY ERROR

**@There are three types of exception**

1) Checked exception

2) Unchecked exception

3) Errors

* **Actually every exception occur at runtime only**

\* Difference between Exception and Error

-Exception occur becz of programmer Mistakes where errors are occur becz of lack of system resources

-Exception are recoverable where Errors are not recoverable.

-Exception can be 2 types checked and unchecked where error can be only one type runtime (unchecked)

* **Checked Exception(compile time exception)**

-The exception which are checked by compiler for smooth execution of program are called Checked exception.

OR

-The classes which directly inherit Throwable class except RuntimeException and Error are known as checked exceptions e.g. IOException, SQLException etc. Checked exceptions are checked at compile-time.

OR

-Except the runtime exception and error everything is checked exception.

-if there is a chance of rising checked exception .then compulsory we should handle that exception [either by try catch or throws keyword otherwise we will get compile time error.

* **Unchecked Exception(Run time Exception)**

**-T**he classes which inherit RuntimeException are known as unchecked exceptions e.g. ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException etc. Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

* **Try Catch Block-**

-If an exception occurs at the particular statement of try block, the rest of the block code will not execute. So, it is recommended not to keeping the code in try block that will not throw an exception.

-Java try block must be followed by either catch or finally block.

-**When exception occur in any method then this method is responsible to create the exception object this object contain(exception name,details and stack trace location) then this method give this exception object to JVM and JVM will check have you handling the exception or not if not then it will calls default exception handler and abnormally terminate the execution.**

**-THERE ARE ThREE METHOD TO HANDLE EXCEPTION**

1. **using printStackTrace();**
2. **Using SOP;**
3. **Using SOP(e.getMessage());**
4. **SOP(e.toString());**whenever we trying to call object reference toString method by default is called

* **Finally Block-**

**-**we put the clean up code in the finally block .it will execute either exception occur or not. But only one finally block is allowed.

-Finalized method is invoked by GC before destroying the unreferenced object to perform the clean-up operation.

#### -Note: If you don't handle exception, before terminating the program, JVM executes finally block(if any).

#### -Rule: For each try block there can be zero or more catch blocks, but only one finally block.

#### -The finally block will not be executed if program exits(either by calling System.exit() or by causing a fatal error that causes the process to abort).

* **Throw keyword-**

The Java throw keyword is used to explicitly throw an exception.

-We can throw either checked or unchecked exception in java by throw keyword. The throw keyword is mainly used to throw custom exception

**-In This case throw programmer will create the object and throw the exception not main method creating object**

to throw the exception

SYNTAX:

throw new Exception\_class\_name("----");

ex:

throw new ArithematicException();

- Throw exception use for only for custom exception or user define exception you can

use for default exception but it is good for user defined exception

-if we extend from runtime exception then it will be unchecked exception or

-if you used extend exception then it will be checked exception but used always unchecked one

-we can not write anything after throw statement otherwise get c.t error saying not reachable any way

-If we compile time exception which we are throwing using throw keyword and if we using exception propagation then we have to handle that either through try catch in same method or through throws keyword if we are not handle then we get error in case of checked exception

-in unchecked exception if we are not handling then caller method is handling we don’t get any error

* **Throws Keyword:**

it always used for checked exception means compile time exception

if any null pointer exception occur its fault of programmer

**VIP POINTS**

**-If super class method does not throws any exception and child class method throwing checked exception then we will get compile time error.**

**-if super class method does not throws exception and we overriding the method and throwing exception which is runtime exception then program can execute as it is not c.t we get.**

**-if superclass method throws exception as subclass method override the superclass method then it must be throw exception same as super class it can not be more than superclass otherwise get c.t.**

**-if subclass method use the same exception then its ohk.**

**-if superclass use exception e and subclass use ArithmeticExcetion then its ohk.**

**-If superclass throws exception and subclass not throwing still its ohk.**