Exception Handling in Java 😀

Learning about exceptions and how do we handle them in a program so that our program doesn't crash at running.

Types of exceptions :

1. Syntax error
2. Logical error
3. Runtime error

Syntax Error : When we write something that the compiler doesn't understand that their is no dedicated rule for understanding that case then it falls in the compile time exceptions these are happens are compile time.

Logical Error : These exceptions happens when our logic fails out like we need to divide two numbers but we multiply them so these are fall under logical error.

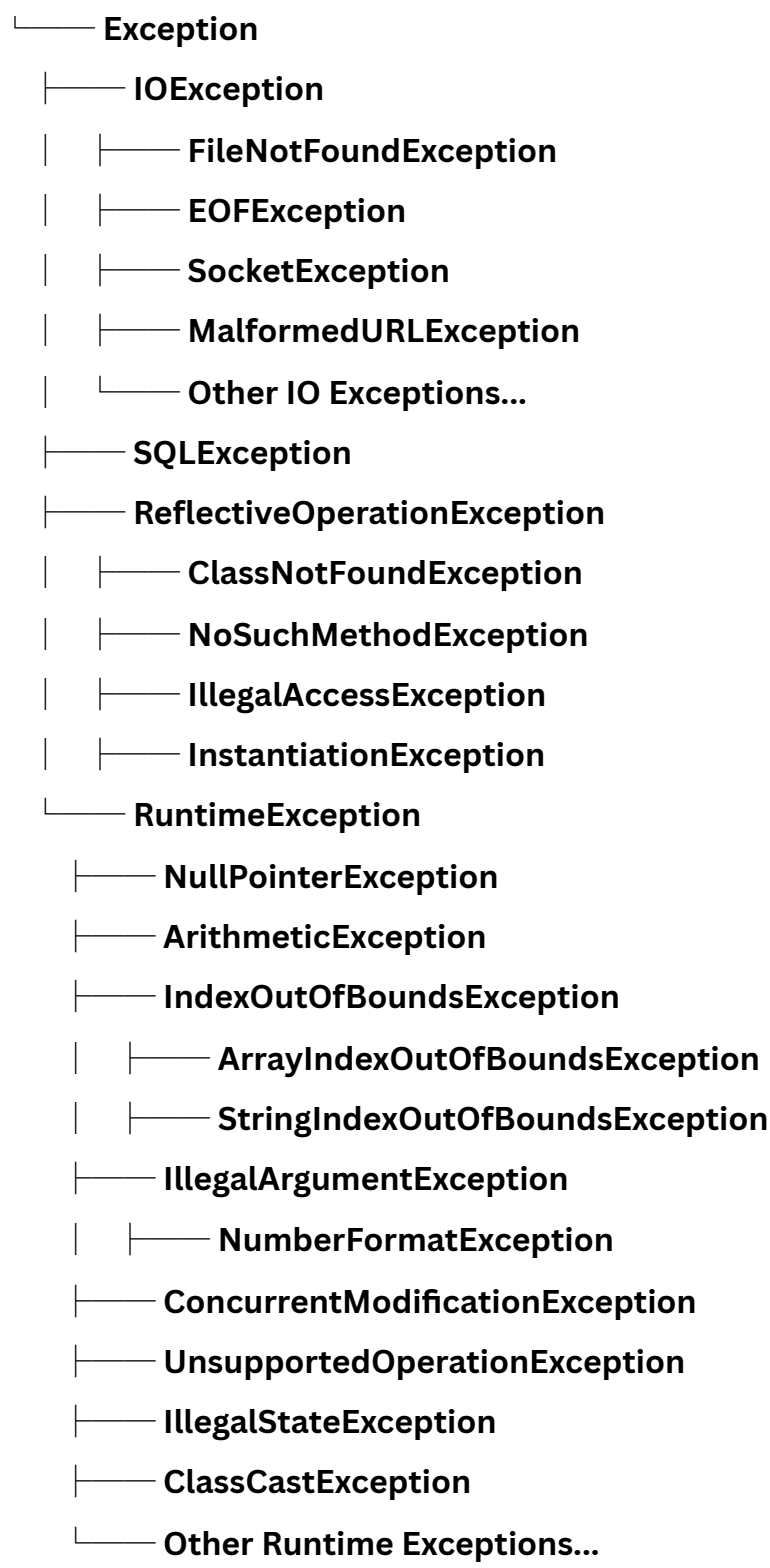
Runtime Error : Runtime errors are happened at runtime when we don't know but has a possibility that the particular case will throw out some error so we have to handle these exceptions.

Exception Handling : Exception handling is a way to handle the runtime errors so that the normal flow of the application can be maintained.

→ Exception is an event that disrupts the normal flow of the program. It is an object which is thrown at runtime.

Exception Hierarchy in Java 😀

Throwable
├── Error
│ ├── VirtualMachineError
│ │ ├── StackOverflowError
│ │ ├── OutOfMemoryError
│ │ └── InternalError
│ ├── AssertionError
│ ├── LinkageError
│ │ ├── ClassNotFoundError
│ │ ├── NoClassDefFoundError
│ │ ├── UnsupportedClassVersionError
│ └── Other Errors...



STACK TRACE IN JAVA :

Stack Trace is a way of giving a detailed information about the method calls like if a error occurred then in which method it has been occurred it gives a detailed info.

ex : e.printStackTrace(); // for tracking out the method calls.

CHECKED AND UNCHECKED EXCEPTIONS :

UNCHECKED EXCEPTIONS : The Exceptions which are not checked at the compile time are known as unchecked exceptions. These exceptions are occurred at runtime.

example : NullPointerException, ArrayIndexOutofBoundsException, ArithmeticException etc.

CHECKED EXCEPTIONS : Some of the exceptions which are occurred during compile time are known as checked exceptions.

These exceptions can be handled through two ways by catching out the exception using the try-catch block or implementing the throws keyword to the method signature.

Throws keyword : It aware the programmer that the certain method will throw out the particular exception so you want to handle it carefully.

Throw Keyword : This keyword is used for forcefully throwing out an exception. It can throw out any exception.

syntax : throw new FileNotFoundException(); // we are forcefully creating out an exception.

FINALLY KEYWORD : This keyword is followed by the try catch block and this block runs every time it is similar to the try except and else block in python but else block runs when no exception were occured.

try {} catch (Exception e) {} finally {} - syntax

Use Case of finally : Finally is mainly used for closing out the resources.

Try with resources : It is used for automatically closing out the resources in java. It is the successor for the finally block. JVM automate the closing process.

Custom Exceptions : We can create our custom exceptions for better logging by extending the custom exception class to the Exception class.