SKILLS

Lecture

Exception Handling

List of Concepts Involved:

- Different types of Errors in Java
- What is an Exception?
- try-catch
- Multiple catch block
- Handling vs Ducking an Exception
- Rethrowing an Exception(throw, throws,finally) and Custom Exception
- Hierarchy of an Exception class
- Control flow of Exception Handling concept
- try with Resources

Topics covered Yesterday's Session:

- Interface and Lambda Expression

Different types of Errors in Java

In any programming language we categorise errors into 3 types

1. Syntax Error
2. Logical Error
3. Runtime Error

What is an Exception?

- An unwanted/expected event that disturbs the normal flow of execution of a program is called "Exception handling".
- The main objective of Exception handling is to handle the exception.
- It is available for graceful termination of program.

PW SKILLS

Try-catch

Syntax of Exception handling

```
try{
    //risky code
}catch(Exception e){
    //handling logic
}
```

Try with multiple catch Blocks

The way of handling the exception is varied from exception to exception,hence for every exception type it is recommended to take a separate catch block. That is try with multiple catch blocks is possible and recommended to use.

```
try{
    ....
    ....
    ....
}catch(FileNotFoundException fe){

}catch(ArithmeticException ae){

}catch(SQLException se){

}catch(Exception e){

}
```

Handling vs Ducking an Exception

- It is highly recommended to handle exceptions
- In our program the code which may rise exception is called "risky code"
- We have to place our risky code inside the try block and corresponding handling code inside the catch block.

Rethrowing an Exception(throw, throws, finally) and Custom Exception

throw keyword in java

- This keyword is used in java to throw the exception object manually and inform jvm to handle the exception.

Syntax: throw new ArithmeticException("/ by zero");

Customized Exceptions (User defined Exceptions)

- Sometimes we can create our own exception to meet our programming requirements.
- Such type of exceptions are called customised exceptions (user defined exceptions).

Example

- InSufficientFundsException
- TooYoungException
- TooOldException

finally block

- It is not recommended to clean up code inside a try block because there is no guarantee for the execution of every statement inside a try block.
- It is not recommended to place clean up code inside the catch block becoz if there is no exception then the catch block won't be executed.
- we require some place to maintain clean up code which should be executed always irrespective of whether exceptions are raised or not raised and whether or not handled.
- Such type of best place is nothing but finally block.
- Hence the main objective of finally block is to maintain cleanup code.

Syntax:

```
try{
   risky code
}catch( X e){
   handling code
}finally{
   cleanup code
}
```

Difference b/w final,finally and finalize

final

- final is the modifier applicable for classes, methods and variables
- If a class is declared as the final then child class creation is not possible.
- If a method is declared as the final then overriding of that method is not possible.
- If a variable is declared as the final then reassignment is not possible.

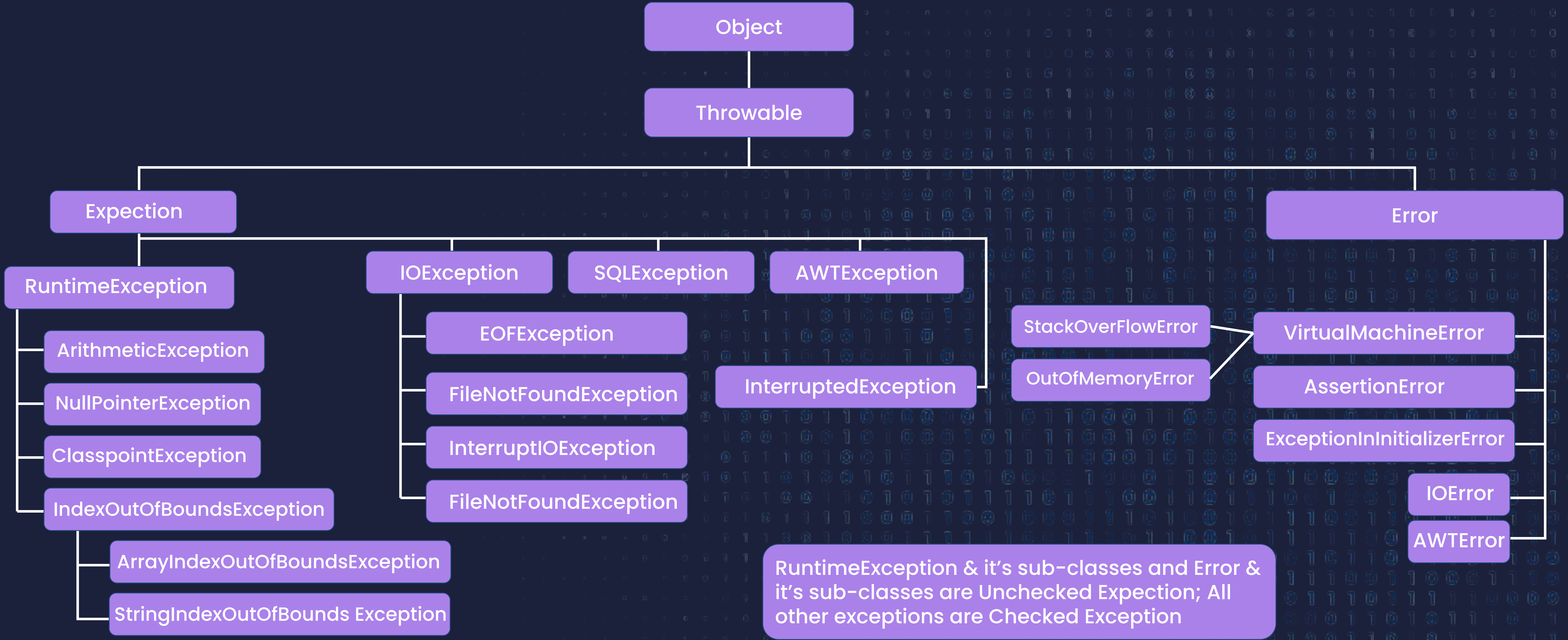
finally

It is a final block associated with try-catch to maintain clean up code, which should be executed always irrespective of whether exceptions are raised or not raised and whether handled or not handled.

finalize

It is a method, always invoked by Garbage Collector just before destroying an object to perform cleanup activities.

Hierarchy of an Exception class



RuntimeException & it's sub-classes and Error & it's sub-classes are Unchecked Expection; All other exceptions are Checked Exception

Control flow of Exception Handling concept

Control flow in try catch finally

```
try{
    statement-1
    statement-2
    statement-3
}catch(Exception e){
    statement-4
}finally{
    statement-5
}
     statement-6
```

Control flow in try catch finally

Case1:

If there is no exception.

Case2:

If an exception is raised at statement2 and the corresponding catch block is matched.

Case3

If an exception is raised at statement2 and corresponding catch block is not matched

Case4

If an exception is raised at statement4

Case5

If an exception is raised at statement5

try with resources

- In this approach, the resources which are opened as a part of try block will be closed automatically once the control reaches to the end of
- try block normally or abnormally,so it is not required to close explicitly so the complexity of the program would be reduced.
- It is not required to write a finally block explicitly,so length of the code would be reduced and readability is improved.

```
try(BufferedReader br=new BufferedReader(new FileReader("abc.txt"))){
    //use br and perform the necessary operation
      //once the control reaches the end of try automatically br will be closed
}catch(IOException ie){
    //handling code
}
```

Next Lecture

- MultiThreading

SKILLS

THANK YOU