Top of Form

#### ****Exception Object:****

An exception object is an instance of an exception class. It gets created and handed to the Java runtime when an exceptional event occurred that disrupted the normal flow of the application. This is called “to throw an exception” because in Java you use the keyword “throw” to hand the exception to the runtime. When a method throws an exception object, the runtime searches the call stack for a piece of code that handles it.

#### ****Exception Hierarchy:****

All exception classes are subtypes of the java.lang.Exception class. The exception class is a subclass of the Throwable class. Other than the exception class there is another subclass called Error which is derived from the Throwable class.

Errors are abnormal conditions that happen in case of severe failures, these are not handled by the Java programs. Errors are generated to indicate errors generated by the runtime environment. Example: JVM is out of memory. Normally, programs cannot recover from errors.

The Exception class has two main subclasses: IOException class and RuntimeException Class.

#### ****Exceptions Methods****

Following is the list of important methods available in the Throwable class.

#### Exception and Error

Exceptions and errors both are subclasses of Throwable class. The error indicates a problem that mainly occurs due to the lack of system resources and our application should not catch these types of problems. Some of the examples of errors are system crash error and out of memory error. Errors mostly occur at runtime that's they belong to an unchecked type.

Exceptions are the problems which can occur at runtime and compile time. It mainly occurs in the code written by the developers. Exceptions are divided into two categories such as checked exceptions and unchecked exceptions.

#### ****Example of Error****

public class ErrorExample {

  public static void main(String[] args){

   recursiveMethod(10)

  }

  public static void recursiveMethod(int i){

   while(i!=0){

     i=i+1;

     recursiveMethod(i);

   }

  }

}

#### ****Output****

Exception in thread "main" java.lang.StackOverflowError

  at ErrorExample.ErrorExample(Main.java:42)

#### ****Example of Exception****

public class ExceptionExample {

  public static void main(String[] args){

   int x = 100;

   int y = 0;

   int z = x / y;

  }

}

#### ****Output****

java.lang.ArithmeticException: / by zero

  at ExceptionExample.main(ExceptionExample.java:7)

Bottom of Form

Top of Form

Bottom of Form

Top of Form

##### **Reference Links:**

**ONLINE NOTES LINKS:**

[https://www.tutorialspoint.com/java/java\_exceptions.htm](https://www.tutorialspoint.com/java/java_exceptions.htm#:~:text=An%20exception%20(or%20exceptional%20event,exceptions%20are%20to%20be%20handled)

<https://www.javatpoint.com/exception-handling-in-java>

<https://www.geeksforgeeks.org/exceptions-in-java/>

<https://beginnersbook.com/2013/04/java-exception-handling/>

<https://docs.oracle.com/javase/tutorial/essential/exceptions/definition.html>

**VIDEO LINKS:**

<https://www.javatpoint.com/exception-handling-in-java>

<https://www.edureka.co/blog/java-exception-handling>

<https://marcus-biel.com/advanced-exception-handling-in-java/>

Bottom of Form