**Exception:**

* During the execution of java program, JVM faces the abnormal situations based on the code declaration.
* If the JVM faces the abnormal situations then, JVM triggers an event, this event is known as Exception.
* If exception event got generated in the java program then it results in the termination of the java program.
* If the termination of the java program takes place then the code is considered to be non- feasible code for the execution.
* If any event is generated by the JVM then the program need to handle the event, so that all the lines present in the program gets executed.
* E.g. public class **Demo1**{

public static void main(string args[])

{ int a=10;

int b=0;

int c=a/b;

System.out.println(“Result is ”+c);

}

}

**Output** : Exception in thread main.Java.Lang.Arithmatic.Exception / by zero at Demo1.main

**Exception Handling:** Handling the event generated by JVM during program execution is known as execution handling.

It is recommended to handle events during execution flow only.

Int a= 10/0;

Result in the normal form of the program.

All remaining statements are going to get executed.

JVM dies before the JVM prints the reason for the termination

Handled?

No No Yes

To handle the event or exception, below blocks are used.

**Syntax:**

try{

//risky code

}

catch(event/exceptionName refVariable){

//event handled message

}

**Try Block:**

* It is used to declare risky code only.
* Controller visits inside the try block only once through out the life time of the program.
* Try block should be followed by either catch or finally
* Multiple try blocks are not allowed.

**Catch Block:**

* It is used to handle the event generated from try block.
* Catch block will get executed only if the event get generated in the try block.
* Catch block should be declared after try block.
* Any number of catch blocks can be declared for single try block.

**Finally**:

* Finally is a block used to close the costly resources of current program.
* Finally block will get executed in all the circumstances.
* Finally block should be followed by catch block you can declare finally block after try block, but it is not recommended.

**printStacktrace():**

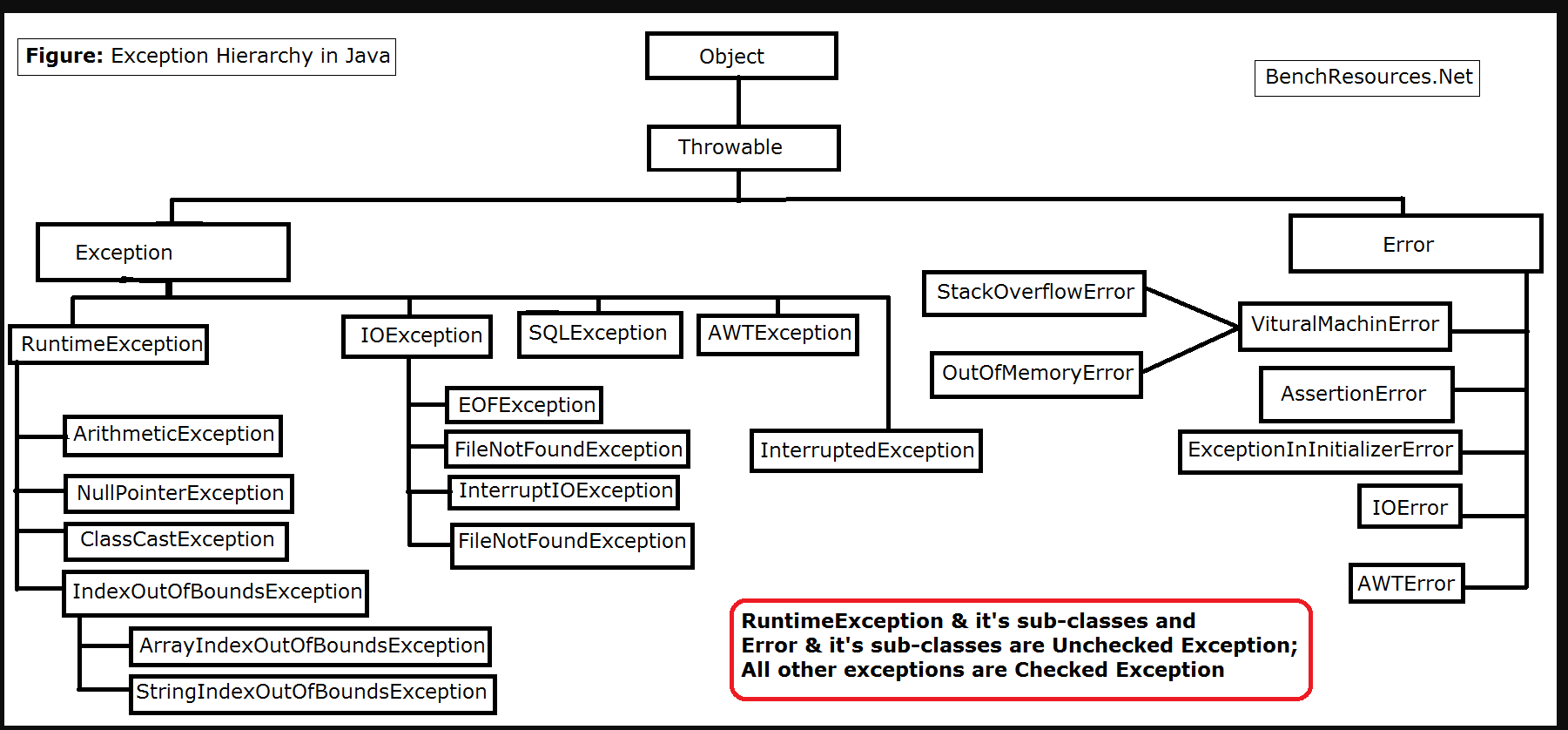
It is a method used to get fully qualified details of an execution.

**Difference Between “Throw” and “Throws”:**

**Throw**: throw is a keyword use to throw new custom exception in current block.

**Throws**: throws is a keyword use to show / declared type of exception generates inside method or class.





What is Exception in Java?

In Java, an exception is an event that **disrupts** the normal flow of the program. It is an object which is thrown at **runtime**.

The core advantage of exception handling is **to maintain the normal flow of the application**. An exception normally disrupts the normal flow of the application; that is why we need to handle exceptions.

Difference between **Checked** and **Unchecked** Exceptions

1) Checked Exception

The classes that directly inherit the Throwable class except RuntimeException and Error are known as checked exceptions. For example, IOException, SQLException, etc. Checked exceptions are checked at compile-time.

2) Unchecked Exception

The classes that inherit the RuntimeException are known as unchecked exceptions. For example, ArithmeticException, NullPointerException, ArrayIndexOutOfBoundsException, etc. Unchecked exceptions are not checked at compile-time, but they are checked at runtime.

3) Error

Error is irrecoverable. Some example of errors are OutOfMemoryError, VirtualMachineError, AssertionError etc.

Java Exception Keywords

Java provides five keywords that are used to handle the exception. The following table describes each.

|  |  |
| --- | --- |
| **Keyword** | **Description** |
| try | The "try" keyword is used to specify a block where we should place an exception code. It means we can't use try block alone. The try block must be followed by either catch or finally. |
| catch | The "catch" block is used to handle the exception. It must be preceded by try block which means we can't use catch block alone. It can be followed by finally block later. |
| finally | The "finally" block is used to execute the necessary code of the program. It is executed whether an exception is handled or not. |
| throw | The "throw" keyword is used to throw an exception. |
| throws | The "throws" keyword is used to declare exceptions. It specifies that there may occur an exception in the method. It doesn't throw an exception. It is always used with method signature. |

**package** exception;

**public** **class** ArithMaticExp {

**public** **static** **void** main(String[] args)

{

**int** a=10;

**int** b=0;

**int** div = 0;

System.***out***.println(a);

System.***out***.println(b);

**try**

{

div=a/b;// risky code

}

**catch** (ArithmeticException e)

{

System.***out***.println("we cant divide by zero, please check value of b");

}

System.***out***.println("Ans is "+div);

}

}

**package** exception;

**import** java.io.File;

**public** **class** NullPointerStudy {

**public** **static** **void** main(String[] args) **throws** InterruptedException

{

String a="abc";

String b=**null**;

Thread.*sleep*(10);

**try**

{

//System.out.println(a.charAt(10)); //risky code

System.***out***.println(10/0);

}

**catch** (NullPointerException e)

{

System.***out***.println("check String, string should not be null");

//e.printStackTrace();

}

**catch** (StringIndexOutOfBoundsException e)

{

System.***out***.println("check String index or size");

}

**catch** (Exception e)

{

System.***out***.println("There some exception, plz check");

}

System.***out***.println("Hi GM");

}

}

**package** exception;

**public** **class** TryCatchFinally {

**public** **static** **void** main(String[] args)

{

String a="abc";

**try**

{

System.***out***.println(a.charAt(10));// risky code

}

**catch** (NullPointerException e)

{

System.***out***.println("Check index, index should in betn 0-2");

}

**finally**

{

System.***out***.println("Finally block is running");

}

}

}

**package** ExceptionStudy;

**public** **class** UseOfFinally {

**public** **static** **void** main(String[] args)

{

**int** a=10;

**int** b=0;

**try**

{

**int** div=a/b;

System.***out***.println(div);

}

**finally**

{

System.***out***.println("Finally print this");

}

System.***out***.println("hi good even");

// catch (ArithmeticException e)

//

// {

// System.out.println("Divide by zero is not possible");

// }

}

}

**package** ExceptionStudy;

**public** **class** TryCatchFinally {

**public** **static** **void** main(String[] args)

{

String a="Testing";

**try**

{

System.***out***.println(a.length());

}

**catch** (NullPointerException e)

{

System.***out***.println("Please check string value, should not be null");

}

**finally**

{

System.***out***.println("Good evening");

}

}

}