# Exception Handling in Java

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.

**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.**

**An exception is an unwanted or unexpected event, which occurs during the execution of a program i.e at run time, that disrupts the normal flow of the program’s instructions.**

**Error vs Exception**

**Error: An Error indicates serious problem that a reasonable application should not try to catch.**

**Exception: Exception indicates conditions that a reasonable application might try to catch.**

**Exception Hierarchy:**

All exception and errors types are sub classes of class **Throwable**, which is base class of hierarchy. One branch is headed by **Exception**.

**This class is used for exceptional conditions that user programs should catch.**

**NullPointerException is an example of such an exception.**

Another branch,**Error** are used by the Java run-time system([JVM](https://www.geeksforgeeks.org/jvm-works-jvm-architecture/)) to indicate errors having to do with the run-time environment itself(JRE).

**StackOverflowError is an example of such an error.**

[](http://cdncontribute.geeksforgeeks.org/wp-content/uploads/Exception-in-java1.png)

**Try Catch Block is used for implementing Exceptional handling:**

**How to use try-catch clause**

**try {**

**// block of code to monitor for errors**

**// the code you think can raise an exception**

**}**

**catch (ExceptionType1 exOb) {**

**// exception handler for ExceptionType1**

**}**

**catch (ExceptionType2 exOb) {**

**// exception handler for ExceptionType2**

**}**

**// optional**

**finally {**

**// block of code to be executed after try block ends**

**}**

|  |
| --- |
| **// Java program to demonstrate ArithmeticException**  **class ArithmeticException\_Demo**  **{**  **public static void main(String args[])**  **{**  **try**  **{**  **int a = 30, b = 0;**  **int c = a/b;  // cannot divide by zero**  **System.out.println ("Result = " + c);**  **}**  **catch(ArithmeticException e)**  **{**  **System.out.println ("Can't divide a number by 0");**  **}**  **}**  **}** |

**Output:**

**Can't divide a number by 0**

|  |
| --- |
| **//Java program to demonstrate NullPointerException**  **class NullPointer\_Demo**  **{**  **public static void main(String args[])**  **{**  **try {**  **String a = null; //null value**  **System.out.println(a.charAt(0));**  **}**  **catch(NullPointerException e)**  **{**  **System.out.println("NullPointerException..");**  **}**  **}**  **}** |

**Output:**

**NullPointerException**..

|  |
| --- |
| **// Java program to demonstrate StringIndexOutOfBoundsException**  **class StringIndexOutOfBound\_Demo**  **{**  **public static void main(String args[])**  **{**  **try**  **{**  **String a = "This is like chipping "; // length is 22**  **char c = a.charAt(24); // accessing 25th element**  **System.out.println(c);**  **}**  **catch(StringIndexOutOfBoundsException e)**  **{**  **System.out.println("StringIndexOutOfBoundsException");**  **}**  **}**  **}** |

**Output:**

**StringIndexOutOfBoundsException**

|  |
| --- |
| **//Java program to demonstrate FileNotFoundException**  **import java.io.File;**  **import java.io.FileNotFoundException;**  **import java.io.FileReader;**    **class File\_notFound\_Demo**  **{**  **public static void main(String args[])  {**  **try**  **{**  **// Following file does not exist**  **File file = new File("**[**E://file.txt**](file:///E:\file.txt)**");**  **FileReader fr = new FileReader(file);**  **}**  **catch (FileNotFoundException e)**  **{**  **System.out.println("File does not exist");**  **}**  **}**  **}** |

**Output:**

**File does not exist**

|  |
| --- |
| **// Java program to demonstrate NumberFormatException**  **class  NumberFormat\_Demo**  **{**  **public static void main(String args[])**  **{**  **try {**  **// "akki" is not a number**  **int num = Integer.parseInt ("akki") ;**    **System.out.println(num);**  **}**  **catch(NumberFormatException e)**  **{**  **System.out.println("Number format exception");**  **}**  **}**  **}** |

**Output:**

**Number format exception**

|  |
| --- |
| **// Java program to demonstrate ArrayIndexOutOfBoundException**  **class ArrayIndexOutOfBound\_Demo**  **{**  **public static void main(String args[])**  **{**  **try{**  **int a[] = new int[5];**  **a[6] = 9; // accessing 7th element in an array of**  **// size 5**  **}**  **catch(ArrayIndexOutOfBoundsException e){**  **System.out.println ("Array Index is Out Of Bounds");**  **}**  **}**  **}** |

**Output:**

**Array Index is Out Of Bounds**

**throw and throws in Java**

**throw**

**The throw keyword in Java is used to explicitly throw an exception from a method or any block of code. We can throw either**[**checked or unchecked exception**](https://www.geeksforgeeks.org/checked-vs-unchecked-exceptions-in-java/)**.**

**The throw keyword is mainly used to throw custom exceptions.**

**Syntax:**

**throw *Instance***

**Example:**

**throw new ArithmeticException("/ by zero");**

**But this exception i.e, *Instance* must be of type Throwable or a subclass of Throwable. For example Exception is a sub-class of Throwable and**[**user defined exceptions typically extend Exception class**](https://www.geeksforgeeks.org/g-fact-32-user-defined-custom-exception-in-java/)**.**

|  |
| --- |
| **// Java program that demonstrates the use of throw**  **class ThrowExcep**  **{**  **static void fun()**  **{**  **try**  **{**  **throw new NullPointerException("demo");**  **}**  **catch(NullPointerException e)**  **{**  **System.out.println("Caught inside fun().");**  **throw e; // rethrowing the exception**  **}**  **}**    **public static void main(String args[])**  **{**  **try**  **{**  **fun();**  **}**  **catch(NullPointerException e)**  **{**  **System.out.println("Caught in main.");**  **}**  **}**  **}** |
|  |
|  |
|  |

**Output:**

**Caught inside fun().**

**Caught in main.**

**Another Example:**

|  |
| --- |
| **// Java program that demonstrates the use of throw**  **class Test**  **{**  **public static void main(String[] args)**  **{**  **System.out.println(1/0);**  **}**  **}** |

**Output:**

**Exception in thread "main" java.lang.ArithmeticException: / by zero**

**throws**

**throws is a keyword in Java which is used in the signature of method to indicate that this method might throw one of the listed type exceptions. The caller to these methods has to handle the exception using a try-catch block.**

**Syntax:**

**type method\_name(parameters) throws exception\_list**

**exception\_list is a comma separated list of all the exceptions which a method might throw.**

|  |
| --- |
| **// Java program to demonstrate working of throws**  **class ThrowsExecp**  **{**  **static void fun() throws IllegalAccessException**  **{**  **System.out.println("Inside fun(). ");**  **throw new IllegalAccessException("demo");**  **}**  **public static void main(String args[])**  **{**  **try**  **{**  **fun();**  **}**  **catch(IllegalAccessException e)**  **{**  **System.out.println("caught in main.");**  **}**  **}**  **}** |

**Output:**

**Inside fun().**

**caught in main.**