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.

In this page, we will learn about Java exceptions, its type and the difference between checked and unchecked exceptions.

What is Exception in Java?

**Dictionary Meaning:** 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.

## Hierarchy of Java Exception classes

The java.lang.Throwable class is the root class of Java Exception hierarchy , only objects that are instances of this class are thrown by the JVM. It is inherited by two subclasses: Exception and Error. A hierarchy of Java Exception classes are given below:



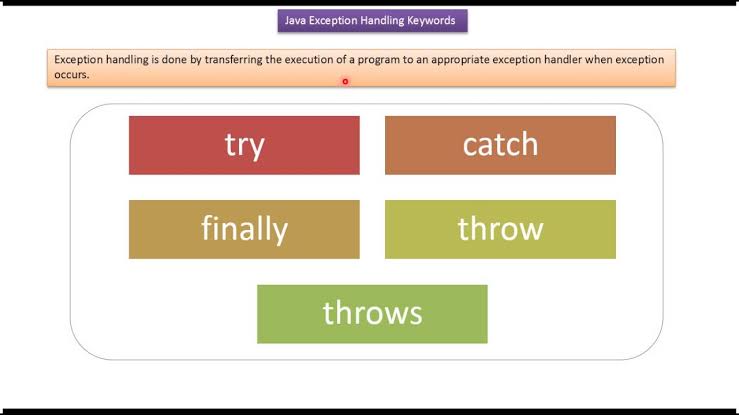
### Types of Java Exceptions

There are mainly two types of exceptions: checked and unchecked. Here, an error is considered as the unchecked exception.

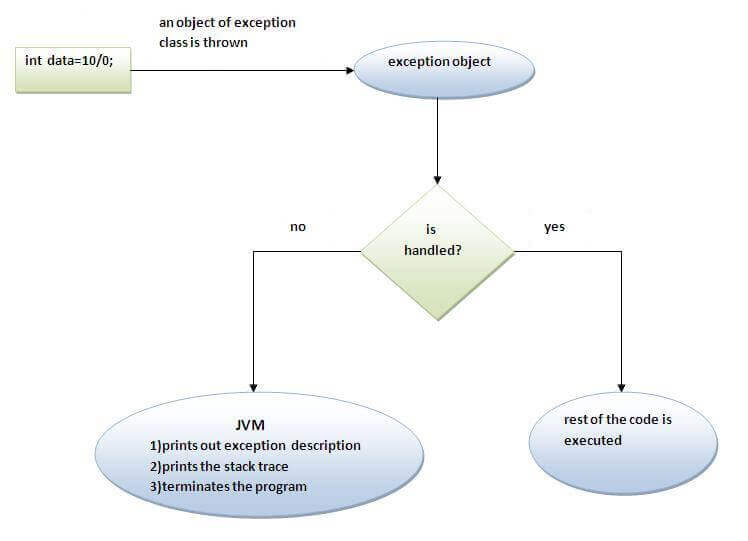
# Errors V/s Exceptions In Java

|  |  |
| --- | --- |
| Errors | Exceptions |
| Errors in java are of type java.lang.Error. | Exceptions in java are of type java.lang.Exception. |
| All errors in java are unchecked type. | Exceptions include both checked as well as unchecked type. |
| Errors happen at run time. They will not be known to compiler. | Checked exceptions are known to compiler where as unchecked exceptions are not known to compiler because they occur at run time. |
| It is impossible to recover from errors. | You can recover from exceptions by handling them through try-catch blocks. |
| Errors are mostly caused by the environment in which application is running. | Exceptions are mainly caused by the application itself. |
| Examples : java.lang.StackOverflowError, java.lang.OutOfMemoryError | Examples : Checked Exceptions : SQLException, IOException Unchecked Exceptions : ArrayIndexOutOfBoundException, ClassCastException, NullPointerException |

## Java Exception Keywords



## Internal working of java try-catch block



# Difference between throw and throws in Java

|  |  |  |
| --- | --- | --- |
| **No.** | **throw** | **throws** |
| 1) | Java throw keyword is used to explicitly throw an exception. | Java throws keyword is used to declare an exception. |
| 2) | Checked exception cannot be propagated using throw only. | Checked exception can be propagated with throws. |
| 3) | Throw is followed by an instance. | Throws is followed by class. |
| 4) | Throw is used within the method. | Throws is used with the method signature. |
| 5) | You cannot throw multiple exceptions. | You can declare multiple exceptions e.g. public void method()throws IOException,SQLException. |

**Syntax:**

void Demo() throws ArithmeticException, NullPointerException

{

// Statements where exceptions might occur.

throw new ArithmeticException();

}

# ExceptionHandling with MethodOverriding in Java

#### **1) Rule: If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception.**

#### **2) Rule: If the superclass method does not declare an exception, subclass overridden method cannot declare the checked exception but can declare unchecked exception.**

#### **3) Rule: If the superclass method declares an exception, subclass overridden method can declare same, subclass exception or no exception but cannot declare parent exception.**