Java Exception Handling

Exception Handling is a mechanism to handle exception at runtime. Exception is a condition that occurs during program execution and lead to program termination abnormally. There can be several reasons that can lead to exceptions, including programmer error, hardware failures, files that need

> to be opened cannot be found, resource exhaustion etc.
>
> Suppose we run a program to read data from a file and if the file is not

available then the program will stop execution and terminate the program by reporting the exception message.

The problem with the exception is, it terminates the program and skip rest of the execution that means if a program have 100 lines of code and at line 10 an exception occur then program will terminate immediately by skipping execution of rest 90 lines of code.

≡ Index

To handle this problem, we use exception handling that avoid program termination and continue the execution by skipping exception code.

Java exception handling provides a meaningful message to the user about the issue rather than a system generated message, which may not be understandable to a user.

Uncaught Exceptions

Lets understand exception with an example. When we don't handle the exceptions, it leads to unexpected program termination. In this program, an

ArithmeticException will be throw due to divide by zero.

```
class UncaughtException
{
 public static void main(String args[])
 {
  int a = 0;
  int b = 7/a;     // Divide by zero, will lead to exception
 }
}
```

This will lead to an exception at runtime, hence the Java run-time system will construct an exception and then throw it. As we don't have any mechanism for handling exception in the above program, hence the default handler (JVM) will handle the exception and will print the details of the exception on the terminal.

Uncaught Exception in java

The uncaught exceptions are the exceptions that are not caught by the compiler but automatically caught and handled by the Java built-in exception handler.

Java Exception

A Java Exception is an object that describes the exception that occurs in a program. When an exceptional events occurs in java, an exception is said to be thrown. The code that's responsible for doing something about the exception is called an **exception handler**

How to Handle Exception

Java provides controls to handle exception in the program. These controls are listed below.

- Try : It is used to enclose the suspected code.

- Catch: It acts as exception handler.
- Finally: It is used to execute necessary code.
- Throw: It throws the exception explicitly.
- Throws: It informs for the possible exception.

We will discuss about all these in our next tutorials.

Types of Exceptions

In Java, exceptions broadly can be categories into checked exception, unchecked exception and error based on the nature of exception.

- **Checked Exception**

  The exception that can be predicted by the JVM at the compile time for **example** : File that need to be opened is not found, SQLException etc. These type of exceptions must be checked at compile time.

- **Unchecked Exception**

  Unchecked exceptions are the class that extends RuntimeException class. Unchecked exception are ignored at compile time and checked at runtime. For **example** : ArithmeticException, NullPointerException, Array Index out of Bound exception. Unchecked exceptions are checked at runtime.

- **Error**

  Errors are typically ignored in code because you can rarely do anything about an error. For **example**, if stack overflow occurs, an error will arise. This type of error cannot be handled in the code.

Java Exception class Hierarchy

All exception types are subclasses of class **Throwable**, which is at the top of exception class hierarchy.

exception handling in java

- **Exception** class is for exceptional conditions that program should catch. This class is extended to create user specific exception classes.
- **RuntimeException** is a subclass of Exception. Exceptions under this class are automatically defined for programs.
- **Exceptions** of type **Error** are used by the Java run-time system to indicate errors having to do with the run-time environment, itself. Stack overflow is an example of such an error.

← Prev

Next →

Explore

Ja Java Programs

Java programs with code and output for practice.

Explore

Sp Spring Framework

Learn the most widely used Java framework in the World.

Explore

Interview Tests

Curious

Practice Coding

Coding Courses

Learn HTML

Learn CSS

Learn JavaScript

Resources

C Language

C++/STL

Java

DBMS

Python

PHP

Android

Game Development

Data Structure & Alog.

Operating System

Computer Network

Computer Architecture

Docker

GO Language

GIT Guide

Linux Guide

More...

Interview Tests

Java Interview Tests

Python Interview Tests

DBMS Interview Tests

Linux Interview Tests

Aptitude Tests

GATE 2022 Tests

More...

Projects/Programs

Python Projects

C Projects

Python Programs

C Programs

C++ Programs

Java Programs