© 2014 IJIRT | Volume 1 Issue 5 | ISSN : 2349-6002

JAVA EXCEPTION HANDLING

Yogesh Bhatia, Sanjeev Verma
Department of Information Technology
Dronacharya College of Engineering, Gurgaon, Hr

***Abstract-* Exception handling is an important and difficult part of the Java programming's teaching and training. The designers of future programming languages must decide whether to include exceptions in their new languages. The paper points out and discusses the outstanding problems in details and the basic ideas of exception handling and locations need to be handled by exception handling methods in Java programs. Exceptions are frequently a controversial language feature with both language designers and programmers. In this paper we will discuss the exceptions in Java language and measures to handle and solve them independently.**

I. INTRODUCTION

Java is a general purpose Object-Oriented Programming language which was developed by Sun Microsystems, USA in 1991. Exception handling is an important and difficult part of the Java programming's teaching and training. Java exception handling enables your Java applications to handle errors sensibly. Exception handling is a very important yet often neglected aspect of writing robust Java applications or components. When an error occurs in a Java program it usually results in an exception being thrown. How you throw, catch and handle these exception matters. Exceptions represent problems or unusual situations that may occur in a program. Java provides various ways to handle exceptions when they occur, Java has a predefined set of exceptions and errors that may occur during the execution of a program, and those can be used to solve problems timely when exceptions occurs, The advantages of Java exception handling is making the program itself has the ability handling the exceptions, and separating the codes handling exceptions from the normal codes performing the required logic. A program is composed of a set of threads executing a sequence of method calls on a set of objects and classes. Objects are instances of classes, and classes are made up of data fields and methods in the Java vernacular.

There are several different ways to do so. Not all are equally efficient and fail safe.
An exception is a problem that arises during the execution of a program. An exception can occur for many different reasons, including the following:

- A user has entered invalid data.
- A file that needs to be opened cannot be found.
- A network connection has been lost in the middle of communications or the JVM has run out of memory.

To understand how exception handling works in Java, you need to understand the
three categories of exceptions:

- **Checked exceptions**: A checked exception is an exception that is typically a user error or a problem that cannot be foreseen by the programmer. For example, if a file is to be opened, but the file cannot be found, an exception occurs. These exceptions cannot simply be ignored at the time of compilation.
- **Runtime exceptions**: A runtime exception is an exception that occurs that probably could have been avoided by the programmer. As opposed to checked exceptions, runtime exceptions are ignored at the time of compilation.
- **Errors**: These are not exceptions at all, but problems that arise beyond the control of the user other programmer. Errors are typically ignored in your code because you can rarely do anything about an error. For example, if a stack overflow occurs, an error will arise. They are also ignored at the time of compilation.

II. METHODS OF SOLVING EXCEPTIONS

Java method body with a try/catch block where the catch expression's type java.lang.Throwable. The rescue clause is not equivalent to Java's finally construct. The code enclosed in a finally block is always executed when a method completes, whether it completes normally or abnormally, while a rescue clause only executes when a routine fails.

Below is an example to handling an exception using try and multiple catch case statement.

```
try
{
// Statements
}catch(ExceptionType1 e1)
{
// catch block
} catch(ExceptionType2 e2)
{
// catch block
} catch(ExceptionType3 e3)
{
// catch block
}
```

III. THE THROWS/THROW KEYWORDS

If a method does not handle a checked exception, the method must declare it using the throws keyword. The throws keyword appears at the end of a method's signature.

You can throw an exception, either a newly instantiated one or an exception that you just caught, by using the throw keyword

IV. THE FINALLY KEYWORD

The finally keyword is used to create a block of code that follows a try block. A finally block of code always executes, whether or not an exception has occurred.

Using a finally block allows you to run any cleanup-type statements that you want to execute, no matter what happens in the protected code.

A finally block appears at the end of the catch blocks

V. DECLARING YOU OWN EXCEPTION

You can create your own exceptions in Java. Keep the following points in mind when writing your own exception classes:

All exceptions must be a child of Throwable.

If you want to write a checked exception that is automatically enforced by the Handle or Declare Rule, you need to extend the Exception class.

If you want to write a runtime exception, you need to extend the RuntimeException class.

VI. CONCLUSION

Exception handling is an important and difficult part of the Java programming's teaching and training, which involving many aspects from the concept of exceptions, the main idea of exception handling to many knowledge points in exception handling mechanism. General ideas and structures of teaching exception handling are
outlined in the paper after referencing, comparing and synthesizing Java teachings of various universities and training institutions. Aimed at the outstanding problems in the teaching, the paper proposes a kind of teaching process which combines the heuristic teaching with the practical case of significance and then figures out
gradually the solving methods have to be learned in Exception handling. The teaching practice shows that the good teaching results have been achieved.

REFERENCES

[1] Research on teaching of Java Exception Handling by Xiaoyuan Zhou. *I.J. Education and Management Engineering* 2012, 9, 1-7. Published Online September 2012 in MECS (http://www.mecs-press.net)
DOI: 10.5815/ijeme.2012.09.01

[2] Exceptions in Java and Eiffel: Two Extremes in Exception Design and Application. Joseph R. Kiniry School of Computer Science and Informatics University College Dublin Belfield, Dublin 8, Irelandjoseph.kiniry@ucd.ie

[3] JAVA - EXCEPTIONS HANDLING. http://www.tutorialspoint.com/java/java_exceptions.htm

[4] Advance Java Book by Kanika Lakhani. KATSON PUBLICATIONS.

[5] Advance Java Book by Gary Cornell.