COJ :: Exception Handling

TalentSprint

Licensed To Skill

Version 1.0.4

Learning Objectives

The content in this presentation is aimed to learn the following:

- Define Exception
- Differentiate Exception and Error
- Explain Types of exceptions
- Use try-catch-finally construct
- Use throws construct

Exception Handling

Define Exception:
Exception is a run-time error which arises during the execution of java program. The term exception stands for an “exceptional event”.
What happens when an exception occur:

- Exception are typically an event or conditions that arise during the execution which interrupt the normal flow of program
- Java Provides with exception handling mechanism using try and catch
- Exception handling is used to ensure graceful termination of program

Exception Handling

Difference between Error and Exception:

Errors:

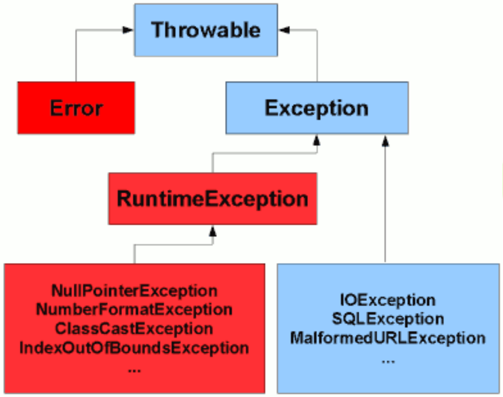
- Error indicates serious problem that a reasonable application should not try to catch
- You cannot handle Error
  Example: JVM out of memory, Stack overflow

Exception:

- Exception indicates conditions that a reasonable application might want to catch
- You can handle Exceptions
  Example: ArithmeticException

Exception Handling

Exception Hierarchy:



Exception Handling

Types of Exceptions:

There are two types of Exceptions:

1. UnChecked Exceptions: These are the exceptions that are not checked at compile time All these exceptions are sub-classes for RuntimeException class
   Example: ArithmeticException, ArrayIndexOutOfBoundsException
2. Checked Exceptions: These are the exceptions that are checked at compile time These exceptions are sub-classes of Exception class and not RuntimeException class
   Example: IOException

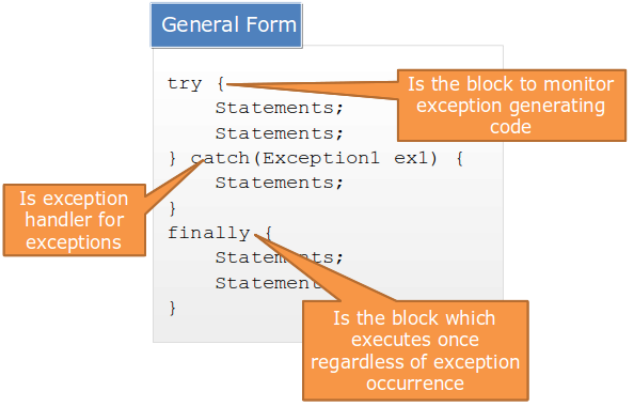
Exception Handling

Exception Constructs:

try	This block encases any statements that might cause an exception to occur.
catch	This block(s) provide a place to handle the exception thrown by the statements within a try block.
finally	The statements in the finally block are always executed. We can write resource clean up code here.
throw	Used to throw a specific exception from the program.
throws	Specifies which exceptions a given method can throw.

Exception Handling

Exception-Handling Blocks:


General Form
Is the block to monitor exception generating code
Is exception handler for exceptions
Is the block which executes once regardless of exception occurrence

```
try {
    Statements;
    Statements;
} catch(Exception1 ex1) {
    Statements;
}
finally {
    Statements;
    Statements;
}
```

Exception Handling

Program without exception handler:

```java
public class ExceptionDemo {
    public static void main(String[] args) {
        int i = 0, j = 10;
        int k = j / i;
        System.out.println("K is: " + k);
    }
}
```

Output

```
Exception in thread "main"
java.lang.ArithmeticException: / by zero at
com.ts.exceptions.ExceptionDemo.main
```

Note

The above code is without exception handling mechanism.

Exception Handling

What happens when an exception occurs and program doesn't have exception handling code:

- When an exception occurs an object of the exception type is created first.
- This object contains the information about the exception like its type, message and when it occured.
- This object is handed over to runtime system which is called as throwing an exception.
- Now the runtime system searches for a block of code that can handle the exception.

Exception Handling

Program with exception handler:

```
public class ExceptionDemo {
    public static void main(String[] args) {
        try {
            int i = 0, j = 10;
            int k = j / i;
            System.out.println("K is: " + k);
        } catch(ArithmeticException ae) {
            System.out.println(“diving by
  zero”);
        }
    }
}
```

Exception Handling

What happens when an exception occurs and program have exception handling code:

- When an exception occurs an object of the exception type is created first.
- This object contains the information about the exception like its type, message and when it occurred.
- This object is handed over to runtime system which is called as throwing an exception.

talent sprint

Exception Handling

What happens when an exception occurs and program have exception handling code:

- Now the runtime system searches for a block of code that can handle the exception. This block of code is called exception handler.
- An exception handler is considered appropriate if the type of the exception object thrown matches the type that can be handled by the handler.
- If the match is found, then the appropriate exception handler block is executed.

Exception Handling

Multiple catch blocks:

```
try {
    Statements;
    Statements;
} catch(Exception1 ex1) {
    Statements;
} catch(Exception1 ex1) {
    Statements;
}
```

- The statements within a single try block can generate different kind of exceptions
- So its a good programming practice, to write a dedicated catch block for each kind of exception

Exception Handling

The **finally** block

- The statements in the finally block are always executed
- This is a good place to write clean up code like releasing file objects etc
- The statements within finally will execute if any exception occurs or not

Syntax

```
finally {
    Statements;
}
```

Exception Handling

Program generating checked Exception:

Example

```
public class ExceptionDemo {
    public static void main(String[] args) {
        Thread.sleep(100);
    }
}
```

Exception Handling

Program generating checked Exception:
After compilation:

```
javac ExceptionDemo.java
ExceptionDemo.java:3: error: unreported
  exception InterruptedException;
  must be caught or declared to be
  thrown Thread.sleep(100);
                      ^
1 error
```

Exception Handling

throws keyword:

- used to delegate the responsibility of exception handling to the caller method
- used to throw checked exceptions out of a method

Usage of throws

```
public class ExceptionDemo {
    public static void main(String[] args)
   throws Interrupted Exception {
        Thread.sleep(100);
    }
}
```

Exception Handling

Rules to remember:

- A try block should be followed by either a catch or finally block
- Even though we have mutliple catch blocks at a time only one exception occurs and only one catch block gets executed
- All catch blocks must be ordered from most specific to most general i.e catch for ArithmeticException must come before catch for Exception
- For handling all the exceptions using single catch block do the following:

```
catch(Exception e) {
    System.out.println("For all Exceptions"); }
```

Exception Handling