Java - Exception Handling

Md. Mohsin Uddin

East West University

mmuddin@ewubd.edu

July 13, 2019

Runtime Exceptions or Unchecked exceptions : ArrayIndexOutOfBoundsException

```
package exceptions;
public class UncheckedDemo {
    public static void main(String args[]) {
      int num[] = {1, 2, 3, 4};
      System.out.println(num[5]);
   }
}
```

When the above code is compiled and executed, it produces the following result:

```
Exception in thread "main" java.lang.ArrayIndexOutOfBoundsException: 5
at exceptions.UncheckedDemo.main(UncheckedDemo.java:5)
Java Result: 1
```

Runtime Exceptions : FileNotFoundException

```
package exceptions;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
public class FileNotFoundDemo {
  public static void main(String args[])
        throws FileNotFoundException{
      File file = new File("file.txt");
      FileReader fr = new FileReader(file);
   }
}
```

When the above code is compiled and executed, it produces the following result:

```
Exception in thread "main" java.io.FileNotFoundException: file.txt (No such file or directory)
at java.io.FileInputStream.open0(Native Method)
at java.io.FileInputStream.open(FileInputStream.java:195)
at java.io.FileInputStream.¡init¿(FileInputStream.java:138)
at java.io.FileReader.¡init¿(FileReader.java:72)
at exceptions.FileNotFoundDemo.main(FileNotFoundDemo.java:10)
Java Result: 1
```

try-catch : toString()

```
package exceptions;
public class UncheckedDemo {
    public static void main(String args[]) {
        try{
            int num[] = {1, 2, 3, 4};
            System.out.println(num[5]);
        }
        catch(Exception e) {
            System.out.println("Exception Info: "+e.toString());
        }
    }
}
```

When the above code is compiled and executed, it produces the following result:

```
Exception Info: java.lang.ArrayIndexOutOfBoundsException: 5
```

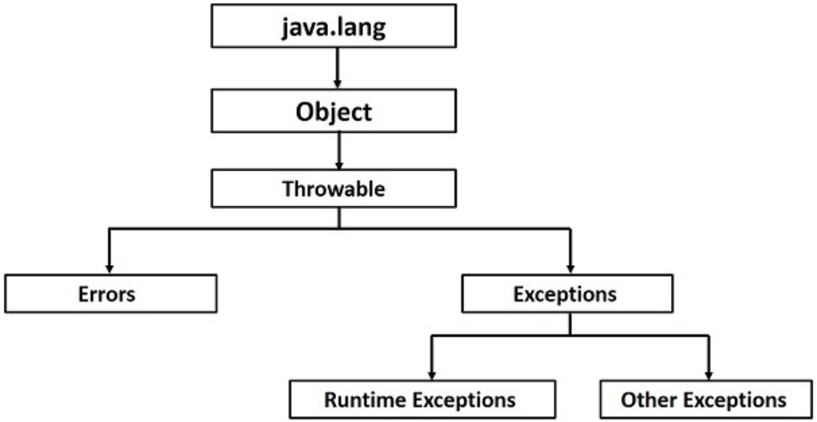
try-catch : printStackTrace()

```
package exceptions;
public class UncheckedDemo {
    public static void main(String args[]) {
        try{
            int num[] = {1, 2, 3, 4};
            System.out.println(num[5]);
        }
        catch(Exception e) {
            e.printStackTrace();
        }
    }
}
```

When the above code is compiled and executed, it produces the following result:

```
java.lang.ArrayIndexOutOfBoundsException: 5
at exceptions.UncheckedDemo.main(UncheckedDemo.java:6)
```

Exception Hierarchy



Multiple catch : Part I

```
package exceptions;
import java.io.File;
import java.io.FileNotFoundException;
import java.io.FileReader;
public class MultipleCatch {
    public static void main(String[] args) {
        try {
            File file = new File("file.txt");
            FileReader fr = new FileReader(file);
            int num[] = {1, 2, 3, 4};
            System.out.println(num[5]);
        }
        catch (FileNotFoundException f) {
            System.out.println(""+f.getMessage());
        }
        catch (ArrayIndexOutOfBoundsException e) {
            System.out.println(""+e.getMessage());
        }
        catch (Exception e) {
```

Multiple catch : Part II

```
                System.out.println(""+e.getMessage());
            }
        }
}
```

The Finally Block

```
public class ExcepTest {
    public static void main(String args[]) {
        int a[] = new int[2];
        try {
            System.out.println("Access element three :" + a[3]);
        } catch (ArrayIndexOutOfBoundsException e) {
            System.out.println("Exception thrown  :" + e);
        }finally {
            a[0] = 6;
            System.out.println("First element value: " + a[0]);
            System.out.println("finally....");
        }
    }
}
```

When the above code is compiled and executed, it produces the following result:

```
Exception thrown :java.lang.ArrayIndexOutOfBoundsException: 3
First element value: 6
The finally statement is executed
```

User-defined Exceptions : Part I

```
package exceptions;
class InsufficientFundsException extends Exception {
    private double amount;

    public InsufficientFundsException(double amount) {
        this.amount = amount;
    }

    public double getAmount() {
        return amount;
    }
}

class CheckingAccount {
    private double balance;
    private int number;

    public CheckingAccount(int number) {
        this.number = number;
```

User-defined Exceptions : Part II

```
}

public void deposit(double amount) {
    balance += amount;
}

public void withdraw(double amount)
        throws InsufficientFundsException {
    if(amount <= balance) {
        balance -= amount;
    }else {
        double needs = amount - balance;
        throw new InsufficientFundsException(needs);
    }
}

public double getBalance() {
    return balance;
}
```

User-defined Exceptions : Part III

```
    public int getNumber() {
        return number;
    }
}


class BankDemo {
    public static void main(String [] args){
        CheckingAccount c = new CheckingAccount(101);
        System.out.println("Depositing␣$500...");
        c.deposit(500.00);
        try {
            System.out.println("\nWithdrawing␣$100...");
            c.withdraw(100.00);
            System.out.println("\nWithdrawing␣$600...");
            c.withdraw(600.00);
        } catch (InsufficientFundsException e) {
            System.out.println
```

User-defined Exceptions : Part IV

```
                ("Sorry, but you are short $" + e.getAmount());
            e.printStackTrace();
        }
    }
}
```

When the above code is compiled and executed, it produces the following result:

```
Depositing $500...
Withdrawing $100...
Withdrawing $600...
Sorry, but you are short $200.0
InsufficientFundsException
at CheckingAccount.withdraw(CheckingAccount.java:25)
at BankDemo.main(BankDemo.java:13)
```

The try-with-resources : automatic resource management

```
package exceptions;
import java.io.FileReader;
import java.io.IOException;

public class TryResources {
  public static void main(String args[]) {
    try(FileReader fr = new FileReader("file.txt")){
      char [] a = new char[50];
      fr.read(a);
      for(char c : a)
      System.out.print(c);
    } catch (IOException e) {
      e.printStackTrace();
    }
  }
}
```

References

DEITEL, Java How to Program, 11/e

Java: the complete reference, Herbert Schildt, McGraw-Hill Education Group