USER DEFINED EXCEPTION IN JAVA

Java allows the user **to create their own exception class** which is derived from built-in class Exception. The Exception class inherits all the methods from the class Throwable. The Throwable class is the superclass of all errors and exceptions in the Java language. It contains a snapshot of the execution stack of its thread at the time it was created. It can also contain a message string that gives more information about the error.

- The Exception class is defined in **java.lang package.**
- User defined exception class must inherit Exception class.
- The user defined exception can be thrown using throw keyword.

Syntax:

class User_defined_name extends Exception{

………..

}

Some of the methods defined by Throwable are shown in below table.

Methods	**Descriptio n**
Throwable fillInStackTrace( )	Fills in the execution stack trace and returns a Throwable object.
String getLocalizedMessage()	Returns a localized description of the exception.
String getMessage()	Returns a description of the exception.
void printStackTrace( )	Displays the stack trace.
String toString( )	Returns a String object containing a description of the Exception.
StackTraceElement[ ]get StackTrace( )	Returns an array that contains the stack trace, one element at a time, as an array of StackTraceEle- ment.

two commonly used constructors of Exception class are:

- Exception() - Constructs a new exception with null as its detail message.
- Exception(String message) - Constructs a new exception with the specified detail message.

Example:

//creating a user-defined exception class derived from Exception class

public class MyException extends Exception

{

public String toString(){   // overriding toString() method

return "User-Defined Exception";

}

public static void main(String args[]){

```
            MyException obj= new MyException();
            try
            {
                    throw new MyException();    // customized exception is raised
        }
            catch(MyException e)
            {
              System.out.println("Exception handled - "+ e);
            }
    }
    }
```

Sample Output:

```
Exception handled - User-Defined Exception
```

In the above example, a custom defined exception class MyException is created by inher- iting it from Exception class. The toString() method is overridden to display the customized method on catch. The MyException is raised using the throw keyword.

Example:

Program to create user defined exception that test for odd numbers.

```
import java.util.Scanner;
class OddNumberException extends Exception
{
    OddNumberException()      //default constructor
    {
       super("Odd number exception");
    }
    OddNumberException(String msg) //parameterized constructor
    {
       super(msg);
    }
}
public class UserdefinedExceptionDemo{
    public static void main(String[] args)
    {
       int num;
       Scanner Sc = new Scanner(System.in); // create Scanner object to read
```

```
            input System.out.println("Enter a number : ");
            num = Integer.parseInt(Sc.nextLine());
            try
            {
               if(num%2 != 0) // test for odd number
                  throw(new OddNumberException()); // raise the exception if number is odd
               else
                   System.out.println(num + " is an even number");
            }
            catch(OddNumberException Ex)
            {
               System.out.print("\n\tError : " + Ex.getMessage());
            }
         }
      }
```

Sample Output1:

```
Enter a number : 11
Error : Odd number exception
```

Sample Output2:

```
10 is an even number
```

Odd Number Exception class is derived from the Exception class. To implement user defined exception we need to throw an exception object explicitly. In the above example, If the value of num variable is odd, then the throw keyword will raise the user defined exception and the catch block will get execute.