13-5-2020

MARCH

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

APRIL

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30		

13th week
087-278

28 Saturday

Another keywords used in Exception handling in Java are — **throw** and **throws**

throw Keyword — It is used to explicitly throw an exception.

```
throw new ArithmeticException("Message");
```

this can be ~~catch~~ followed by catch statement.

→ After throw block, any statement cannot be written because that should be unreachable code type error.

Example :— demo.java

```
class demo
{
    static void valid(int i)
    {
        if(i<18)
        {
        throw new ArithmeticException("Not Valid");
        }
    else{
    System.out.println("Welcome");
        }
    }
public static void main(String []args)
{   try{ valid(Integer.parseInt(args[0]));
    }
```

29 Sunday

Notes

```
        catch(ArithmeticException e)
        {
        System.out.println(e.toString());
        }
    }
}
```

Compile & Run as follows :-

javac demo.java ↵

java demo 12 ↵

Output : java.lang.ArithmeticException: Not Valid

java demo 19 ↵

Output: java.lang.ArithmeticException: Welcome

In above program we have to call function valid() inside try block since this function throws an exception and after that catch block should be there to handle if exception object is thrown to handle it.

throws keyword — throws keyword is used to declare the exception. It provide information to the programmer that there may occur an exception so during call of that method, programmer must use exception handling mechanism.

→ In other words, throws keyword is used to propagate checked exceptions.

throws keyword is used to declare a function that can throw exception while execution, so any other programmer trying to call that function should handle exception by using exception handling mechanism.

Example:- myex.java

```
class myexception extends Exception
{
    myexception(String s)
    {
        super(s);
    }
}
class myex
{
    static void valid(int i) throws myexception
    {
        if(i < 18)
        {
            throw new myexception("Not Valid");
        }else{
            System.out.println("Welcome to Vote");
        }
    }
public static void main(String args[])
{
    try { valid(Integer.parseInt(args[0]));
    }
    catch(myexception my){
    System.out.println(my);}
```

System.out.println("Testing Complete");
}
} // close of class myexe

Compile & Run as :- javac myex.java ↵
java myex 19 ↵
o/p:- Welcome to Vole
Testing Complete.

java myex 12 ↵
o/p myException: Not Valid
Testing Complete

Second output exception name is myexception which is our defined exception class.

Important point regarding — propagation

Can a RuntimeException can be propagated without using throws keyword ? — NO
Let us see with an example :- excep1.java

```
class myexception extends RuntimeException
{
    myexception(String s)
    {
        super(s);
    }
}
```

Notes

Note:- If myexception extends from Exception then we have to use throws myexception in valid method.


```
class excep1
{
    static void valid(int age) throws myexception
    {
        if(age < 18)
        {
            throw new myexception("Not Valid to Vote.");
        }
        else {
            System.out.println("Welcome to Vote.");
        }
    }
    public static void main(String args[])
    { try { valid(Integer.parseInt(args[0]));
        }
        catch(myexception my)
        {
            System.out.println(my)
        }
        System.out.println("Testing Complete.");
    }
} //close of class excep1
```

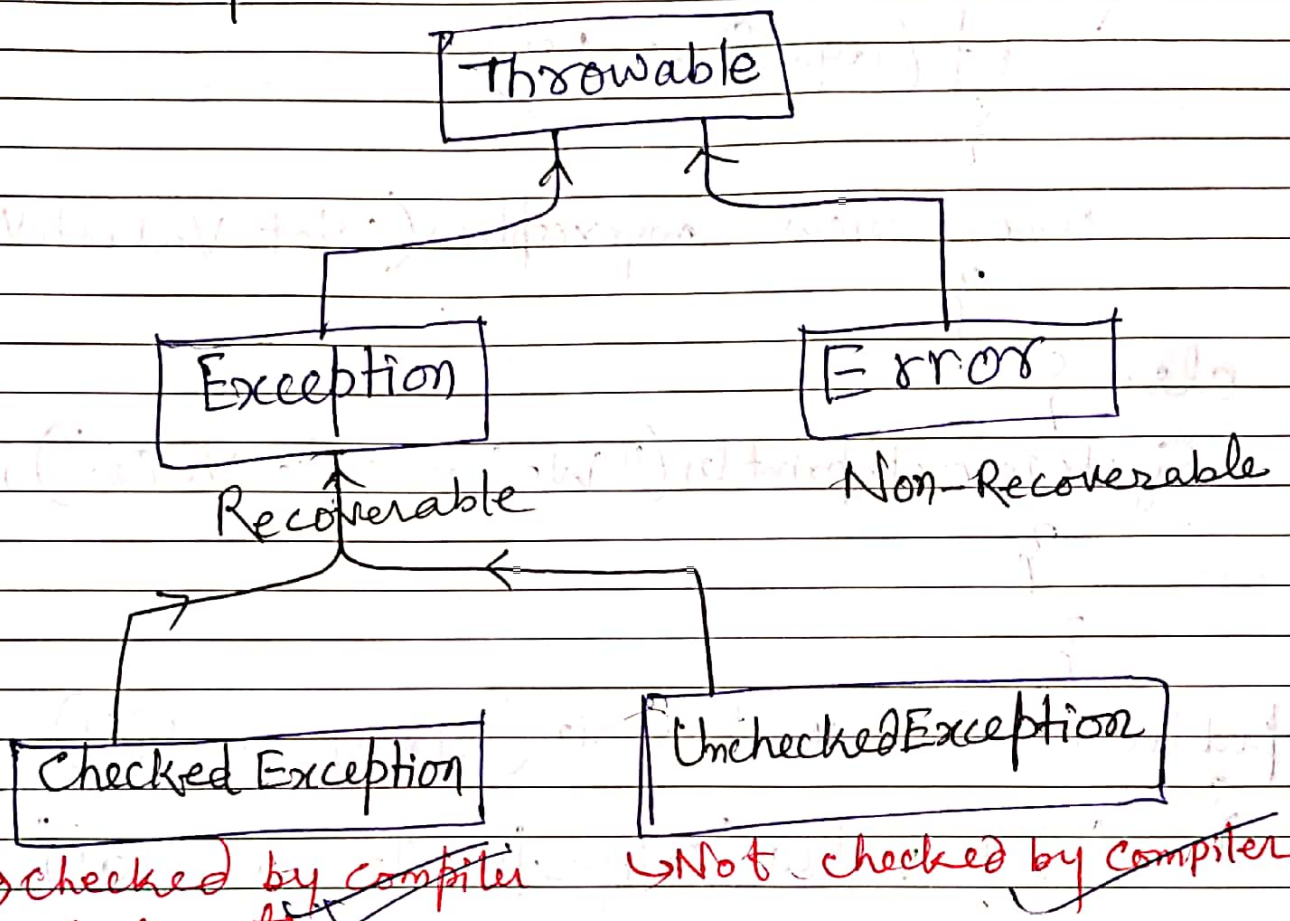
Compile & Run it & see.

14th week
093-272

03 April
Friday

The Organization of Exception classes in Java:—

Throwable class is the root of all the classes for entire Java Exception Handling.



Exception types and the differences between them:—

Checked Exception	Unchecked Exception
(i) Compiler checks	(i) Compiler Doesn't check
(ii) Compile time exception	(ii) Runtime Exceptions
(iii) Except "RuntimeException" class, all are checked.	(iii) "RuntimeException" and its child class are unchecked.
eg:- FileNotFoundException, InterruptedException,	eg:- ArithmeticException NullPointerException

Checked Exception

ClassNotFoundException,
NoSuchMethodException
NoSuchFieldException

Unchecked Exception

ArrayIndexOutOfBoundsException,
EmptyStackException.

04
Saturday

END