

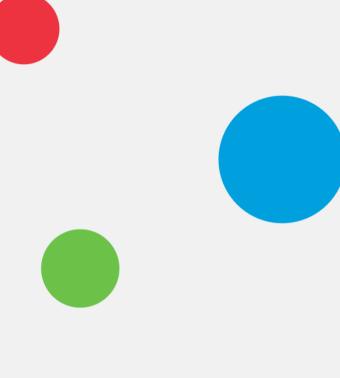
Throws Clause

Agenda



Throws Clause

Throws Clause





Using throws

- Sometimes, a method is capable of causing an exception that it does not handle
- Then, it must specify this behavior so that callers of the method can guard themselves against that exception
- While declaring such methods, you have to specify what type of exception it may throw by using the throws keyword
- A **throws** clause specifies a comma-separated list of exception types that a method might throw:

Sensitivity: Internal & Restricted

type method-name(parameter list) throws exception-list

Using throws (Contd.).

```
class ThrowsDemo{
 static void throwOne(){
   System.out.println("Inside throwOne.");
   throw new FileNotFoundException();
 public static void main(String args[]) {
   throwOne();
```

What happens when this code is compiled ?

Compilation Error.....why?

Implementing throws

```
import java.io.*;
class ThrowsDemo{
 static void throwOne() throws FileNotFoundException{
   System.out.println("Inside throwOne.");
   throw new FileNotFoundException();
 public static void main(String args[]) {
   try{
       throwOne();
   catch (FileNotFoundException e) {
       System.out.println("Caught " + e);
```

Rule governing overriding method with throws

 The overriding method must NOT throw checked exceptions that are new or broader than those declared by the overridden method

For eg: A method that declares(throws) an SQLException cannot be overriden by a method that declares an IOException, Exception or any other exception unless it is a subclass of SQLException

- In other words, if a method declares to throw a given exception, the overriding method in a subclass can only declare to throw the same exception or its subclass
- This rule does not apply for unchecked exceptions

Quiz

 What will be the result, if we try to compile the following code (FileNotFoundException is a subclass of IOException)

```
import java.io.*;
class Super {
 void m1() throws FileNotFoundException {
   FileInputStream fx = new FileInputStream("Super.txt");
                               Yes, it will throw compilation Error
class Sub extends Super {
 void m1() throws IOException {
   FileInputStream fx = new FileInputStream("Sub.txt");
```

 What will be the result, if we try to compile the following code (FileNotFoundException is a subclass of IOException)

```
import java.io.*;
class Super {
 void m1() throws IOException {
   FileInputStream fx = new FileInputStream("Super.txt");
                                         No Error!
                                         Compilation successful
class Sub extends Super {
 void m1() throws FileNotFoundException {
   FileInputStream fx = new FileInputStream("Sub.txt");
} 0
```

confidential

• What will be the result, if we try to compile the following code

```
class Super {
 void m1() throws ArithmeticException {
   int x = 100, y=0;
       int z=x/y;
       System.out.println(z);
                                      No Error!
                                      Compilation successful
class Sub extends Super {
 void m1() throws NumberFormatException {
   System.out.println("Wipro");
```

confidential

- What will be the result, if we try to compile the following code
- (FileNotFoundException & SQLException are not related hierarchically)

```
import java.io.*;
import java.sql.*;
class Super {
  void m1() throws FileNotFoundException {
    FileInputStream fx = new FileInputStream("Super.txt");
                                        It will throw compilation Error. Why?
class Sub extends Super {
  void m1() throws SQLException {
    FileInputStream fx = new FileInputStream("Sub.txt");
```

Sensitivity: Internal & Restricted

What will be the result, if we try to compile and execute the following code

```
import java.io.*;
class Plane {
    public Plane() throws IOException, RuntimeException {
    System.out.println("Plane");
class Jet extends Plane {    // empty class
public class Tester {
    public static void main(String args[]) throws IOException {
    new Plane();
```

It will throw compilation Error. Why?

Summary

In this session, you were able to:

Learn about throws clause





