VIT-AP UNIVERSITY

Course code : **CSE2005**

Course title : **Object Oriented Programming**

Exception Handling – Throws Clause

Objectives

This session will give the knowledge about

- Throws Clause
- Throw 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:
  - Return-type method-name(parameter list) throws exception-list

Using throws

```
public class Main {
        static void method(){
                FileInputStream obj=new FileInputStream("a.txt");
        }
        public static void main(String args[]) {
                method();
        }
}
```

Why this code shows compile time error?

Using throws

```
public class Main {
        static void method() throws FileNotFoundException{
                FileInputStream obj=new FileInputStream("a.txt");
        }
        public static void main(String args[]) {
                try {
                        method();
                } catch (FileNotFoundException e) {
                        e.printStackTrace();
                }
        }
}
```

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

Overriding method with throws

```
class Base{
        static void method() throws FileNotFoundException{
                FileInputStream obj=new FileInputStream("a.txt");
        }
}
class Derived extends Base{
        static void method() throws IOException{
                FileInputStream obj=new FileInputStream("b.txt");
        }
}
```

This will throw a compile time error

Overriding method with throws

```
class Base{
        static void method() throws IOException {
                FileInputStream obj=new FileInputStream("a.txt");
        }
}
class Derived extends Base{
        static void method() throws FileNotFoundException {
                FileInputStream obj=new FileInputStream("b.txt");
        }
}
```

Overriding method with throws

```
class Base{
        static void method() throws ArithmeticException{
                int x=10/0;
        }
}
class Derived extends Base{
        static void method() throws RuntimeException{
                int x= Integer.parseInt("23.34");
        }
}
```

Quiz

```
class Base{
        static void method() throws FileNotFoundException {
                FileInputStream obj=new FileInputStream("a.txt");
        }
}
class Derived extends Base{
        static void method() throws SQLException {
                FileInputStream obj=new FileInputStream("b.txt");
        }
}
```

Quiz

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

VIT-AP UNIVERSITY

Course code : **CSE2005**

Course title : **Object Oriented Programming**

Exception Handling – Throw Clause

Using throw

- System-generated exceptions are thrown automatically
- At times you may want to throw the exceptions explicitly which can be done using the throw keyword. The exception-handler is also in the same block
- The general form of throw is:
  - throw ThrowableInstance
- Here, ThrowableInstance must be an object of type Throwable, or a subclass of Throwable

Using throw

```
public class Main {
        public static void main(String args[]) {
                try {
                        int age = 17;
                        if (age < 18)
                                throw new ArithmeticException();
                } catch (ArithmeticException e) {
                        System.out.println("age is less than 18");
                }
        }
}
```

Quiz

```
public class Main {
        void method() throws IOException {
                throw new IOException("Method Error");
        }
        public static void main(String args[]) {
                Main m = new Main();
                m.method();
                System.out.println("main ends...");
        }
}
```

Match the following:

Match the content of Column A with the most appropriate content from column B :

Column A	Column B
a) An exception is	a) Used to throw an exception explicitly
b) Throwable	b) Caused by Dividing an integer by zero
c) ArithmeticException is	c) An event that can disrupt the normal flow of instructions
d) Catch Block	d) This class is at the top of exception hierarchy
e) “throw” is	e) Exception Handler

Summary

We have discussed about

- Throws Clause
- Throw Clause