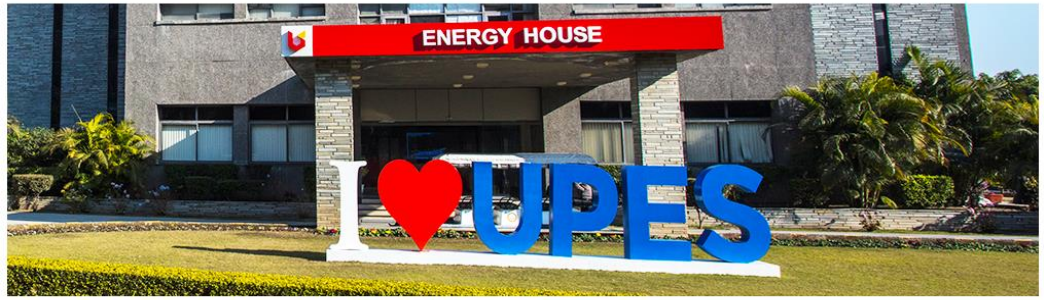


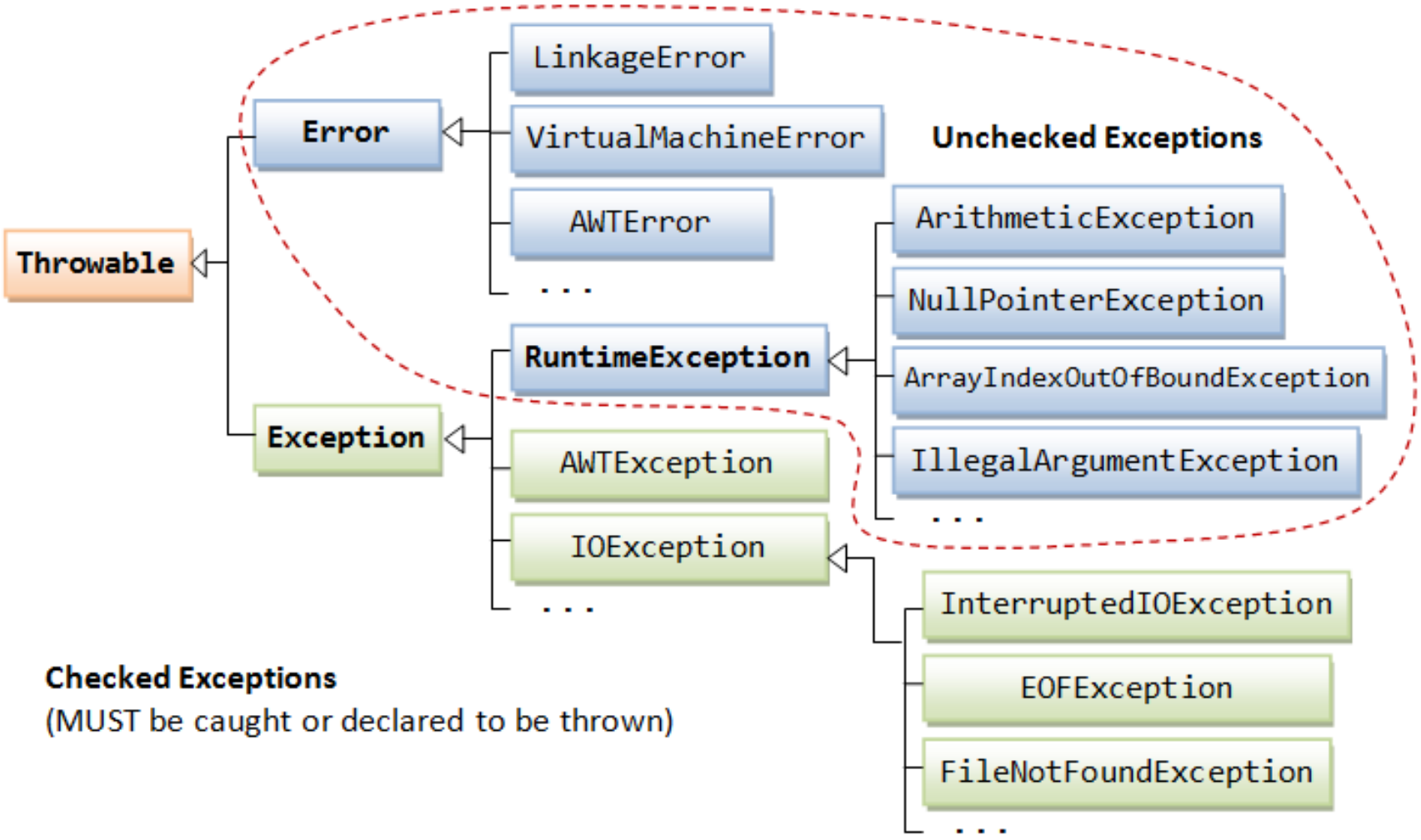
UPES

UNIVERSITY WITH A PURPOSE

Object Oriented Programming

UPES
UNIVERSITY WITH A PURPOSE

Exception Types



Source: https://www.ntu.edu.sg/home/ehchua/programming/java/J5a_ExceptionAssert.html

UPES
UNIVERSITY WITH A PURPOSE

Exception Types

- Class Exception represents Exceptions that a program faces due to abnormal or special condition during execution.
- Exception can be of two types:
  - Checked Exceptions
  - Unchecked Exceptions
- Unchecked Exceptions are `RuntimeException` and any of its sub classes.

Exception Handling

```
try
{ //block of code
}
catch (<ExceptionType> <parameter>){
//block of code
}
finally{
//block of code
}
```

Exception Handling

2. Default throw and user defined catch

```
Class ExceptionDemo4{
Public static void main(String [] args){
try{
        System.out.println(10/0);
        System.out.println("In try block");
        }
Catch(ArithmeticException e){
        System.out.println("Exception is:"+e.getMessage());
        }
System.out.println("After Exception");
}
}
```

Points to Remember

Note:

1. For each try block there may be zero or more catch block but only one finally block.
2. There may be multiple catch block with one try block.
3. A try block must be followed by either at least one catch block or one finally block.
4. The order exceptions handled in the catch block must be from the most specific exception to the most generous one if the exceptions have parent-child relationship.

Ex: *`FileNotFoundException`* must be caught before *`IOException`*

5. Order is not important if exceptions do not have parent-child relationship.
6. In case of default catch mechanism, the program will get terminated after displaying the error message.
7. If no catch block matches with the Exception type then finally will get executed if present and then the default catch mechanism will work.
8. Finally block will get executed every time even if no exception raised inside the try block

Exception Handling

3. User-defined throw and default catch

- A program can explicitly throw an exception using the throw statement besides the implicit exception thrown.

```
throw <throwable Instance> ;
```

- The exception reference must be of type Throwable class or one of its subclasses.
- A detailed message can be passed to the constructor when the exception object is created.

Exception Handling

3. User-defined throw and default catch

```
class ExceptionDemo5{
public static void main(String [] args){
int balance=5000;
int withdrawlAmt=2000;
if (balance<withdrawlAmt)
        throw new ArithmeticException("Insufficient balance in the account");
balance= balance-withdrawlAmt;
System.out.println("Tranaction Successful");
System.out.println("Thanks!");
}
}
```

Exception Handling

4. User-defined throw and user-defined catch

```
class ExceptionDemo6{
public static void main(String [] args){
int balance=5000;
int withdrawlAmt=6000;
try{
        if (balance<withdrawlAmt)
                throw new ArithmeticException("Insufficient balance in the account");
        balance= balance-withdrawlAmt;
        System.out.println("Tranaction Successful");
        }
catch(ArithmeticException e){
        System.out.println("Exception: "+e.getMessage());
        }
System.out.println("Thanks!"); } }
```

References

Schildt, H. (2014). *Java: the complete reference*. McGraw-Hill Education Group.