Java 프로그래밍 길잡이

원리를 알면 IT가 맛있다

Java Programming for Beginners

chapter 09.

예외 처리

❑ 학습목표

- 자바의 예외 및 예외처리에 관하여 학습한다.
- 자바의 예외 클래스 계층구조에 관하여 학습한다.
- Compile checked 예외와 compile unChecked 예외에 관하여 학습한다.
- try~catch 및 throws을 이용한 예외처리 방법에 관하여 학습한다.
- 사용자 정의 예외클래스 생성 및 사용법에 관하여 학습한다.

❑ 1) 예외 [exception]

❍ 예외 (exception)

프로그램 실행 중에 발생되는 의도하지 않은 문제발생 을 의미한다.

예외가 발생되면 프로그램은 비정상종료 된다.

❍ 예외처리 (exception handling)

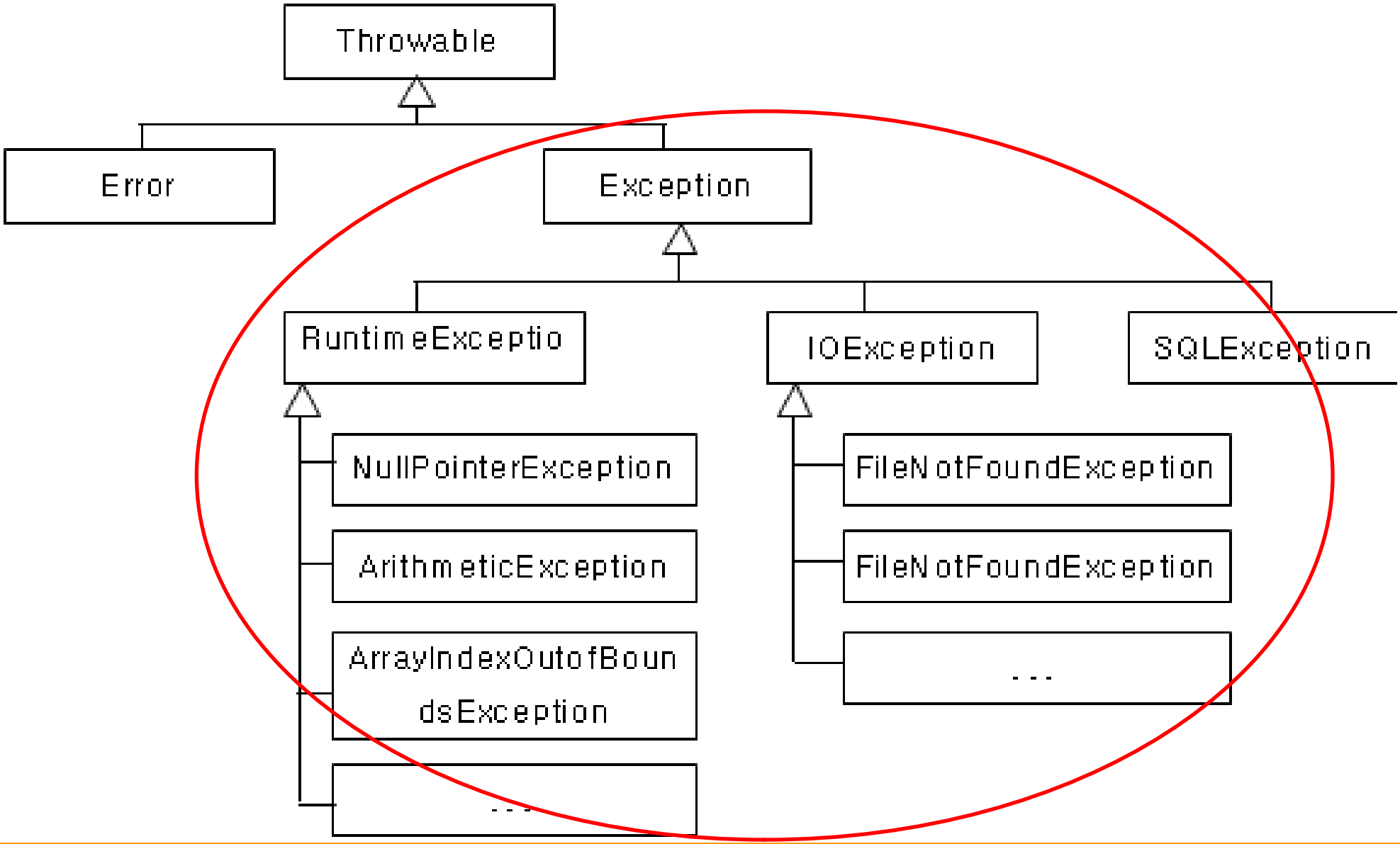
- 예외발생시 비정상 종료 되는 프로그램을 '정상종료'로 처리하는 작업을 의미한다.
- 예외처리를 담당하는 예외클래스가 제공된다.
- 개발자가 할 수 있는 최대한의 예외처리는 다음과 같다.

가. 최대한 자세하게 end-user에게 예외가 발생된 이유를 출력한다.

나. 최대한 end-user가 이해하기 쉬운 언어로 출력한다.

(예외가 발생된 코드를 수정하는 것이 예외처리가 아니다. 이것은 불가능하다. 이유는 순차문이기 때문이다. 한번 실행된 문장은 다시 실행시킬수 없다.)

❑ 2) 예외클래스 계층구조


Throwable
Error
Exception
RuntimeExceptio
IOException
SQLException
NullPointerException
ArithmeticException
ArrayIndexOutofBoun
dsException
. . .
FileNotFoundException
FileNotFoundException
. . .

❑ 3) 예외종류 2가지

가. Compile checked 예외

- 컴파일시 예외처리 여부를 컴파일러가 체크한다. 예외처리가 안되어 있으면 컴파일 에러가 발생된다.
- IOException 과 SQLExceptino 계열

➔ 자바I/O 및 데이터베이스 관련 작업을 수행하는 메소드를 사용하기 위해서는 반드시 예외처리를 해야 된다.

나. Compile unchecked 예외

- 컴파일시 예외처리 여부를 컴파일러가 체크하지 않는다.

컴파일러가 체크하지 않는 이유는 개발자가 코드를 잘못 작성했기 때문에 발생된 예외이기 때문이다.

- RuntimeExceptions계열

❑ 4) 예외처리 방법 2가지

가. try~ catch 문 사용

예외가 발생된 코드 내에서 예외를 처리하는 방법이다.

문법:

```
try{
    문장1;
    문장2;
}catch( 예외클래스명  e){
    예외처리코드;
}
```

*예외처리 코드에서 사용가능한 메소드
e.getMessage()
e.printStackTrace()

나. throws 문 사용

예외가 발생된 코드가 아닌 호출한 메소드로 예외를 떠 넘기는 방식이다.
호출한 메소드에서는 try~catch문을 사용하여 예외처리를 해야 된다.
최종적으로 , main 메소드로 떠 넘기게 된다.

*다중 catch 문 사용

try 블록 내에서 실행되는 문장이 여러 개 있는 경우에 , 발생되는 예외가 달라질 수 있다. 이런 경우에 사용 가능하다. (다형성 적용 가능)

주의할 점은 반드시 계층구조가 낮은 클래스부터 catch해야 된다.

문법:

```
try{
    문장1;
    문장2;
}catch( 예외클래스명  변수명){
    예외처리코드;
}catch( 예외클래스명  변수명2){
    예외처리코드;
 }
```

* finally 문

예외 발생 여부와 상관없이 항상 실행되어야 하는 문장을 지정한다.

문법:

```
try{
    문장1;
    문장2;
}catch( 예외클래스명  변수명){
    예외처리코드;
}catch( 예외클래스명  변수명2){
    예외처리코드;
}finally{
    반드시 수행되는 문장
}
```

* try ~ finally 문

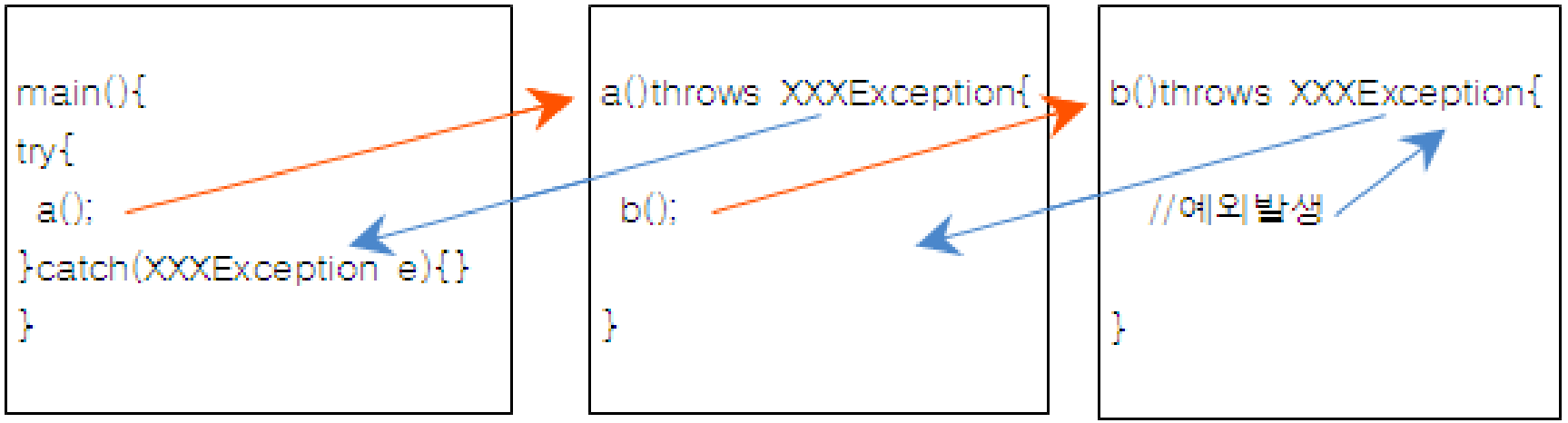
```
try{

}finally{

}
```

- throws 문을 이용한 예외처리 방법.

예외가 발생되면 호출한 메소드로 예외를 떠 넘기는 방식이다.

문법:

[지정자] 리턴타입 메소드명 throws 예외클래스, 예외클래스2{ }



- throws 문을 이용해서 예외 처리하는 경우.

가. 사용자가 지정한 특정조건에 위배될 경우에 예외를 명시적으로 발생시켜 처리하는 경우이다.

나. 발생된 예외클래스 대신에 사용자가 만든 예외클래스로 처리하기 위해서 이다.

❑ 5) 명시적 예외발생

- throw 을 이용한 명시적 예외 발생

사용자가 만든 어플리케이션에서 특정 조건을 위반했을 경우에 명시적으로 예외를 발생시키는 방법이다.

문법:

```
if( 조건 ){
    throw  new  예외클래스("예외처리문자열");
}
```

실제 예>

```
public  void  xxx ()  throws  예외클래스{
        if( 조건식){
            throw  new  예외클래스("문자열");
        }
}
```

❑ 6] 사용자 정의 예외 클래스

사용자가 만든 어플리케이션에서 특정 조건을 위반했을 경우에 명시적으로 예외를 발생 시킬수 있다. 이때 발생시킨 예외를 처리할 수 있는 예외클래스는 API에서 제공해주지 못한다.

이유는 어떤 상황이 예외상황인지를 모두 파악할 수 없다. 따라서 API는 포괄적인 예외클래스를 제공하며, detail한 예외, 즉 사용자가 발생시킨 예외등은 사용자가 명시적으로 예외 클래스를 만들어 처리하는 것이 적합하다.

문법:

```
public class 클래스명 extends Exception{
    public 클래스명( Sring mesg){
      super(mesg);
    }
}
```

Thank you