03-8. 예외처리

Java 객체지향

예외처리(Exception handling) - 개념

- 예외(Exception)
  - 프로그램이 실행되는 동안 발생할 수 있는 비정상적인 조건
  - 컴파일시의 에러가 아닌 실행시의 에러를 예외라 함

- 자바에서의 예외처리
  - 예외처리를 위한 Exception 클래스 정의
  - 기본적인 예외는 자바에 미리 정의된 예외를 통해 처리 가능
  - 사용자가 필요한 예외를 직접 정의할 수 있음
  - 예상되는 예외는 미리 처리해주면 무조건적인 프로그램의 종료를 피할 수 있음
  - 예외처리의 사용은 프로그램의 신뢰성을 높여줌

예외처리 - 예제

```
public class ExceptionTest {
  public static void main(String[] args) {
        int a = 0;
        double b;
        b = 100/a; //java.lang.ArithmeticException 발생
        System.out.println("Some more codes"); //예외 발생으로 수행되지 않음
  }
}
```

예외처리 방법

- try/catch/finally 구문 이용
- 프로그램에서 예외가 발생했는지 검사하고 발생된 예외를 처리하기 위한 문법적 구조
- try 블록에서 예외가 발생하면 더 이상 try 블록 내부의 문장을 수행하지 않고 catch 블록으로 프로그램의 흐름이 변경됨
- 예외가 발생하면 해당 예외에 대한 객체가 생성되고 catch 블록에서 예외 객체를 참조할 수 있음
- 하나의 try 블록에서 여러 종류의 예외가 발생할 수 있는 경우 각 예외 별로 catch 블록을 할당할 수 있음
- finally 블록은 예외의 발생과 상관없이 항상 실행되는 코드가 위치하도록 하며 생략할 수 있음

예외처리 방법 - 예제

```
public class ExceptionTest {
  public static void main(String[] args) {
        int a = 0;
        double b;
        try {
                b = 100/a;
                System.out.println("Some more codes in try block");
        }catch(ArithmeticException e) {
                System.out.println("Exception occurred : "+e);
        }catch(IOException e2) {
                System.out.println("One more catch block");
        }finally {
                System.out.println("Some more codes in finally block");
        }
  }
}
```

예외 발생의 예와 종류(1)

- 예외의 구분
  - Checked Exception : 컴파일 할 때 확인 되는 예외로 예외처리가 필요함
  - Runtime Exception : 실행시점에 확인되는 예외로 예외처리를 하지 않아도 컴파일 됨

- 예외처리가 유용한 경우
  - 배열과 연관된 for 반복문
    - 배열 참조값이 배열의 크기를 벗어난 경우 예외 발생
  - 파일을 다루는 경우
    - 해당 파일이 존재하지 않거나
    - 다른 프로세스에 의해 사용중인 경우 예외 발생
  - 입출력을 다루는 경우
    - 이미 닫힌 입출력 스트림에 대해 작업하려 할 경우 예외 발생

예외 발생의 예와 종류(2)

예외의 종류	발생하는 경우
ArithmeticException	0으로 나누거나 0으로 나눈 나머지를 구하려 할 경우 발생하는 예외
NullPointerException	객체가 할당되지 않은 레퍼런스를 통하여 멤버 변수나 멤버 메쏘드에 접근하려할 경우 발생하는 예외
IOException	올바른 입출력 동작이 아닐 경우 발생하는 예외
FileNotFoundException	읽거나 쓰고자 하는 파일이 존재하지 않거나 사용 가능하지 않은 경우 발생하는 예외
ArrayIndexOutOfBoundsException	배열의 참조가 배열의 크기를 벗어난 경우 발생하는 예외

위험요소가 있는 메쏘드

- 메쏘드를 정의할 때 메쏘드의 내부에서 예외가 발생할 가능성이 있을 경우
  - try/catch 문으로 예외를 직접 처리하거나
  - 해당 메쏘드를 호출하는 메쏘드에서 예외를 처리하도록 명시할 수 있음
- throws 키워드를 사용하여 예외의 종류를 적어줌

```
public class ThrowsText {
  public void SuspiciousMethod() throws IOException, FileNotFoundException
  {
        throw new IOException(); //강제로 예외 발생
  }
}
```

사용자 정의 예외

- 예외의 최상위 클래스인 Exception 클래스를 상속받아 새로운 예외를 정의할 수 있음
- 일반적으로 생성자만 구현

```
public class DivideByZeroException extends Exception {
  public DivideByZeroException()
  {
        super("Dividing by 0");
  }
  public DivideByZeroException(String msg)
  {
        super(msg);
  }
}
```

감사합니다

단단히 마음먹고 떠난 사람은
산꼭대기에 도착할 수 있다.
산은 올라가는 사람에게만 정복된다.

윌리엄 셰익스피어
William Shakespeare