예외와 예외처리 + 입출력 스트림

작성자: 신채린

오류란 무엇인가?

- **컴파일 오류**: 프로그램 코드 작성 중 발생하는 문법적 오류

- **실행 오류**: 실행 중인 프로그램이 의도하지 않은 동작을 하거나 (bug) 프로그램이 중지되는 오류(runtime error)
- 자바는 예외 처리를 통하여 프로그램의 비정상 종료를 막고 **log를 남길 수 있음**

오류와 예외 클래스

- 시스템 오류 (error)
  - 가상 머신에서 발생
  - 프로그래머가 처리할 수 없음
  - 동적 메모리를 다 사용한 경우, stack overflow 등
- 예외 (Exception)
  - 프로그램에서 제어할 수 있는 오류
  - 읽으려는 파일이 없는 경우, 네트워크나 소켓 연결 오류 등
  - 자바 프로그램에서는 예외에 대한 처리를 수행함

예외 클래스

- 모든 예외 클래스의 최상위 클래스는 Exception 클래스

try-catch-finally 문으로 예외 처리하기

try { 예외가 발생할 수 있는 코드 부분} catch(처리할 예외 타입 e) { try 블록 안에서 예외가 발생했을 때 수행되는 부분} finally { 예외 발생 여부와 상관 없이 항상 수행되는 부분 리소스를 정리하는 코드를 주로 씀 }

try-with-resources 문

- 리소스를 자동으로 해제하도록 제공해주는 구문
- 해당 리소스가 AutoCloseable을 구현한 경우, close()를 명시적으로 호출하지 않아도 try{} 블록에서 오픈된 리소스는 정상적인 경우나 예외가 발생한 경우 모두 자동으로 close()가 호출됨
- 자바 7부터 제공됨
- FileInputStream의 경우, AutoCloseable을 구현하고 있음

AutoCloseable 인터페이스 사용하기

- AutoCloseable 인터페이스를 구현한 클래스를 만들고 close()가 잘 호출되는지 확인해본다.

예외 처리 미루기

- throws를 사용하여 예외처리 미루기
- try{} 블록으로 예외를 처리하지 않고, 메서드 선언부에 throws를 추가
- 예외가 발생한 메서드에서 예외 처리를 하지 않고 이 메서드를 호출한 곳에서 예외 처리를 하겠다는 의미
- main()에서 throws를 사용하면 가상머신에서 처리됨

예외처리

다중 예외 처리하기

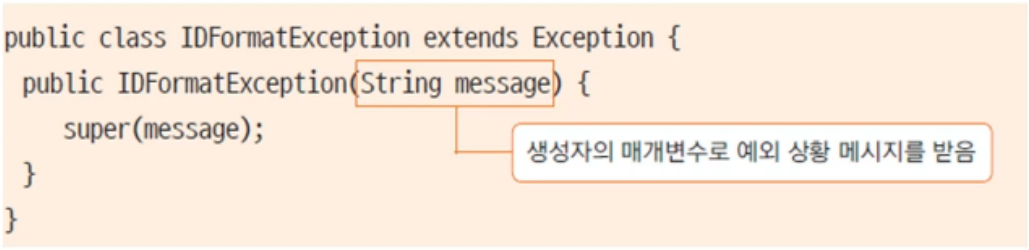
- 하나의 try {} 블록에서 여러 예외가 발생하는 경우 catch{} 블록 한 곳에서 처리하여 여러 catch{} 블록으로 나누어 처리할 수 있음
- 가장 최상위 클래스인 Exception 클래스는 가장 마지막 블록에 위치 해야함

```
public static void main(String[ ] args) {
 ThrowsException test = new ThrowsException( );
 try {
    test.loadClass("a.txt", "java.lang.String");
 } catch (FileNotFoundException e) {
    e.printStackTrace( );
 } catch (ClassNotFoundException e) {
    e.printStackTrace( );
 } catch(Exception e) {
    e.printStackTrace( );
 }
}
```

Exception 클래스로 그 외 예외 상황 처리

사용자 정의 예외

- JDK에서 제공되는 예외 클래스 외에 사용자가 필요에 의해 예외 클래스를 정의하여 사용
- 기존 JDK 클래스에서 상속받아 예외 클래스 만듬



- throw 키워드로 예외를 발생시킴

자바 입출력 스트림

- 자바의 입출력을 추상화해놓은 스트림

입출력 스트림이란?

- 네트워크에서 자료의 흐름이 물과 같다는 의미에서 유래
- 다양한 입출력 장치에 독립적으로 일관성 있는 입출력 방식 제공
- 입출력이 구현되는 곳에서는 모두 I/O 스트림을 사용
  - 키보드, 파일 디스크, 메모리 등

입출력 스트림 구분?

- I/O 대상 기준: 입력 스트림 + 출력 스트림
- 자료의 종류: 바이트 스트림, 문자 스트림
- 스트림의 기능: 기반 스트림, 보조 스트림

입출력 스트림과 출력 스트림

- 입력 스트림: 대상으로부터 자료를 읽어들이는 스트림
- 출력 스트림: 대상으로 자료를 출력하는 스트림

- 스트림의 예

종류	예시
입력 스트림	FileInputStream, FileReader, BufferedInputStream, BufferedReader 등
출력 스트림	FileOutputStream, FileWriter, BufferedOutputStream, BufferedWriter 등

바이트 단위 스트림과 문자 단위 스트림

- 바이트 단위 스트림: 바이트 단위로 자료를 읽고 씀 (동영상, 음악 파일 등)
- 문자 단위 스트림: 문자는 2바이트씩 처리 해야 함

- 스트림의 예

종류	예시
바이트 스트림	FileInputStream, FileOutputStream, BufferedInputStream, BufferedOutputStream 등
문자 스트림	FileReader, FileWriter, BufferedReader, BufferedWriter 등

기반 스트림과 보조스트림

- 기반 스트림: 대상에 직접 자료를 읽고 쓰는 기능의 스트림 (read, write)
- 보조 스트림: 직접 읽고 쓰는 기능은 없고 추가적인 기능을 제공해주는 스트림. 기반 스트림이나 또 다른 보조 스트림을 생성자의 매개변수로 표현함. decorator 패턴 기반
- 스트림의 예

종류	예시
기반 스트림	FileInputStream, FileOutputStream, FileReader, FileWriter 등
보조 스트림	InputStreamReader, OutputStreamWriter, BufferedInputStream, BufferedOutputStream 등

표준 입출력

System.in 사용하여 입력 받기

- 한 바이트씩 읽어 들임
- inputStream은 1개의 바이트만 읽으므로, 2개의 바이트를 읽고 싶다면 보조 스트림이 필요하다!
    - 예시 - `InputStreamReader isr = new InputStreamReader(System.in);`

Scanner 클래스

- `java.util` 패키지에 있는 입력 클래스
- 문자 뿐만 아니라 정수, 실수 등 다양한 자료형을 읽을 수 있음
- 생성자가 다양하여 여러 소스로부터 자료를 읽을 수 있음

생성자	설명
Scanner(File source)	파일을 매개변수로 받아 Scanner를 생성합니다.
Scanner(InputStream source)	바이트 스트림을 매개변수로 받아 Scanner를 생성합니다.
Scanner(String source)	String을 매개변수로 받아 Scanner를 생성합니다.

Console클래스

- System.in 을 사용하지 않고 콘솔에서 표준 입출력이 가능
- 이클립스와는 연동되지 않음
- Console 클래스의 메서드

메서드	설명
String readLine( )	문자열을 읽습니다.
char[ ] readPassword( )	사용자에게 문자열을 보여 주지 않고 읽습니다.
Reader reader( )	Reader 클래스를 반환합니다.
PrintWriter writer( )	PrintWriter 클래스를 반환합니다.

바이트 단위 입출력 스트림

바이트 단위 스트림

- InputStream: 바이트 단위 입력 스트림 최상위 클래스

- OutputStream: 바이트 단위 출력 스트림 최상위 클래스
- 추상 메서드를 포함한 추상 클래스로 하위 클래스가 구현하여 사용
- 주요 하위 클래스

스트림 클래스	설명
FileInputStream	파일에서 바이트 단위로 자료를 읽습니다.
ByteArrayInputStream	Byte 배열 메모리에서 바이트 단위로 자료를 읽습니다.
FilterInputStream	기반 스트림에서 자료를 읽을 때 추가 기능을 제공하는 보조 스트림의 상위 클래스입니다(보조 스트림은 '15-5 보조 스트림'에서 자세히 설명합니다).

스트림 클래스	설명
FileOutputStream	바이트 단위로 파일에 자료를 씁니다.
ByteArrayOutputStream	Byte 배열에 바이트 단위로 자료를 씁니다.
FilterOutputStream	기반 스트림에서 자료를 쓸 때 추가 기능을 제공하는 보조 스트림의 상위 클래스입니다.

  - FileInputStream - 파일의 끝에 도달하면 -1을 출력

FileInputStream과 FileOutputStream 사용하기

- 파일에 한 바이트씩 자료를 읽고 쓰는데 사용
  - 한글 같이 multibyte인 경우 FileReader, FileWriter를 사용해야 함
- 입력 스트림은 파일이 없는 경우 예외 발생 (FileNotFoundException)
- 출력 스트림은 파일이 없는경우 파일 생성하여 출력

문자 단위 입출력 스트림

문자 단위 스트림

- Reader: 문자 단위로 읽는 최상위 스트림

- Writer: 문자 단위로 쓰는 최상위 스트림
- 추상 메서드를 포함한 추상 클래스로 하위 클래스가 상속받아 구현
- 하위 클래스

스트림 클래스	설명
FileReader	파일에서 문자 단위로 읽는 스트림 클래스입니다.
InputStreamReader	바이트 단위로 읽은 자료를 문자로 변환해 주는 보조 스트림 클래스입니다.
BufferedReader	문자로 읽을 때 배열을 제공하여 한꺼번에 읽을 수 있는 기능을 제공해 주는 보조 스트림입니다.

스트림 클래스	설명
FileWriter	파일에 문자 단위로 출력하는 스트림 클래스입니다.
OutputStreamWriter	파일에 바이트 단위로 출력한 자료를 문자로 변환해 주는 보조 스트림입니다.
BufferedWriter	문자로 쓸 때 배열을 제공하여 한꺼번에 쓸 수 있는 기능을 제공해 주는 보조 스트림입니다.

FileReader와 FileWriter

- 파일에 문자를 읽고 쓸 때 가장 많이 사용하는 클래스
- 문자의 인코딩 방식을 지정할 수 있음

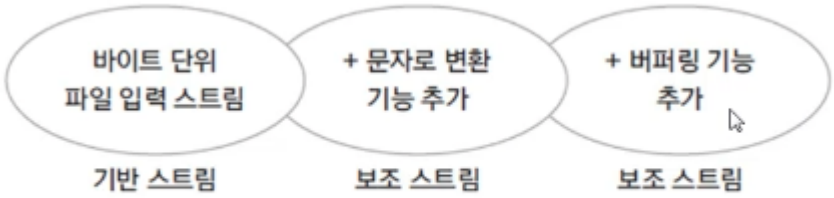
보조 스트림

보조 스트림

- 실제 읽고 쓰는 스트림이 아닌 보조적인 기능을 추가하는 스트림
- FilterInputStream과 FilterOutputStream이 보조 스트림의 상위 클래스
- 생성자의 매개 변수로 또 다른 스트림을 가짐

생성자	설명
protected FilterInputStream(InputStream in)	생성자의 매개변수로 InputStream을 받습니다.
public FilterOutputStream(OutputStream out)	생성자의 매개변수로 OutputStream을 받습니다.

- 데코레이터 패턴



여러가지 보조 스트림 사용하기

- Buffered 스트림
  - 내부에 8192 바이트 배열을 갖고 있음
  - 글을 읽거나 쓸 때 속도가 빠름
- DataInputStream/DataOutputStream
  - 자료가 **저장된 상태 그대로 자료형을 유지**하며 읽거나 쓰는 기능을 제공하는 스트림

직렬화

직렬화 (Serialization)

- 인스턴스의 상태를 그대로 저장하거나 네트웍으로 전송하고 이를 다시 복원(deserialization)하는 방식
- ObjectInputStream과 ObjectOutputStream
- 보조 스트림

Serializable 인터페이스

- 직렬화는 인스턴스의 내용이 외부 (파일, 네트워크)로 유출되는 것이므로 프로그래머가 객체의 직렬화 가능 여부를 명시함
- 구현 코드가 없는 mark interface

입출력 클래스와 데코레이터 패턴

- File 클래스
  - 파일 개념을 추상화한 클래스
  - 입출력 기능은 없고 파일의 속성, 경로, 이름 등을 알 수 있음
- RandomAccessFile 클래스
  - 입출력 클래스 중 유일하게 파일 입출력을 동시에 할 수 있는 클래스
  - 파일 포인터가 있어서 읽고 쓰는 위치의 이동이 가능함
  - 다양한 자료형에 대한 메서드가 제공됨

데코레이터 패턴 (Decorator Pattern)

- 자바의 입출력 스트림은 데코레이터 패턴을 사용
- 실제 입출력 기능을 가진 객체(컴포넌트)와 그 외 다양한 기능을 제공하는 데코레이터(보조스트림)을 사용하여 다양한 입출력 기능을 구현
- 상속보다 유연한 확장성을 가짐
- 지속적인 서비스의 증가와 제거가 용이함