

자바프로그래밍

09 예외처리

9.1 예외와 예외 클래스

9.2 예외 처리 코드

9.3 예외 종류에 따른 처리

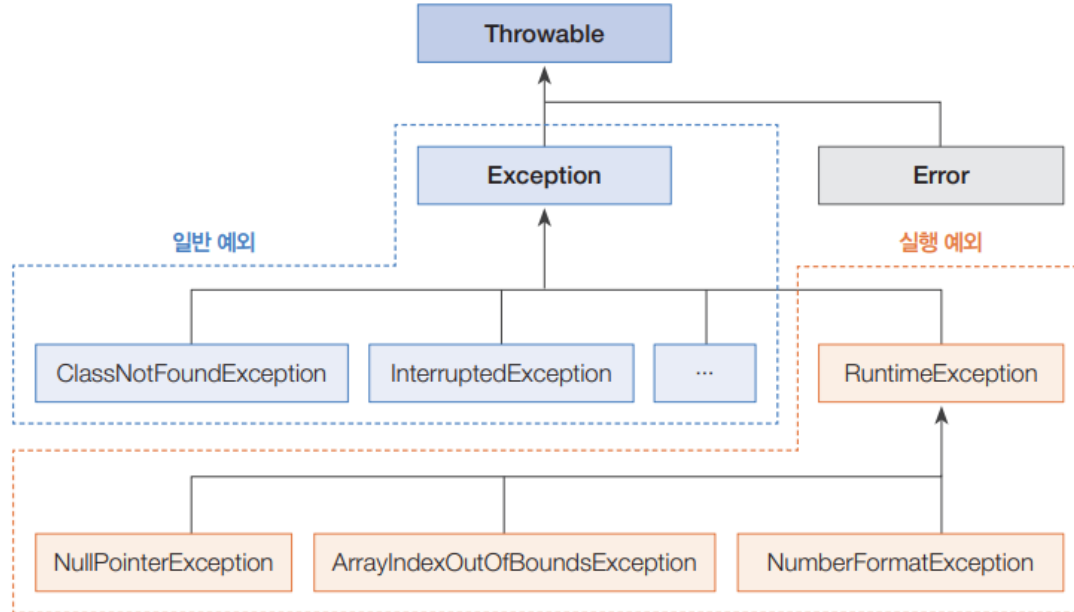
9.4 리소스 자동 닫기

9.5 예외 떠넘기기

9.6 사용자 정의 예외

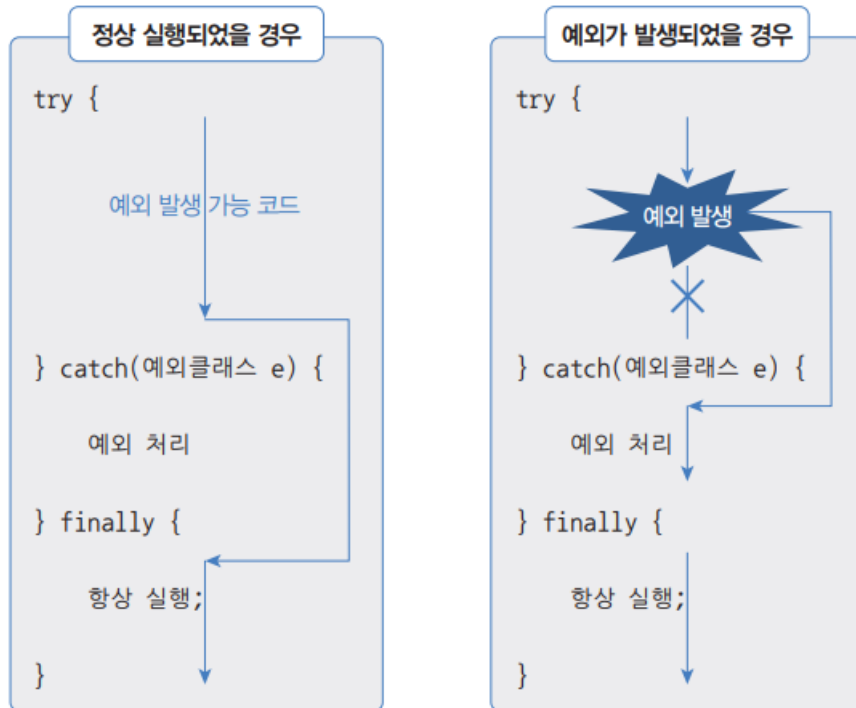
예외와 에러

- 예외: 잘못된 사용 또는 코딩으로 인한 오류
- 에러와 달리 예외 처리를 통해 계속 실행 상태를 유지할 수 있음
- 일반 예외(Exception): 컴파일러가 예외 처리 코드 여부를 검사하는 예외
- 실행 예외(Runtime Exception): 컴파일러가 예외 처리 코드 여부를 검사하지 않는 예외



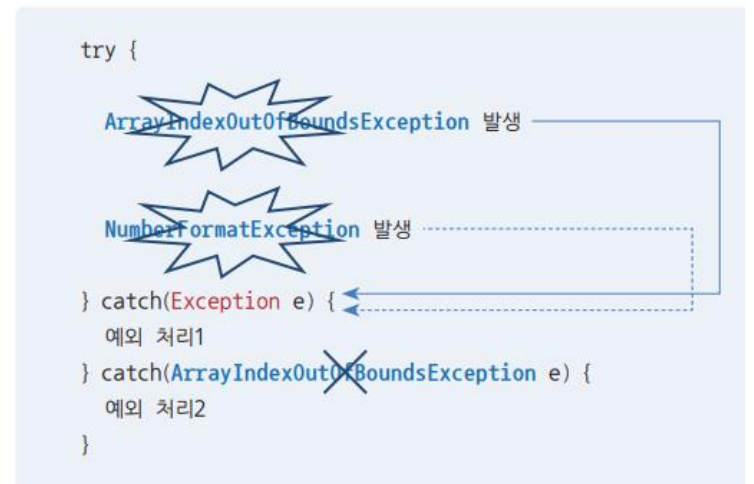
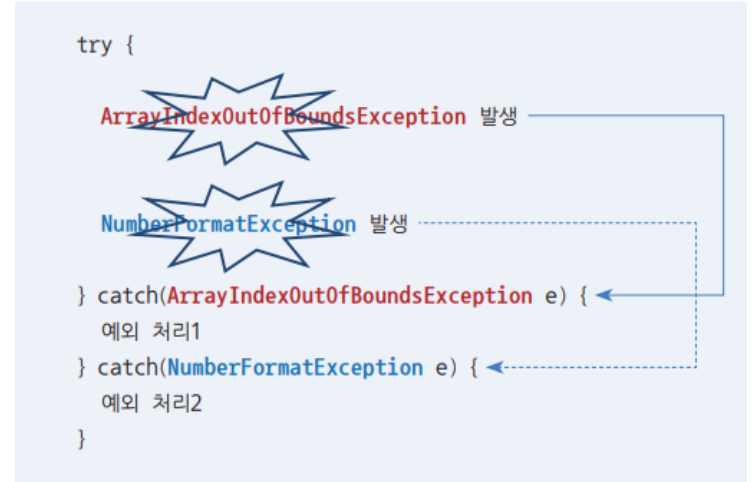
예외 처리

- 예외 발생 시 프로그램의 갑작스러운 종료를 막고 정상 실행을 유지할 수 있게 처리하는 코드
- 예외 처리 코드는 try-catch-finally 블록으로 구성
- trycatch-finally 블록은 생성자 내부와 메소드 내부에서 작성



다중 catch로 예외 처리하기

- catch 블록의 예외 클래스는 try 블록에서 발생한 예외의 종류를 말함. 해당 타입의 예외가 발생하면 catch 블록이 선택되어 실행
- catch 블록이 여러 개라도 catch 블록은 단 하나만 실행됨
- 처리해야 할 예외 클래스들이 상속 관계에 있을 때는 하위 클래스 catch 블록을 먼저 작성하고 상위 클래스 catch 블록을 나중에 작성해야 함



리소스

- 데이터를 제공하는 객체
- 리소스는 사용하기 위해 열어야(open) 하며, 사용이 끝난 다음에는 닫아야(close) 함
- 리소스를 사용하다가 예외가 발생할 경우에도 안전하게 닫는 것이 중요
- try-with-resources 블록을 사용하면 예외 발생 여부와 상관없이 리소스를 자동으로 닫아줌

```
FileInputStream fis = null;
try {
    fis = new FileInputStream("file.txt");
    ...
} catch(IOException e) {
    ...
} finally {
    fis.close();
}
```

파일 열기

파일 닫기

```
try(FileInputStream fis = new FileInputStream("file.txt")){
    ...
} catch(IOException e) {
    ...
}
```

예외 처리하기

- 메소드 내부에서 예외 발생 시 throws 키워드 이용해 메소드를 호출한 곳으로 예외 처리하기
- throws는 메소드 선언부 끝에 작성. 처리할 예외 클래스를 쉼표로 구분해서 나열

```
리턴타입 메소드명(매개변수,...) throws 예외클래스1, 예외클래스2, ... {  
}
```

```
public void method1() {  
    try {  
        method2(); //method2() 호출  
    } catch(ClassNotFoundException e) {  
        System.out.println("예외 처리: " + e.getMessage());  
    }  
}  
  
public void method2() throws ClassNotFoundException {  
    Class.forName("java.lang.String2");  
}
```



- 나열할 예외 클래스가 많으면 throws Exception 또는 throws Throwable 만으로 모든 예외 처리하기

```
리턴타입 메소드명(매개변수,...) throws Exception {  
}
```

사용자 정의 예외

- 표준 라이브러리에는 없어 직접 정의하는 예외 클래스
- 일반 예외는 Exception의 자식 클래스로 선언.
실행 예외는 RuntimeException의 자식 클래스로 선언

```
public class XXXException extends [ Exception | RuntimeException ] {  
    public XXXException() {  
    }  
  
    public XXXException(String message) {  
        super(message);  
    }  
}
```

기본 생성자

예외 메시지를 입력받는 생성자

예외 발생시키기

- throw 키워드와 함께 예외 객체를 제공해 사용자 정의 예외를 직접 코드에서 발생시킬 수 있음
- 예외의 원인에 해당하는 메시지를 제공하려면 생성자 매개값으로 전달

```
throw new Exception()  
throw new RuntimeException();  
throw new InsufficientException();
```

```
throw new Exception("예외메시지")  
throw new RuntimeException("예외메시지");  
throw new InsufficientException("예외메시지");
```