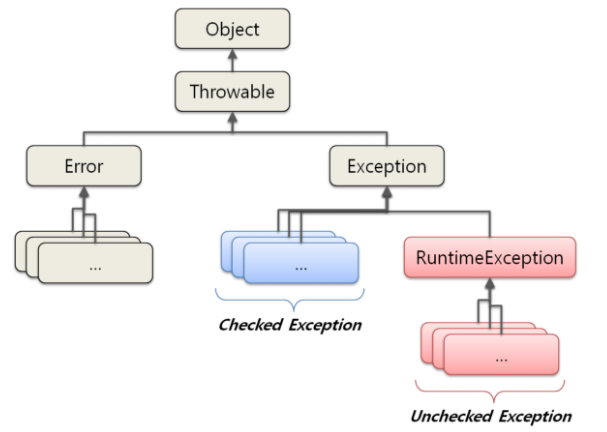
Error (오류)

* 시스템에 비정상적인 상황이 생겼을 때 발생.
* 시스템 레벨에서 발생하여 애플리케이션에서 고려할 필요 없음
* Thread death, out of memory

Exception (예외)

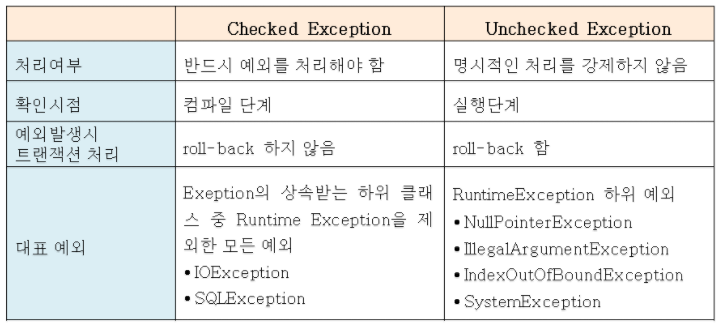
* 개발자가 구현한 로직에서 발생
* 개발자가 처리할 수 있기 때문에 처리요망



둘 다 Object의 자식 클래스인 Throwable을 상속받음

Exception의 자식 클래스 중 Runtime Exception과 그 하위 클래스를 Unchecked Exception

나머지는 Checked Exception이라 불림.



Checked Exception

* 개발할 때 try catch를 강제함 그래서 처리를 하지 않으면 로직 에러발생
* 컴파일 단계에서 명확하게 Exception check가 가능함
* File, sql, io

Unchecked Exception

* 피할 수 있지만 부주의로 발생하는 경우가 대부분이라 강제하지는 않음
* 컴파일 단계가 아니라 Runtime 단계에서 확인가능
* Null point, index out of bound, arithmetic (divide by zero)
* 트랜잭션 (여러 작업을 묶어놓은 것)을 roll-back함

예외 처리 방법

1. 예외 복구

* 예외가 발생하여도 애플리케이션이 정상 작동하게
* 네트워크 상태가 좋지 않아서 끊긴 경우 catch에서 일정시간 대기 후 다시 연결하는..

1. 예외처리 회피

* Throws를 통해 호출한 쪽으로 예외를 던지고 거기서 예외를 처리하게끔 하는 것
* 이건 예외를 던진 곳에서 반드시 예외를 처리한다는 확신이 있어야 됨

1. 예외 전환

* catch에서 throw를 통해 좀 더 자세한 exception을 던져주는 것

throw와 throws의 차이

* throw는 프로그래머의 판단에 따른 처리 (메소드 안에서 상위 블록으로 예외를 던지는)
* 그니까 try catch가 있고 catch안에서 exception 발생시킴
* throws는 전가
* 예외를 발생한 곳에서 처리하지 않고, 자신을 호출한 메소드에게 책임을 전가하는 것