예외처리

목표:

- 예외가 프로그램에서 발생하는 상황에 대해서 처리할 프로세스 구조를 안다.
- 자바에서 예외를 처리하는 프로그램의 기본 구조를 안다.
- 여러가지 예외를 처리하는 **api**에 대해서 기능에 대해서 알고 실행할 줄 안다.
- 사용자 정의 예외 클래스를 선언하고 활용할 줄 안다.

생각해봅시다:

- 프로그래밍을 수행할 때, 발생할 수 있는 예외는 어떤 것이 있을까?
  - 컴파일 에러
  - 실행 에러
  - 외부 환경에 의해

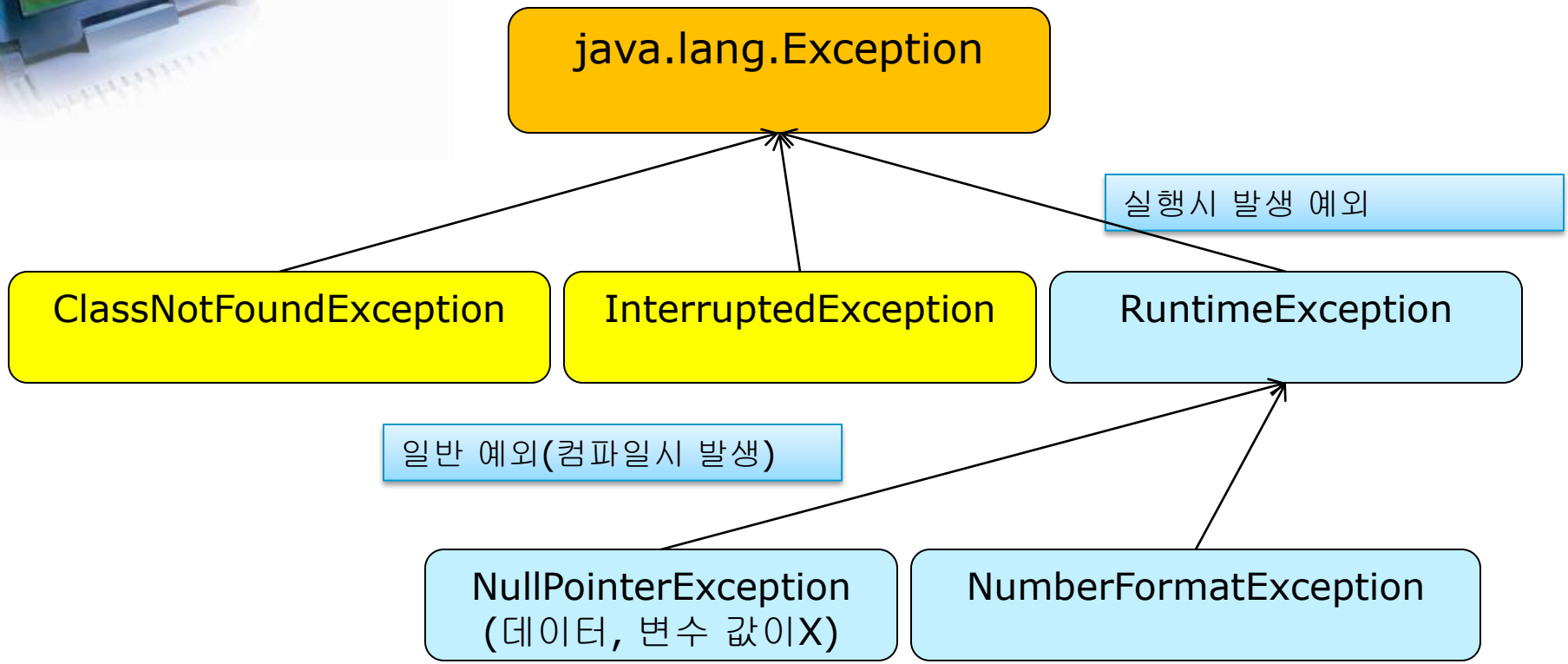
예외란?

- 컴퓨터 하드웨어적으로 오동작, 고장으로 응용프로그램 실행이 발생하는 것을 자바에서는 error라고 한다.
- JVM 실행 문제가 생기는 것으로 프로그램이 견고하게 만들었어도 실행이 불능하게 된다.
- 이런 프로그램에서 개발자는 대처할 방법이 없다.
- 자바에서 error 이외에 예외(Exception)라고 부르는 오류가 있다. 예외란 사용자의 잘못된 조작 또는 개발자 코딩 문제 발생하는 **프로그램 오류**를 말한다. 원래 이런 오류가 발생하면 프로그램 종료된다. 자바에서는 예외처리(Exception) 프로그램을 종료하지 않고 실행 상태 유지되도록 한다.
- 예외(Exception)

  일반 예외: 컴파일 체크 예외

  실행 예외: 컴파일 후, 실행 처리

예외 클래스 구조


java.lang.Exception
실행시 발생 예외
ClassNotFoundException
InterruptedException
RuntimeException
일반 예외(컴파일시 발생)
NullPointerException
(데이터, 변수 값이X)
NumberFormatException

실행 예외들.

- 컴파일 시 발생하는 것이 아니기에 개발자 경험이나, 간단한 테스트 코드로 예외처리를 하여야 하는 것을 말한다.
- NullPointerException
  - 객체 참조가 없는 상태, null값은 갖는 참조 변수로 접근 연산자(.) 사용해서 멤버를 호출했을 때 발생하는 예외
- ArrayIndexOutOfBoundsException
  - 배열에서 인덱스의 범위를 초과해서 사용했을 때, 발생하는 예외를 말한다.
- NumberFormatException
  - 문자열 → 숫자를 변경하는 경우, 문자열을 숫자로 변환하는데 숫자형 문자이어야 하는데, 그렇지 않을 때, 처리하는 객체 메소드에서 예외발생

예외 처리 코드:

- try{
  - 1.예외가 발생할만한 코드..
  - 2. 예외발생시, catch쪽에 정의된 예외클래스로 던짐
    - String name=null; System...(name.length());
    - NullpointerException 객체를 던짐.
  - 3. 상위 라인에 예외가 발생하면, 실행되지 않음
  - 4. 상위 라인에 예외가 발생하면, 실행되지 않음
- }catch(예외클래스 참조){
  - NullpointerException 이상의 상위클래스가 정의되어 있
- }

예외 확인 예제

- 1단계
  - id를 입력하세요
  - id=null; 입력이 안된 경우, id="himan";
  - 입력된 id: @@@
  - catch
    - 아이디가 입력이 되지 않았습니다.
- 2단계
  - id, pass
  - id와 password가 유효할 때, @@님 환영합니다.
  - catch
    - id또는 password가 입력되지 않았습니다.

다중 catch

- 예외는 동시 다발적으로 여러 예외를 발생할 수 있지만, 여러 예외를 하나의 try{}catch(예외1){}catch(예외2){} 구문으로 처리할 수 있다. 그 외 예외 내용은 상위 클래스인 Exception을 통해 처리할 수 있다.
- 형식
  - try{
    - 여러 예외를 발생할 만한 코드..
    - 예외1
    - 예외2
  - }catch(예외1){
  - }catch(예외2){
  - }catch(**Exception** e){}

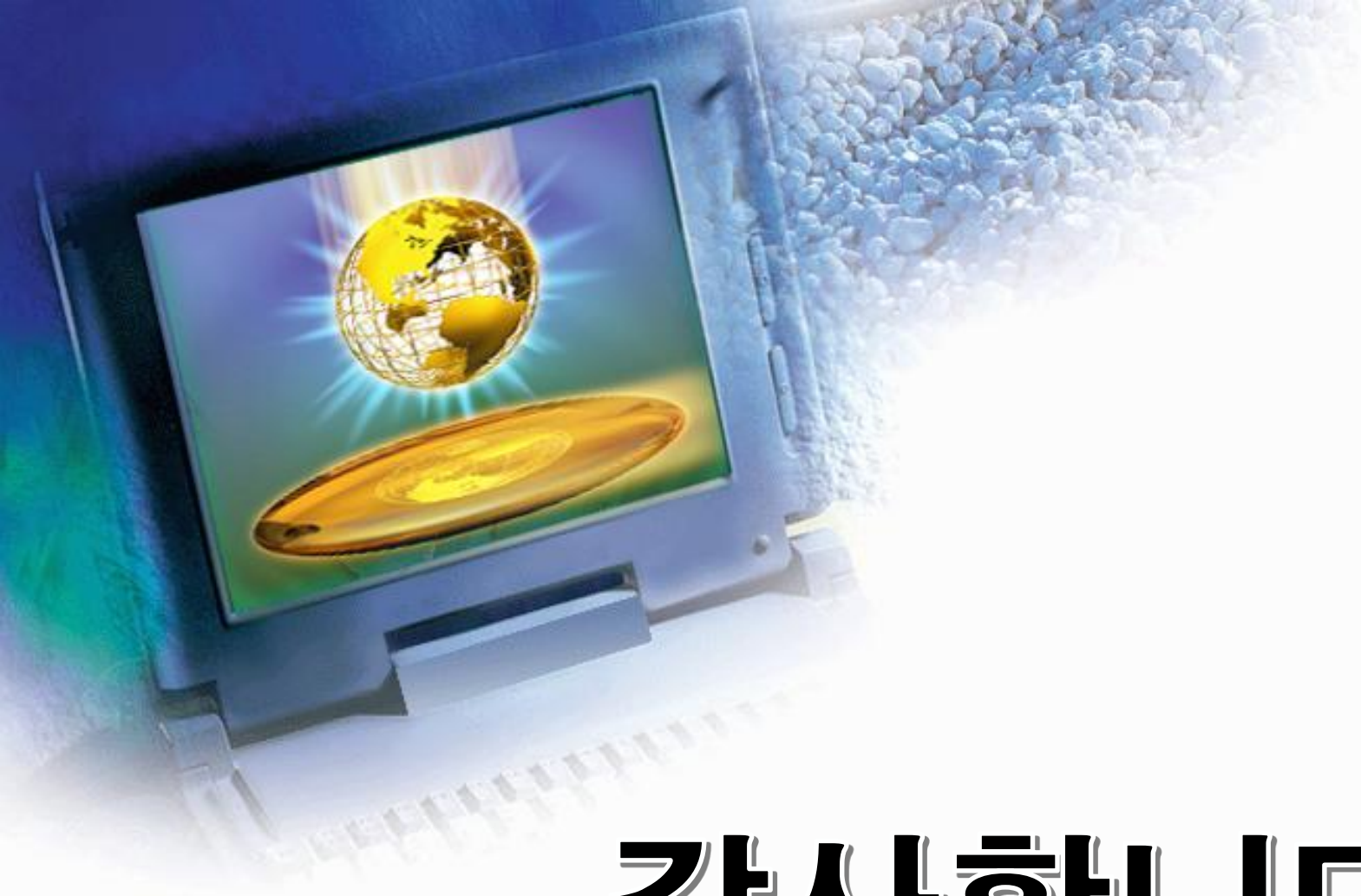
finally

- 예외를 발생하는 경우나 발생하지 않고 정상적으로 처리하는 경우 모두 다, 처리할 프로세스나 프로그램이 있을 때는 try catch문 마지막 블럭으로 finally{} 구문을 포함 시킨다.
- ex)
  - 파일입출력 클래스 선언
  - try{
  - 참조 = 파일입출력 객체생성..
  - ... 예외/ 발생하지..
  - }catch(예외){
  - }**finally**{
  - 파일입출력 객체 자원해제 .close();
  - }

예외 클래스에서 활용되는 메서드

- catch(예외 클래스 참조){
  - 참조.메서드();
- }
- e.getMessage() : 예외 가지고 있는 메시지를 문자열로 return
- e.printStackTrace() : 예외의 발생 경로를 추적내용 출력

정리 및 확인하기



감사합니다 !