**Day11 o(\*￣▽￣\*)ブ**

🚨 오류 Error

컴파일 오류: 문법(Syntax)이 올바르지 않은 상태

실행오류(Runtime Error)

에러: 시스템이 정상적 기능을 수행할 수 없는 상태 – 수습 불가

예외: 실행 중 예기치 않은 문제 발생한 상태 – 수습 가능

논리오류: 프로그래머가 원하는 결과가 산출되지 않은 상태

Exception: 예외 잘못된 사용, 코딩으로 인한 오류이다.

일반 예외(Exception): 컴파일러가 예외 처리 코드 여부를 검사하는 예외이다.

실행 예외(Runtime Exception): 컴파일러가 예외 처리 코드 여부를 검사하지 않는 예외이다.

🛠 예외 처리

갑작스러운 종료를 방지해 프로그램을 정상 수행할 수 있도록 하는 것이다.

catch 문을 여러개 작성하면 해당 오류를 가장 먼저 만나는 catch문이 실행된다. 아래와 같이 작성하면 FileNotFoundException에 해당하는 예외처리가 실행된다. Exception을 작성할 때는 모든 예외를 다 처리하기 때문에 가장 아래쪽에 작성하는 것이 좋다.

왜냐하면 모든 예외 처리 클래스는 Exception 클래스를 상속받았기 때문에 Exception 예외는 모든 예외를 처리할 수 있다.

쓰레드

입출력 명령이 늘어나 CPU의 Idle time(유휴 시간)이 늘어나게 되었다. 따라서 CPU를 효율적으로 사용하기 위해서 (유휴시간을 줄이기 위해) 멀티 쓰레드 개념으로 컨텍스트 스위칭을 수행한다.

메모리의 크기가 쓰레드의 개수에 영향을 주며 쓰레드의 개수는 동시접속자에 영향을 준다.

CPU의 유휴시간을 최소화

응답시간을 최소화

동시 실행 효과를 위해 쓰레드 사용

Meta Data

= Data of Date

데이터에 대한 설명 데이터이다.

이미지 데이터의 경우 실질 데이터는 색상정보를 이용해 시각정보를 표현하는 데이터이다. 메타 데이터는 해당 이미지데이터의 속성정보를 말한다. 이름, 생성일, 이미지 크기, 만든 날짜 등을 말한다. 그리고 HTML의 경우 <h1> !내용! </h1> 의 명령줄에서 데이터는 “ !내용! ”이고 해당 데이터의 메타데이터는 <h1>을 말한다. 그 외 주석문 또한 메타 데이터이다.

즉 추가적 정보를 말한다. 하지만 주석문은 제어도 할 수 없고 프로그램에게 적용할 수 없이 컴파일 단계에서 사라지기 때문에 오직 개발자에게만 제공되는 정보이다. 그래서 현재는 명령어에 해당 추가적 정보가 영향을 줄 수 있도록 확장되었다.

어노테이션은 주석문처럼 사용자에게만 제공되는 정보가 아닌 시스템에게도 정보를 제공하기위해 사용되는 기술이다. 프로그램에게 메타데이터를 제공하는 것이다. 컴파일이나 실행할 때 어떻게 처리할

어노테이션은 적용 시점과 제거 시점을 지정할 수 있다. 설정 정보, 설명문

[프로젝트 관련]

새로운 주제 선정 시

추가 라이브러리, 프레임워크 사용 금지

순수 자바 기본 라이브러리만 사용 가능

코드 분석 (채팅, 오목 : 네트워크기반, 그래픽)하고 수정 작성

제출 형식 :

폭포수 모델

Step1. WHAT: final의 경우 비즈니스 적인 목표, 공부만의 목적이면 안됨 – 요구사항정의서(엑셀 권장) 기능적(회원가입기능) 요구사항, 비기능적(보안 기능, 1초안에 반응하겠다. 등 가시적인 기능이 아닌 시스템에서 제공해야 할 것들) 요구사항 모두 작성할 것 -> UML의 UseCase Diagram을 이용해 작성…

* 브레인 스토밍: 회의 시 포스트 잇으로 사용해 결과물 사진 찍어 포트폴리오 제출.

Step2. HOW: 아키텍처(그림으로 간단하게), 데이터베이스 사용 시 ERD, UX/UI

* 간트 차트 작성할 것
* 다양한 툴 사용해보는 경험이 도움이 될 것: UX/UI는 구글 많이 사용함

Step3. 설계: Class Diagram (역할, 클래스 간의 관계: 정적인 관계만 확인 가능), Sequence Diagram(객체간 상호작용)

Step4. 구현: 실행화면, 화면 캡쳐

Step5. 테스트: 단위 별, 통합테스트, 코드 커버리지 등 : 이번엔 체크리스트로 단위 별 기능 테스트만 진행할 것.