OOP vs PP vs FP

날짜 : 2024-08-09 13:47

주제:

메모:

1. 객체지향 프로그래밍 (Object-Oriented Programming, OOP)

- **기본 개념**: 객체지향 프로그래밍은 데이터를 중심으로 프로그램을 구조화한다. 프로그램을 객체(object)라는 단위로 나누고, 각 객체는 데이터(속성)와 데이터를 처리하는 메서드(함수)를 포함한다.
- **중심 요소**: 클래스와 객체. 클래스는 객체를 생성하기 위한 청사진이고, 객체는 클래스의 인스턴스이다.
- **주요 특징**:
  - **캡슐화**: 객체 내부의 데이터를 외부에서 직접 접근하지 못하도록 보호하고, 공개된 메서드를 통해서만 접근하도록 한다.
  - **상속**: 기존 클래스의 특성을 물려받아 새로운 클래스를 생성할 수 있다.
  - **다형성**: 동일한 메서드가 객체의 타입에 따라 다르게 동작할 수 있다.
  - **추상화**: 객체의 복잡성을 숨기고, 필요한 부분만을 보여준다.
- **장점**:
  - 코드의 재사용성이 높고, 유지보수가 쉽다.
  - 현실 세계의 객체와 유사한 구조로 모델링이 가능하다.
- **단점**:
  - 초기 설계가 복잡할 수 있다.
  - 작은 프로젝트에서는 과도한 설계가 될 수 있다.

2. 절차지향 프로그래밍 (Procedural Programming)

- **기본 개념**: 절차지향 프로그래밍은 프로그램을 순차적으로 실행되는 절차(Procedure)로 구성한다. 각 절차는 특정 작업을 수행하는 함수 또는 서브루틴으로 구성된다.
- **중심 요소**: 함수와 절차. 프로그램은 여러 개의 함수로 구성되며, 함수 간의 호출 순서에 따라 프로그램이 실행된다.
- **주요 특징**:
  - **순차적 흐름**: 코드가 위에서 아래로 순차적으로 실행된다.
  - **모듈화**: 기능별로 함수를 나누어 프로그램을 구성한다.
  - **전역 변수 사용**: 함수 간의 데이터 공유를 위해 전역 변수를 사용할 수 있다.
- **장점**:
  - 이해하고 구현하기 쉽다.
  - 함수로 모듈화하여 코드 재사용이 가능하다.
- **단점**:
  - 코드가 복잡해질수록 유지보수가 어렵다.

    - 데이터와 함수가 분리되어 있어 코드의 의도를 파악하기 어려울 수 있다.
    - 프로그램 규모가 커질수록 비효율적일 수 있다.

3. 함수형 프로그래밍 (Functional Programming, FP)

- **기본 개념**: 함수형 프로그래밍은 함수를 일급 시민(First-class citizen)으로 취급하며, 프로그램을 함수의 조합으로 구성한다. 여기서 함수는 수학적 의미의 함수처럼 입력을 받아 출력을 내는 역할만을 수행한다.
- **중심 요소**: 순수 함수(Pure Function)와 불변성(Immutable). 순수 함수는 부작용이 없고, 동일한 입력에 대해 항상 동일한 출력을 반환합니다.
- **주요 특징**:
    - **불변성**: 데이터는 한 번 생성되면 변경되지 않으며, 변화를 적용할 때는 새로운 데이터를 생성한다.
    - **고차 함수**: 함수를 인수로 받거나 함수를 반환하는 함수가 자주 사용된다.
    - **함수 합성**: 작은 함수들을 결합하여 더 복잡한 함수를 생성한다.
    - **부작용 없는 코드**: 함수는 외부 상태를 변경하지 않으며, 프로그램의 상태 관리가 명확해진다.
- **장점**:
    - 병렬 처리에 유리하며, 프로그램의 동작을 예측하기 쉽고 테스트가 용이하다.
    - 코드의 가독성과 유지보수성이 좋다.
- **단점**:
    - 절차지향이나 객체지향 패러다임에 익숙한 개발자들에게는 접근이 어려울 수 있다.
    - 순수 함수 사용의 제약이 실용적인 문제를 야기할 수 있다.

결론

- **객체지향**은 데이터와 그 데이터를 처리하는 메서드를 함께 묶어 현실 세계의 객체를 모델링하는 데 중점을 둔다.
- **절차지향**은 프로그램의 흐름을 제어하는 절차(함수) 중심으로 구조화되며, 간단하고 직관적인 코드 작성을 목표로 한다.
- **함수형**은 순수 함수와 불변성을 강조하며, 수학적 함수 개념을 기반으로 하여 프로그램을 구성한다.

출처(참고문헌)

-

연결문서

-