노출모듈 패턴

작성 날짜	@2023년 11월 23일
구분	

노출모듈 패턴

- 노출모듈패턴이란?
  즉시 실행 함수를 통해 private, public 같은 접근 제어자를 만드는 패턴
- 사용목적
  요소 분리, 유지/보수/재사용 증가, 내부 동작은 은닉하고 필요한 인터페이스만 외부에 노출시킴으로 모듈 간 의존성을 낮추고, 모듈 간 인터페이스를 명확히하여 상호 작용할 수 있도록 한다.
  상세설명
  소프트웨어 디자인 패턴 중 하나로, 객체 지향 프로그래밍에서 사용되는 디자인 패턴 중 하나입니다. 이 패턴은 복잡한 소프트웨어 시스템을 구성하는 요소들을 분리하여 유지보수성을 높이고 재사용성을 증가시키기 위해 사용한다.
  노출모듈 패턴의 핵심 아이디어는 내부 동작이나 구현 세부 사항을 외부로부터 숨기고, 오직 필요한 인터페이스만을 외부에 노출시키는 것입니다. 이를 통해 모듈 간 의존성을 낮추고, 모듈 간 인터페이스를 명확히하여 상호 작용할 수 있도록 합니다.
- 사용방법
  자바스크립트는 private나 public 같은 접근 제어자가 존재하지 않고 전역 범위에서 스크립트가 실행되기 때문에 노출모듈 패턴을 통해 private나 public 접근 제어자를 구현하기도 한다. → 전역 범위에서 실행되는 프로그램은 내부 어플리케이션과 종속된 라이브러리 코드의 데이터들로 인해 충돌이 발생 할 수 있음.

▼ 🤔 즉시 실행 함수(Immediately Invoked Function Expression, IIFE)?
즉시 실행 함수(Immediately Invoked Function Expression, IIFE)는 JavaScript에서 사용되는 특별한 함수로 함수를 정의함과 동시에 즉시 호출하는 방식을 가진다.

- 사용목적
  일반적으로 함수를 정의하고 그 함수를 나중에 호출하는 것과는 달리, IIFE는 함수를 정의함과 동시에 즉시 호출하여 실행합니다. 이는 함수를 선언하자마자 실행하여 생성된 변수들을 전역 스코프로부터 격리시키고, 코드의 모듈화나 변수 충돌을 방지하는 데 사용된다.
- 초기화 코드, 라이브러리 내 전역 변수의 충돌 방지 등에 사용한다.
- 예시 1

```
(function () {
    // statements 여기에 코드 작성
})()
```

- `(function() { /* 코드 작성 */ })`는 익명 함수를 감싸는 괄호로, 함수를 정의하는 부분입니다. 그리고 마지막의 `()`를 사용하여 해당 함수를 즉시 실행시킵니다.
  또한, IIFE를 사용할 때 함수 내부에 매개변수를 전달하여 외부에서 값을 전달하거나, 함수 내부에서 반환값을 얻을 수 있습니다. 이를 통해 모듈 패턴과 비슷한 효과를 낼 수 있습니다.
- 예시 2

```
javascriptCopy code
(function(name) {
    console.log('안녕하세요, ' + name + '님!');
})('Alice');
```

IIFE는 주로 프로젝트 내에서 개별 모듈이나 라이브러리로 구분되는 등의 스코프를 구분하기 위해 사용되며, 코드를 더 깔끔하게 유지하고 전역 스코프의 오염을 방지하는 데 유용하다.

▼ 🤔 접근 제어자

- **public**클래스에 정의된 함수에서 접근 가능하며 ***자식 클래스와 외부 클래스***에서 접근 가능한 범위

- **protected**클래스에 정의된 함수에서 접근 가능, ***자식 클래스에서 접근 가능하지만 외부 클래스에서 접근 불가능한*** 범위
- **private**클래스에 정의된 함수에서 접근 가능하지만 ***자식 클래스와 외부 클래스에서 접근 불가능한*** 범위

장점

- 개발자에게 깔끔한 접근 방법을 제공
- ***private 데이터 제공***
- 전역 변수를 덜 더럽힘
- 클로저를 통해 함수와 변수를 지역화
- 스크립트 문법이 더 일관성 있음
- 명시적으로 public 메소드와 변수를 제공해 명시성을 높임

단점

- ***private 메소드 접근할 방법이 없음*** (그래서 테스트 불가란 얘기를 하는데 private은 테스트 할지 안할지 고민해봐야)
- private 메소드에 대해 함수 확장하는데 어려움이 있음
- private 메소드를 참조하는 public 메소드를 수정하기 어려움

```
const pukuba = (() => {
    const a = 1
    const b = () => 2
    const public = {
        c: 2,
        d: () => 3
    }
    return public
})()

console.log(pukuba)
console.log(pukuba.a)
// { c: 2, d: [Function: d] }
// undefined
```

a와 b는 다른 모듈에서 사용할 수 있는 변수나 함수인 private 범위를 가집니다.

다른 모듈에서 접근할 수 없고 c와 d는 다른 모듈에서 사용할 수 있는 변수나 함수인 public 범위를 가집니다.

참고로 이 원리를 기반으로 만든 자바스크립트 모듈 방식으로는 CJS(CommonJS) 모듈 방식이 있습니다.

https://velog.io/@juijeong8324/Design-Pattern-8