14장. 전역 변수의 문제점

14.1 변수의 생명 주기

변수는 자신이 선언된 위치에서 생성되고 소멸한다.

== 변수의 생명 주기는 메모리 공간이 확보된 시점부터 메모리 공간이 해체되어 가용 메모리 풀에

반환되는 시점까지(가비지 콜렉터에 의해 해제)

지역 변수의 생명 주기는 함수의 생명 주기와 일치

전역 변수는 마지막 문이 실행되어 더 이상 실행할 문이 없을 때 종료(=전역 객체의 생명 주기와 일치)

var 키워드로 선언한 전역 변수의 생명 주기는 전역 객체의 생명 주기와 일치

14.2 전역 변수의 문제점

- **암묵적 결합이란?**

  모든 코드가 전역 변수를 참조하고 변경할 수 있는 것

1. 전역 변수는 생명 주기가 길어 메모리 리소스도 오랜 기간 소비함
2. 전역 변수의 검색 속도가 가장 느림
3. 파일이 분리되어 있어도 하나의 전역 스코프를 공유하므로 예상치 못한 결과 초래

14.3 전역 변수의 사용을 억제하는 방법

전역 변수를 반드시 사용해야 할 이유가 없다면 지역 변수를 사용해야 함

→ 변수의 스코프는 좁을수록 좋음

1. 모든 코드를 즉시 실행 함수로 감싸면 모든 변수는 즉시 실행 함수의 지역 변수가 된다.
2. 네임스페이스 역할을 담당할 객체 생성 및 전역 변수로 사용하고싶은 변수 프로퍼티로 추가
3. 모듈 패턴을 사용
4. ES6 모듈 사용하기

- **모듈 패턴이란?**

  클래스를 모방해서 관련있는 변수와 함수를 모아 즉시 실행 함수로 감싸 하나의 모듈로 만드는 것

- **정보 은닉이란?**

  캡슐화는 객체의 특정 프로퍼티나 메서드를 감출 목적으로 사용하는 것