Chap15 let, const, 키워드와 블록 레벨 스코프

15-1 var 키워드로 선언한 변수의 문제점

▼ 1. 변수 중복 선언 허용

var키워드로 선언한 변수는 중복 선언이 가능하다.

```
var x=1;
var x=100;
console.log(x); //100
```

▼ 2.스코프

var 키워드는 함수 레벨 스코프에서만 지역 스코프로 인정된다는 점이다. 따라서 블록 레벨 스코프에서 var키워드로 변수를 선언할 시 저역 변수가 되버린다.

▼ 3.변수 호이스팅

var 키워드로 변수를 선언하면 변수 호이스팅에 의해 변수 선언문이 스코프의 선두로 끌어 올려진 것처럼 동작한다. 이는 에러를 발생시키지는 않지만 프로그램 흐름 상 맞지않으며 가독성을 떨어뜨린다.

15-2 let 키워드

▼ 1.변수 중복 선언을 금지한다.

let 키워드로 선언한 변수는 중복 선언이 불가능하다.

```
let x=1;
let x=2; //SyntaxError:Identifier 'x' has already been dec
```

▼ 2.스코프

let키워드로 선언한 변수는모든 코드 블록을 지역 스코프로 인정하는 블록 레벨 스코프를 따른다.

▼ 3.변수 호이스팅

호이스팅이 발생하지 않는 것처럼 동작한다.

let 키워드로 선언한 변수는 '선언 단계'와 '초기화 단계'가 분리되어 진행된다. 즉 런타임 이전에 자바스크립트 엔진에 의해 암묵적으로 선언 단계가 먼저 실행되지만 초기화 단계는 변수 선언문에 도달했을 때 실행된다.

따라서 초기화 단계가 실행되기 이전에 변수에 접근하려고 하면 참조 에러가 발생하는데 이는 일시적 사각지대에 의해 생긴다.

▼ 일시적 사각지대(TDZ)

스코프 시작 지점부터 초기화 시작 지점까지 변수를 참조할 수 없는 구간

▼ 4.전역 객체와 let

`var`키워드로 전역 변수와 전역 함수, 그리고 선언하지 안흔 변수(암묵적 전역변수)는 모두 `window`객체의 프로퍼티가 된다. (전역 객체의 프로퍼티를 참조할 때`window`를 생략할 수 있다.) 하지만 `let`키워드로 선언된 변수는 전역 객체의 프로퍼티가 아니다.

```
// 이 예제는 브라우저 환경에서 실행해야 한다.
let x = 1;

// let, const 키워드로 선언한 전역 변수는 전역 객체 window의 프로퍼
console.log(window.x); // undefined
console.log(x); // 1
```

15-3 const 키워드

- const 키워드로 선언한 변수는 반드시 선언과 동시에 초기화 해야한다.
- 재할당이 금지 된다.

- 상수이다.
- const 키워드로 선언된 변수에 객체를 할당한 경우 값을 변경할 수 있다.
- cosnt 키워드는 재할당을 금지할 뿐 불변을 의미하지 않는다. 다시말해 새로운 값을 재할당하는 것은 불가능하지만 프로퍼티 동적 생성, 삭제, 프로퍼티 값의 변경을 통해 객체를 변경하는 것은 가능하다.

15-4 var vs let vs const

- ES6를 사용한다면 `var`는 사용하지 않는다.
- 재할당이 필요한 경우에 한정해 `let`키워드를 사용하고, 스코프는 최대한 좁게 만든다.
- 변경이 없고 읽기 전용으로 사용하는 원시 값과 객체에는 `const`를 사용한다.(재할당이 안되므로 안전하다.)