정보를 감추는 목적으로 private 사용하지 않기

Item 56

ryan-dia(강철원)

과거 JavaScript 클래스의 속성을 비공개로 만드는 방법 (1)

```
class Foo {
    _private = 'secret123';
}

```

```
class Foo {
  _private = 'secret123';
}

const f = new Foo();
console.log(f._private); // 'secret123'

```

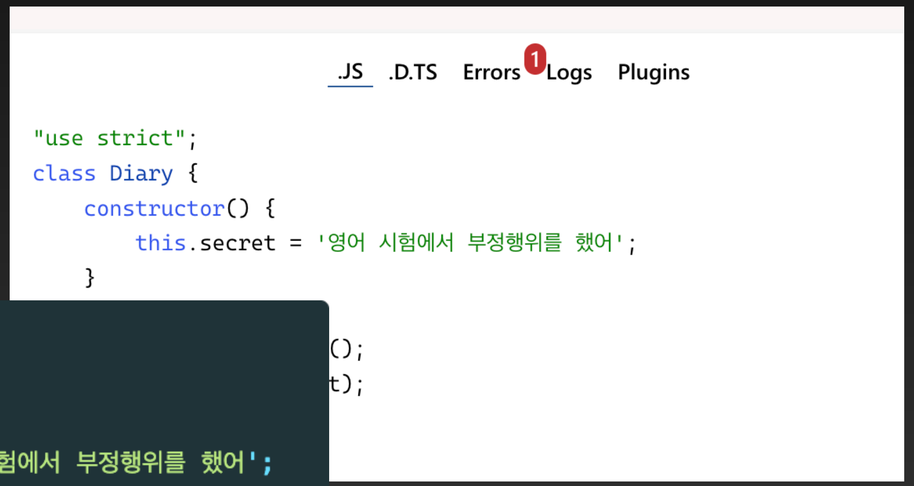
과거 JavaScript 클래스의 속성을 비공개로 만드는 방법 (2)

```
class Diary {
  private secret = '영어 시험에서 부정행위를 했어';
}
const diary = new Diary();
diary.secret;
```

vscode

```
class Diary {
  private secret = '영어 시험에서 부정행위를 했어';
}
const diary = new Diary();
diary.secret;    Property 'secret' is private and only accessible within class 'Diary'.
```

Typescript Playground




```
class Diary {
  private secret = '영어 시험에서 부정행위를 했어';
}

const diary = new Diary();
(diary as any).secret // 허용됨
```

```
class Diary {
  private secret = '영어 시험에서 부정행위를 했어';
}

const diary = new Diary();
diary['secret']; // 허용됨
```

앞서 소개한 두 가지 방법은 비공개 정보를 완벽하게 숨기지 못합니다.

더 견고하게 숨길 수 있는 방법이 없을까?

1. 클로저

```
declare function hash(text: string): number;
class PasswordChecker {
  checkPassword: (password: string) => boolean;
  constructor(passwordHash: number) {
    this.checkPassword = (password: string) => {
      return hash(password) === passwordHash;
    };
  }
}
const checker = new PasswordChecker(hash('비밀번호'));
checker.checkPassword('비밀번호');

```

단점

1. passwordHash가 생성자 외부에서 참조할 수 없기 때문에
   해당 변수를 사용하는 모든 메서드도 생성자 내에서 정의되어야 합니다.
   이로 인해 각 메서드의 복사본이 클래스 인스턴스마다 생성되며, 이는 메모리 사용량 증가로 이어집니다.

2. 같은 클래스의 다른 인스턴스가 비공개 데이터에 접근하지 못합니다.

2. 비공개 필드

```
class PasswordChecker {
  #passwordHash: number;
  constructor(passwordHash: number) {
    this.#passwordHash = passwordHash;
  }
  checkPassword(password: string) {
    return hash(password) === this.#passwordHash;
  }
}
const checker = new PasswordChecker(hash('비밀번호'));
checker.checkPassword('비밀'); // 거짓 반환
checker.checkPassword('비밀번호'); // 참 반환
```

클로저 기법과는 달리 #passwordHash 속성은 클래스 메서드 및 동일한 클래스의 다른 인스턴스에서 접근 가능합니다.

정리

- private 접근 수정자는 타입 시스템을 통해서만 강제됩니다. 런타임중에는 영향을 주지 않으며 단언을 사용하여 우회할 수 있습니다.

- 더 신뢰할 수 있는 정보 숨기기를 위해 클로저 또는 비공개 필드를 사용하는 것이 좋습니다.