Singleton in JavaScript

J105 신준수

Singleton 패턴이란?

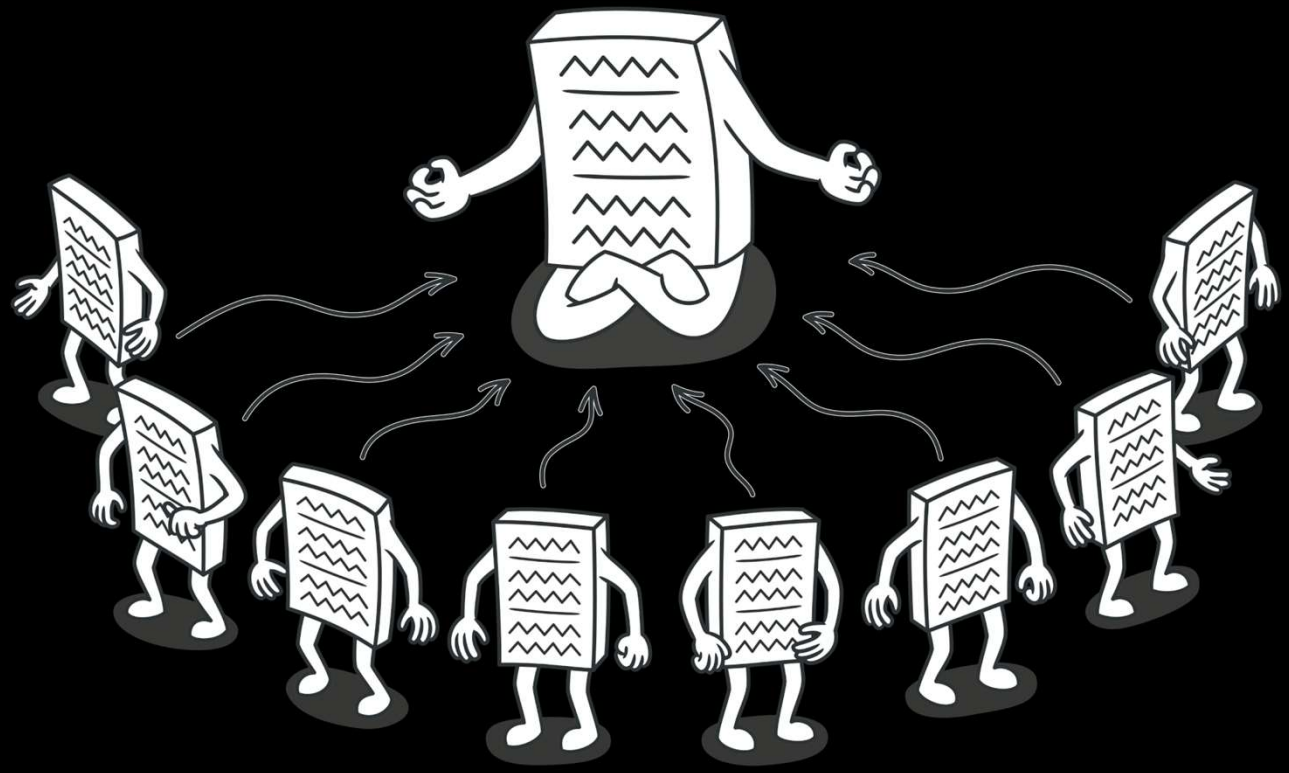


Image source: https://refactoring.guru/design-patterns/singleton

Singleton 패턴이란?

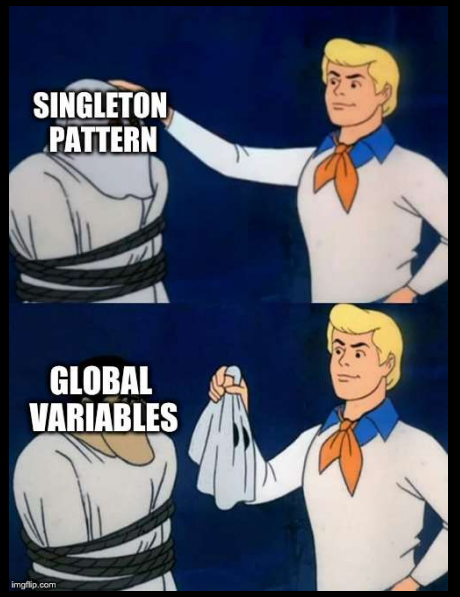
- 단일 인스턴스를 보장하는 디자인 패턴
- 단일 인스턴스에 대한 Global Access를 보장
- 클래스 스스로 인스턴스의 개수를 제어

장점

- 메모리 효율
- 전역변수 namespace 보호
- Lazy initialization 가능

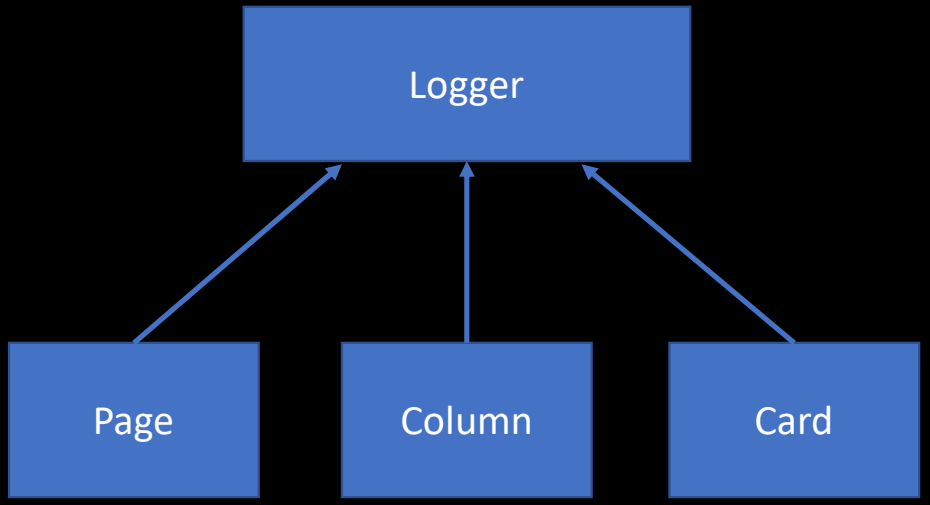
단점

- 클래스간 의존성을 숨김
- 테스트를 어렵게 함
- 화려한 전역변수(?)



언제 사용해야 하나요?

- DB 연결 관리(pool…)
- Socket 관리
- Log 관리
- Facade



…주로 shared resource를 공유해야 할 때!

JavaScript Singleton 구현방법

1. Object Literal
2. Instance module export
3. 외부 instance 변수
4. Class property

구현방법 – Object Literal

singleton > JS object-literal.js > ...

```
1  // logger.js
2
3  module.exports = {
4    write: (log) => {/* write log to file... */},
5    get: () => {
6      /* get logs from file... */
7      return 'this is a log!';
8    }
9  };
```

구현방법 – Export instance

singleton > JS export-variable.js > ...

```
1   // logger.js
2
3   let counter = 0;
4
5   class Logger {
6     constructor() {
7       // do some initialization...
8       counter += 1;
9     }
10
11    write() {
12      // write log to file...
13    }
14
15    get() {
16      // get logs from file...
17      return `this is logger number ${counter}!`;
18    }
19  }
20
21  const instance = new Logger();
22
23  module.exports = instance;
```

구현방법 – 확인하기

singleton > JS sub-module.js > [@] <unknown>

```
1  const objectLiteral = require('./object-literal');
2  const exportedVariable = require('./export-variable');
3
4  module.exports = {
5    objectLiteral,
6    exportedVariable,
7  }
```

submodule에서 싱글턴 import

singleton > JS app-variable.js > ...

```
1  const objectLiteral = require('./object-literal');
2  const exportedVariable = require('./export-variable');
3
4  const subModule = require('./sub-module');
5
6  console.log(objectLiteral === subModule.objectLiteral);
7  console.log(exportedVariable === subModule.exportedVariable);
```

App.js 에서 import한 싱글턴과 비교

PROBLEMS 1 OUTPUT TERMINAL DEBUG CONSOLE

```
Shin@DESKTOP-5V3NKEG MINGW64 /c/신준수/프로그래밍/Boostcamp 2020/멤버십/repos/singleton
$ node app-variable.js
true
true
```

같은 변수임을 확인 가능

구현방법 – 외부 instance 변수

singleton > JS instance-variable.js > ...

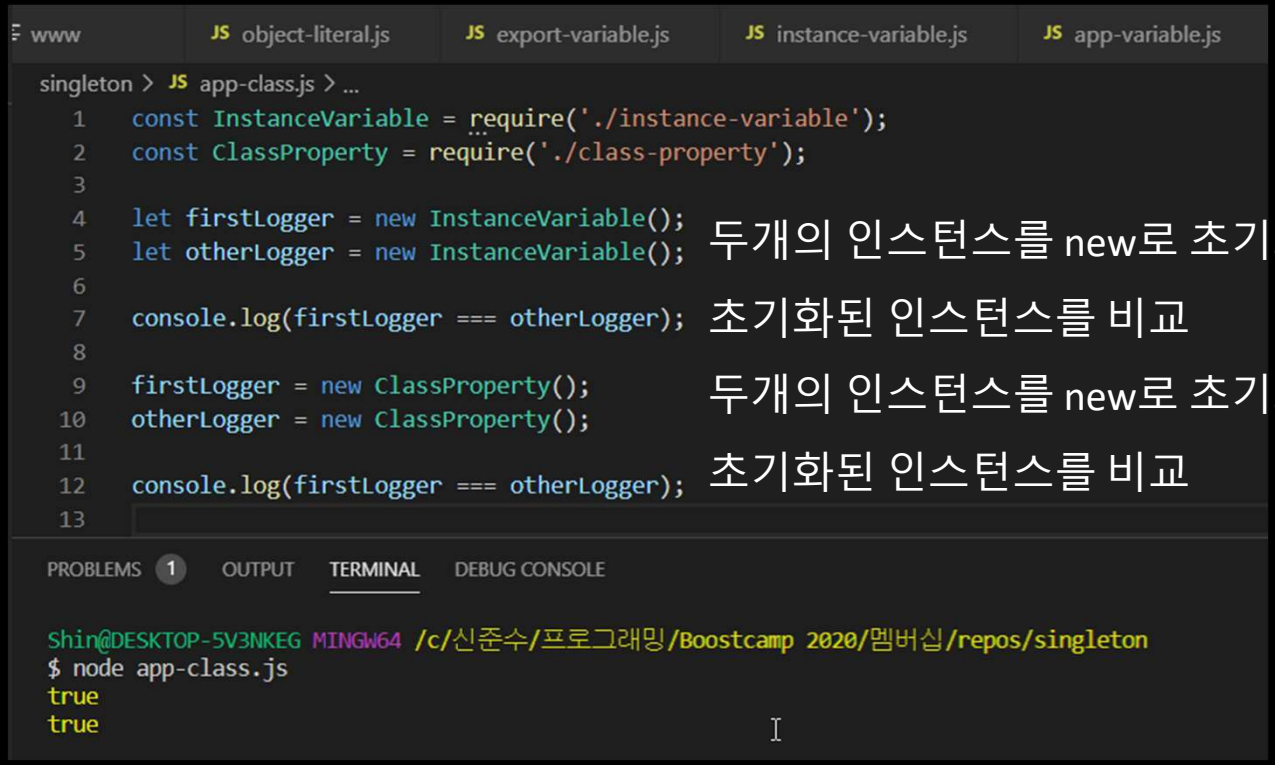
```
1  // logger.js
2
3  let instance;
4
5  class Logger {
6    constructor() {
7      if(instance) return instance;
8      instance = this;
9    }
10
11   write() {
12     // write log to file...
13   }
14
15   get() {
16     // get logs from file...
17     return 'this is a log!';
18   }
19 }
20
21 module.exports = Logger;
```

구현방법 – Static Class Property

singleton > JS class-property.js > Logger > constructor

```
1   // logger.js
2
3   class Logger {
4     static instance; // ECMA 표준이 아님, Experimental Feature (Stage 3)
5
6     constructor() {
7       if(Logger.instance) return Logger.instance;
8       Logger.instance = this;
9     }
10
11    write() {
12      // write log to file...
13    }
14
15    get() {
16      // get logs from file...
17      return 'this is a log!';
18    }
19  }
20
21  module.exports = Logger;
```

구현방법 – 확인하기



두개의 인스턴스를 new로 초기화

초기화된 인스턴스를 비교

두개의 인스턴스를 new로 초기화

초기화된 인스턴스를 비교

같은 인스턴스임을 확인 가능