JS-중급 4

태그	
날짜	@2023년 6월 1일

심볼(symbol)

property key : 문자형

지금까지 객체 프로퍼티는 문자형으로 만들었다

```
const obj = {
    1 : '1입니다.'
    false : '거짓',
}
Object.keys(obj); // ["1","false"]

숫자형이나 불린형으로 만들어도 "1","false" 처럼 문자형으로
반환된다.
```

```
const a = Symbol(); // new를 붙이지 않는다.
const b = Symbol();
유일한 식별자

console.log(a)
Symbol()

console.log(b);
Symbol()

a === b; // false
a == b; // false

Symbol : 유일성 보장

const id = Symbol('id'); // 'id' 설명을 붙혀줄수 있다.
```

```
const id1 = Symbol('id');
// 디버깅이 편해진다.
// 설명이 같아도 비교해보면 다르다고 나온다

> id
Symbol(id)
> id2
Symbol(id)

id === id2 // false
id == id2 // false

-----------------------------------------------------

property key : 심볼형

const id  = Symbol('id');
const user = {
    name : 'Mike',
    age : 30,
    [id] : 'myid'
}

> user
// {name : "Mike", age : 30, Symbol(id) : "myid"}
// user[id] // "myid"

Object.keys(user); 로 확인해보면
["name" ,"age"] 두개만 나온다.

keys, values, entries는 키가 심볼형인것은 건너뛴다.

for(let a in) 도 마찬가지이다.
```

```
Symbol.for() : 전역 심볼

- 하나의 심볼만 보장 받을 수 잇음
- 없으면 만들고, 있으면 가져오기 때문
- 심볼 함수는 매번 다른 심볼 값을 생성하지만,
  symbol.for 메소드는 하나를 생성한 뒤 키를 통해 같은
  symbol을 공유

const id1 = Symbol.for('id');
const id2 = Symbol.for('id');

id1 === id2 ; // true

그냥 심볼과 다르게 true가 나온다
```

```
이름을 알고 싶다면 Symbol.keyFor(id1)을 사용하면
이름을 알 수 있다.

전역 심볼이 아니라면 keyFor를 사용할 수 없다.
대신 description으로 이름을 알 수 있다.

const id = Symbol('id');
id.description; // "id"
```

```
숨겨진 Symbol key 보는법

const id = Symbol('id');

const user = {
    name : 'Mike',
    age : 30,
    [id] : 'myid'
}

심볼을 완전히 숨길 수는 없다.
Object.getOwnPropertySymbols(user); // [Symbol(id)]
심볼들만 볼수 있다.


Reflect.ownKeys(user); // ["name", "age", Symbol(id)]
심볼을 포함한 모든 키를 볼 수 있다.
```

```
// 다른 개발자가 만들어 놓은 객체
const user = {
  name : "Mike",
  age : 30,
}

// 내가 작업
user.showName=function() {};


// 사용자가 접속하면 보는 메시지
for(let key in user){
  console.log(`His ${key} is ${user[key]}.`);
}

객체의 프로퍼티를 순회하면서 아래처럼 찍힌다.
"His name is Mike."
```

```
"His age is 30."
"His showName is function () {}." <- 이거

-----------------------------------------------------
```

이렇게 바꾸면 나오지 않는다

```
const user = {
  name : "Mike",
  age : 30,
}

const showName = Symbol('showName');
user[showName] = function() {
  console.log(this.name);
};

for(let key in user){
  console.log(`His ${key} is ${user[key]}.`);
}

"His name is Mike."
"His age is 30."

-------------------------------------------------

const user = {
  name : "Mike",
  age : 30,
}

const showName = Symbol('showName');
user[showName] = function() {
  console.log(this.name);
};

user[showName]();

for(let key in user){
  console.log(`His ${key} is ${user[key]}.`);
}

"Mike"
"His name is Mike."
"His age is 30."
```

심볼로 만들었기 때문에 forin에서 걸리지 않는다.

다른 개발자가 만든 코드를 건들지 않고 코드를 추가할 수 있다.

이렇게 하면 원래 유저의 이런 이름의 메소드가 있는지 고민할 필요가 없다. 다른 사람이 만들어 놓은 프로퍼티를 덮어씌울 필요도 없다.