삼육대학교 SW융합교육원 김진호

자바스크립트 기초교육

<class>
</body>
<head>
<title>

04. iterable (이터러블)

4-1. iterable

4-2. array and string

4-3. iterable and array like

4. iterable

- ES6 이전의 순회 가능한 데이터 컬렉션, 배열, 문자열, 유사 배열 객체, DOM 컬렉션 등은 통일 된 규약 없이 for문, for...in문, forEach 메서드 등 다양한 방법으로 순회 할 수 있었다.
- ES6에서는 순회 가능한 데이터 컬렉션을 이터레이션 프로토콜을 준수하는 이터러블로 통일하여 for...of문, 스프레드 문법, 배열 디스트럭처링 할당의 대상으로 사용할 수 있도록 일원화했다.

4. iterable

4-1. iterable (이터러블)

4-1-1. iterable (이터러블)

■ 이터러블(iterable) 은 메서드 Symbol.iterator가 구현된 객체이다.

```
let range = {
    from: 1,
    to: 5
};

// Symbol.iterator를 추가한다.
// for..of 최초 호출 시, Symbol.iterator가 호출된다.
range[Symbol.iterator] = function () {

    // Symbol.iterator는 이터레이터 객체를 반환한다.
    return {
        current: this.from,
        last: this.to,

        // for..of 반복문에 의해 반복마다 next()가 호출된다.
        next() {
            // next()는 값을 객체 {done:.., value :...} 형태로 반환한다.
            // done은 반복이 끝났음을 의미하며 끝나지 않았을 경우 value가 다음 값이 된다.
            if (this.current <= this.last) {
                return { done: false, value: this.current++ };
            } else {
                return { done: true };
            }
        }
    };
};

// 객체 선언 후 for..of 반복문을 호출하면 range is not iterable 이라는 오류가 발생한다.
// Symbol.iterator를 추가한 뒤 1~5까지의 숫자가 출력 된다.
for (let num of range) {
    console.log(num);
}
```

4. iterable

4-2. array and string (배열과 문자열)

4-2-1. array and string (배열과 문자열)

- 배열, 문자열 등은 대표적인 이터러블이다.
- 문자열이나 배열 같은 내장 이터러블에도 Symbol.iterator가 구현되어 있어 명시적으로 호출 할 수도 있다.
- 자주 사용하지는 않지만 필요 시 for...of 사용보다 반복 과정을 더 잘 통제할 수 있다.

```
for (let char of "JavaScript") {    // 문자열을 for...of 반복문으로 호출한다.
    console.log(char);
}

let str = "iterable";

let iterator = str[Symbol.iterator]();

while (true) {
    let result = iterator.next();
    if (result.done) break;
    console.log(result.value);
}
```

4. iterable

4-3. iterable and array like (이터러블과 유사 배열)

4-3-1. iterable and array like (이터러블과 유사 배열)

- **이터러블**(iterable) : 메서드 Symbol.iterator가 구현된 객체
- **유사 배열**(array-like) : 인덱스와 length 프로퍼티가 있어서 배열처럼 보이는 객체
- 이터러블이면서 유사 배열일 수 있고 이터러블 객체라고 해서 유사 배열 객체는 아니며 유사 배열 객체라고 해서 이터러블 객체인 것도 아니다. (별도의 개념)
- 이터러블과 유사 배열은 배열 메서드를 사용할 수 없어 불편할 때가 있는데 Array.from을 이용해서 배열로 변경할 수 있다.

4. iterable

4-3. iterable and array like (이터러블과 유사 배열)

4-3-1. iterable and array like (이터러블과 유사 배열)

■ 유사 배열 테스트

```
let arrayLike = {
    0: "배열인듯",
    1: "배열아닌",
    2: "배열같은너",
    length: 3
};

// console.log(arrayLike.pop()); // 배열 메서드 사용 불가 - arrayLike.pop is not a function

//  Array.from()은 넘겨 받은 인수가 이터러블이나 유사 배열인 경우, 새로운 배열을 만들고 객체의 모든 요소를 새롭게 만든 배열로 복사한다.
let arr = Array.from(arrayLike);
console.log(arr.pop());      // 배열 메서드 사용 가능
```

4. iterable

4-3. iterable and array like (이터러블과 유사 배열)

4-3-1. iterable and array like (이터러블과 유사 배열)

■ 이터러블 테스트

```
// 01_iterable에서 복사
// 이터러블(iterable) 은 메서드 Symbol.iterator가 구현된 객체이다.
let range = {
    from: 1,
    to: 5
};

// Symbol.iterator를 추가한다.
// for..of 최초 호출 시, Symbol.iterator가 호출된다.
range[Symbol.iterator] = function () {

    // Symbol.iterator는 이터레이터 객체를 반환한다.
    return {
        current: this.from,
        last: this.to,
    // for..of 반복문에 의해 반복마다 next()가 호출된다.
    next() {
        // next()는 값을 객체 {done:.., value :...} 형태로 반환한다.
        // done은 반복이 끝났음을 의미하며 끝나지 않았을 경우 value가 다음 값이 된다.
        if (this.current<= this.last) {
            return { done: false, value: this.current++ };
        } else {
            return { done: true };
        }
    }
    };
};

let arr2 = Array.from(range);
console.log(arr2.pop());    // 배열 메서드 사용 가능

// Array.from()에는 매핑(mapping) 함수를 선택적으로 넘겨줄 수 있다. 요소 추가 전 각 요소를 대상으로 매핑 함수를 적용해 반환된 값이 추가 된다.
let arr3 = Array.from(range, num => num * num);
console.log(arr3);
```

■ 이터러블은 데이터의 소비자(for...of, 스프레드 문법, 배열 디스트럭처링 할당 등)와 공급자(Array, String, DOM 컬렉션)를 연결하는 인터페이스의 역할을 한다.