제너레이터 함수 (8월 8일)

배정	
상태	8월
속성	@2021년 8월 8일 오후 3:32
언어	

ES6에서 도입된 제너레이터(Generator)는 코드 블록의 실행을 일시 중지했다가 필요한 시점에 재개할 수 있는 특수한 함수이다.

1. 제너레이터 함수는 함수 호출자에게 함수 실행의 제어권을 양도할 수 있음

   ▼ 세부 설명

   함수 호출자는 함수를 호출한 이후 함수 실행을 제어 할 수 없다. 제너레이터 함수는 실행을 함수 호출자가 제어할 수 있다. 다시말해, 함수 호출자가 함수 실행을 일시중지시키거나 재개가 가능함. 함수제어권을 함수가 독점하는 것이 아니라 함수 호출자에게 양도 할 수 있다.

2. 제너레이터 함수는 함수 호출자와 함수의 상태를 주고받을 수 있다.

   ▼ 세부 설명

   함수가 실행되고 있는 동안에는 함수 외부에서 함수 내부로 값을 전달하여 함수의 상태를 변경할 수 없다. 제너레이터 함수는 함수 호출자와 양방향으로 함수의 상태를 주고받을 수 있음.

   다시말해, 제너레이터 함수는 함수 호출자에게 상태를 전달할 수 있고 함수 호출자로부터 상태를 전달받을 수도 있다.

3. 제너레이터 함수를 호출하면 제너레이터 객체를 반환한다.

   ▼ 세부 설명

   일반 함수를 호출하면 함수 코드를 일괄 실행하고 값을 반환한다. 제너레이터 함수를 호출하면 함수 코드를 실행하는 것이 아니라 이터러블이면서 동시에 이터레이터인 제너레이터 객체를 반환함.

`function*` 키워드로 선언하고 하나 이상의 **yield 표현식**을 포함한다.

```
// 제너레이터 함수 선언문
function* genDecFunc() {
  yield 1;
}

// 제너레이터 함수 표현식
const genExpFunc = function* () {
  yield 1;
};

// 제너레이터 메서드
const obj = {
  * genObjMethod() {
    yield 1;
  }
};

// 제너레이터 클래스 메서드
class MyClass {
  * genClsMethod() {
    yield 1;
  }
}

--------------------------------------------------------------------------------
// 애스터리스크(*)의 위치는 function 키워드와 함수 이름 사이라면 어디든지 상관없다.
// 하지만 일관성을 유지하기위해 function 키워드 바로뒤에 붙이는 것을 권장한다.

function* genFunc() { yield 1; }

function * genFunc() { yield 1; }

function *genFunc() { yield 1; }

function*genFunc() { yield 1; }

--------------------------------------------------------------------------------
// 제너레이터 함수는 화살표 함수로 정의 할수 없음.

const genArrowFunc = * () => {
  yield 1;
}; // SyntaxError: Unexpected token '*'

--------------------------------------------------------------------------------
// 제너레이터 함수는 new 연산자와 함께 생성자 함수로 호출 할 수 없다.
function* genFunc() {
  yield 1;
}

new genFunc(); // TypeError: genFunc is not a constructor
```

제너레이터 객체

제너레이터 함수를 호출하면 일반 함수처럼 함수 코드 블록을 실행하는 것이 아니라 제너레이터 객체를 생성해 반환한다. 제너레이터 함수가 반환한 제너레이터 객체는 이터레이블이면서 동시에 이터레이터다.