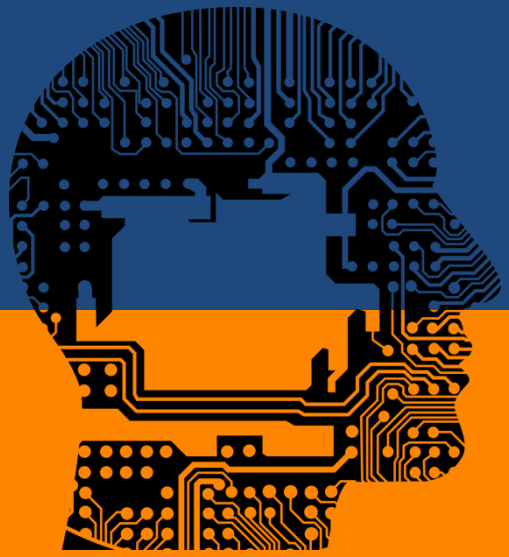
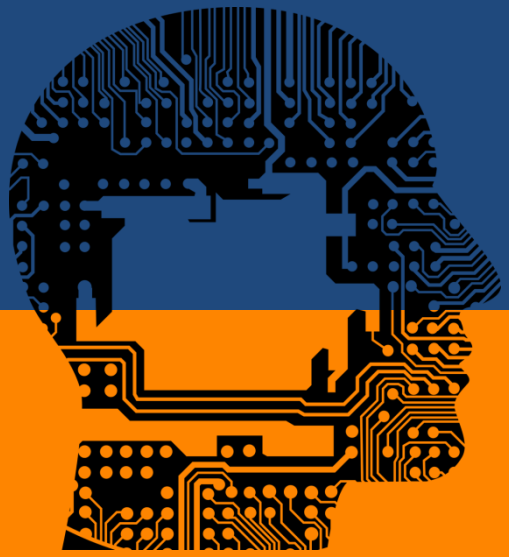
한국IT진흥부설
정보보호교육학원 아이섹



JAVA 웹 개발자 양성과정
Javascript
9강 - 다양한 형태의 함수와 스코프

By SoonGu Hong

한국IT진흥부설
정보보호교육학원 아이섹



JAVA 웹 개발자 양성과정

Javascript

1. 다양한 형태의 함수

* 즉시 실행 함수(IIFE: Immediately Invoked Function Expression)

```
(function () {
    // ....
}());
```

- 함수 정의와 동시에 즉시 호출되는 함수를 즉시 실행함수라고 합니다.
- 즉시 실행 함수는 단 한 번만 호출되며 다시 호출할 수 없습니다.

* 재귀 함수(Recursive Function)

```
function countdown(n) {
    if (n < 0) return;
    console.log(n);
    countdown(n - 1);
}
```

- 함수가 자기 자신을 호출하는 것을 재귀 호출이라 합니다. 재귀 함수란 자기 자신을 호출하는 재귀 호출을 수행하는 함수를 말합니다.
- 재귀 함수는 자신을 무한으로 재귀 호출하기 때문에 반드시 **탈출 조건**을 만들어야 합니다.

* 중첩 함수(Nested Function)

```
function outer() {
    var x = 1;

    function inner() {
        var y = 2;
        console.log(x + y);
    }
    inner();
}
```

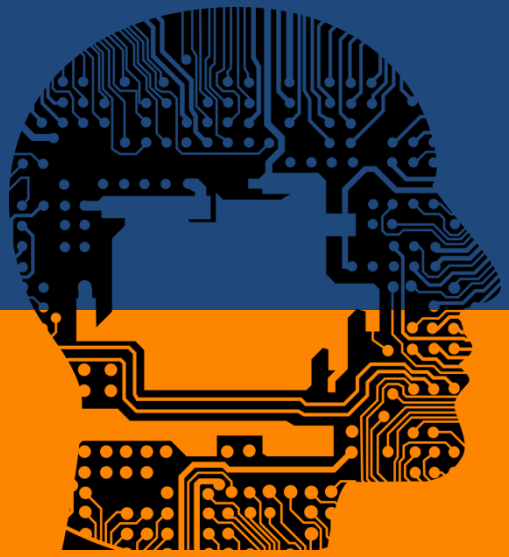
- 함수 내부에 또 정의된 함수를 중첩 함수 또는 내부 함수라고 부릅니다. 중첩 함수는 외부 함수 내부에서만 호출할 수 있습니다.
- 일반적으로 중첩함수는 자신을 감싸고 있는 외부함수를 돕는 헬퍼 함수의 역할을 합니다.

* 콜백 함수(Callback Function)

```
function showNumbers(n, callback) {
    for (var i = 1; i <= n; i++) {
        callback(i);
    }
}
```

- 함수의 매개변수를 통해 다른 함수의 내부로 전달되는 함수를 콜백 함수라고 하며, 매개 변수를 통해 외부에서 콜백 함수를 전달받은 함수를 고차 함수라고 합니다.

한국IT진흥부설
정보보호교육학원 아이섹



JAVA 웹 개발자 양성과정

Javascript

2. 스코프(scope)

* 스코프(scope)란?

```
function foo(x) {
    var y = 3;
    console.log(x); //x 참조가능
    console.log(y); //y 참조가능
    return x + y;
}
foo(5);
console.log(x); //x 참조 불가능
console.log(y); //y 참조 불가능
```

- 모든 식별자(변수, 함수, 클래스 등)는 자신이 선언된 위치에 의해 다른 코드가 자신을 참조할 수 있는 유효 범위가 결정되는데 이를 스코프라고 부릅니다

* 전역(global) 스코프와 지역(local) 스코프

```
var x = 'global';

function func() {
    var x = 'local';
    console.log(`x: ${x}`); // ?
}
console.log(`x: ${x}`); // ?
func();
```

- 각각 어떤 값이 출력될까요?

* 전역 스코프와 전역 변수

```
var x = 'global x';
var y = 'global y';

function outer() {
    console.log(x);
    console.log(y);

    function inner() {
        console.log(x);
        console.log(y);
    }
    inner();
}
outer();
console.log(x);
console.log(y);
```

- 코드의 가장 바깥쪽 영역에서 선언된 변수를 전역 변수라고 부르며, 전역 변수는 코드 전체의 전역 스코프에서 참조가 가능합니다.
- 전역 변수는 코드 모든 부분에서 사용이 가능합니다.

* 지역 스코프와 지역 변수

```
var x = 'global x';
function outer(x) {
    var y = 'outer local y';
    console.log(x);
    console.log(y);
    // console.log(z); //참조 불가능

    function inner() {
        var z = 'inner local z';
        console.log(x);
        console.log(y);
        console.log(z);
    }
    inner();
}
outer('outer local x');
// console.log(x);  //모두 참조 불가능
// console.log(y);
// console.log(z);
```

- 지역이란 함수 몸체 내부를 말합니다. 지역 변수는 자신이 선언된 지역과 하위 지역(중첩 함수)에서만 참조할 수 있습니다.
- 즉, 지역 변수는 자신의 지역 스코프와 하위 지역 스코프에서만 유효합니다.
- 전역 변수와 지역 변수의 이름이 같을 경우 함수 내부에선 지역 스코프를 먼저 참조합니다.

* 전역 변수의 문제점

1. 암묵적 결합

- 전역 변수를 선언한 의도는 모든 코드가 전역 변수를 참조하고 변경할 수 있는 암묵적 결합(Implicit coupling)을 허용하는 것입니다.
- 변수의 유효 범위가 클 수록 코드의 가독성이 나빠지고 의도치 않게 상태가 변경될 수 있는 위험성도 높아집니다.

2. 긴 생명 주기

- 전역 변수는 살아있는 시간이 길어서 메모리 리소스도 오랜 기간 소비합니다. 더욱이 var 키워드는 변수의 중복 선언을 암묵적으로 허용해주므로 변수 이름이 중복된 전역 변수는 의도치 않는 재할당으로 인해 변경이 일어납니다.

3. 스코프 체인 상에서 종점에 존재

- 변수를 검색할 때 지역 스코프부터 검색하므로 전역 변수는 검색속도가 가장 느립니다.

감사합니다
THANK YOU