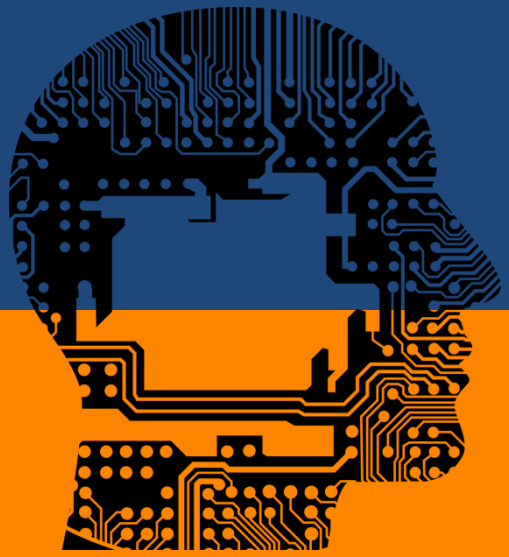
한국IT진흥부설 정보보호교육학원 아이섹

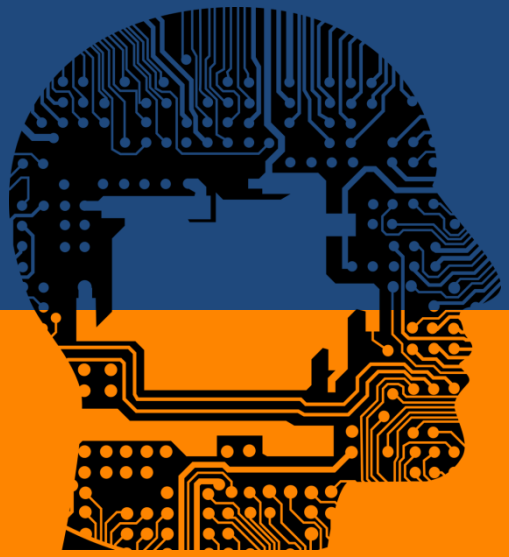


JAVA 웹 개발자 양성과정
Javascript

2강 - 자바스크립트 기본구조

By SoonGu Hong

한국IT진흥부설
정보보호교육학원 아이섹



JAVA 웹 개발자 양성과정

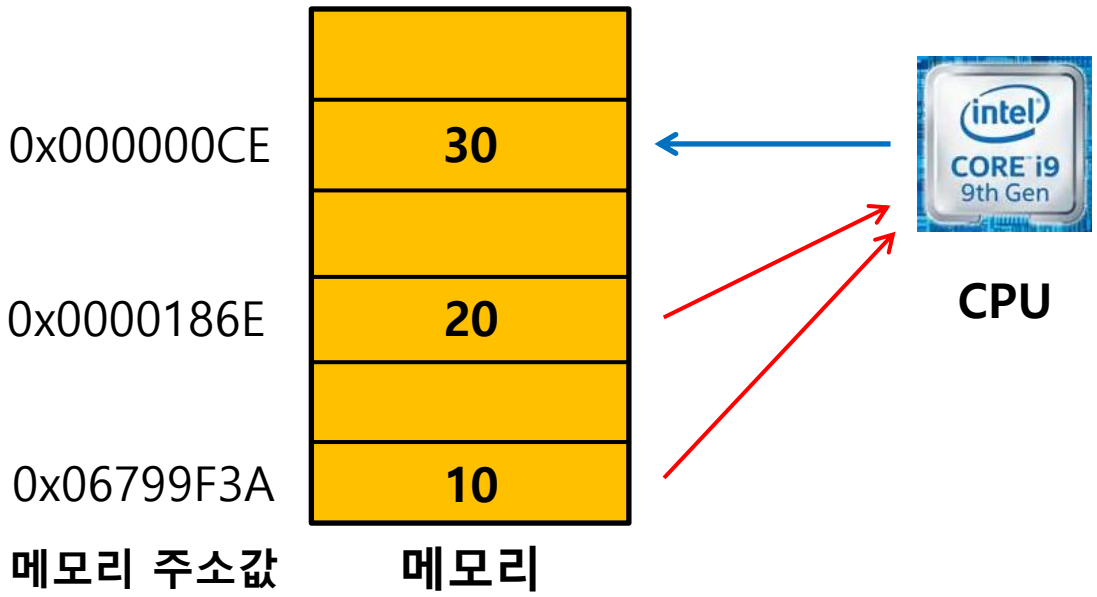
Javascript

1. 변수(variable)

* 변수가 필요한 이유

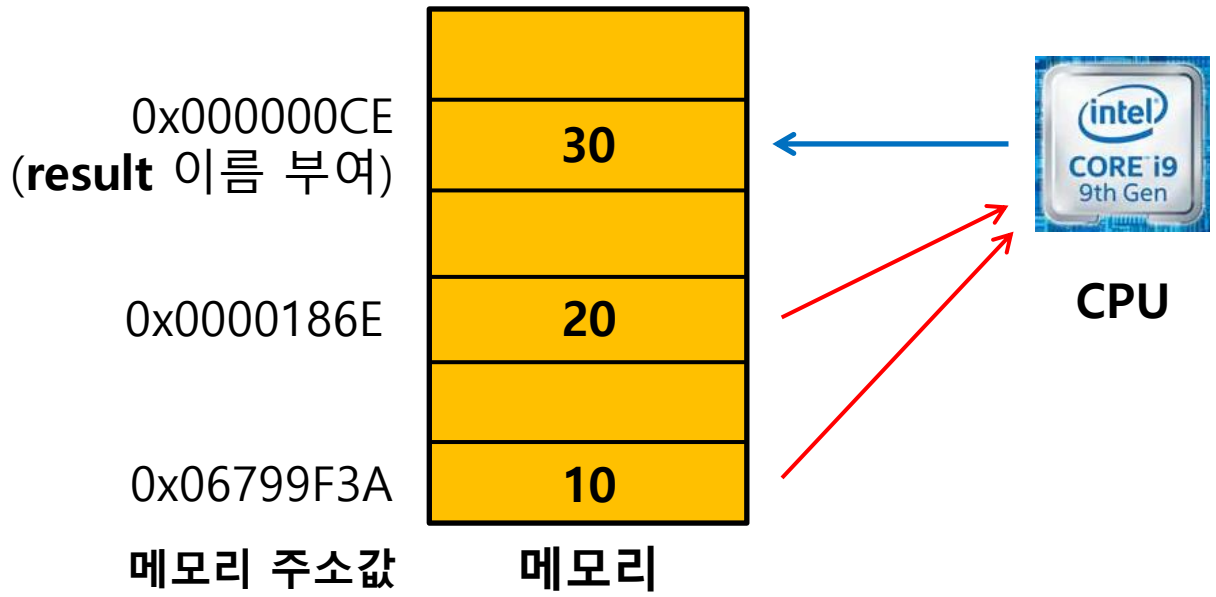
10 + 20

- 여러분은 위 수학식을 보고 뭐가 떠오르나요? 아마도 30이 떠오르실겁니다.
- 그렇다면 컴퓨터에게 같은 질문을 한다면 어떨까요??



* 변수가 필요한 이유

- 여기서 문제점이 하나 생깁니다. 컴퓨터는 10과 20의 합인 30을 정확히 연산했으며 메모리에 저장도 했습니다.
- 그런데 개발자가 30이란 값을 사용하기 위해서는 30이 저장된 곳의 메모리 주소값을 알아야 한다는 것이죠...
- 이 때 변수라는 개념을 사용해서 30이 저장된 공간에 사람이 식별하기 쉬운 이름을 달아줍니다. (ex: result)
- 우리는 이제부터 result라는 이름을 통해 30이란 값을 사용할 수 있게 됩니다.



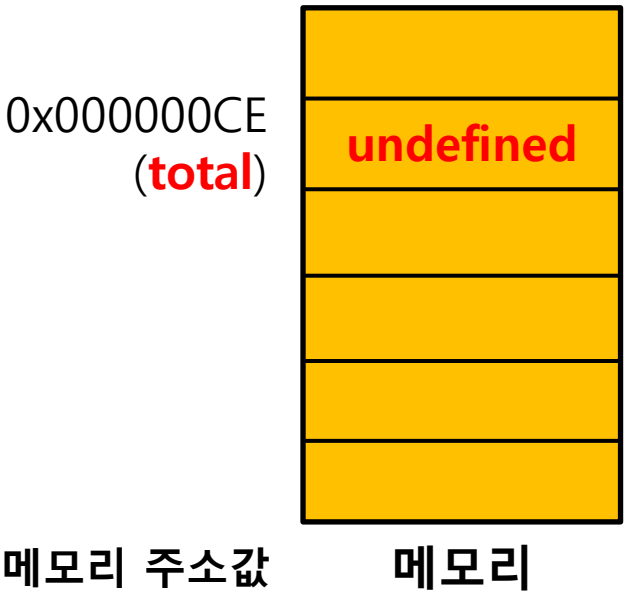
* 변수의 선언

- 변수를 선언한다는 것은 메모리에 데이터를 저장하는 공간을 확보하는 것을 의미합니다.
- 변수를 선언할 때는 **var, let, const** 키워드를 사용합니다. ES6에서 let, const가 도입되기 전까지는 모두 var를 사용했습니다.
- 우선은 var를 사용하여 변수를 선언해보고 차후에 let, const를 설명하겠습니다.

< 변수 선언 방법>

var 변수이름;

ex) var total;



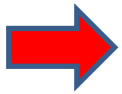
* 변수의 초기화와 할당

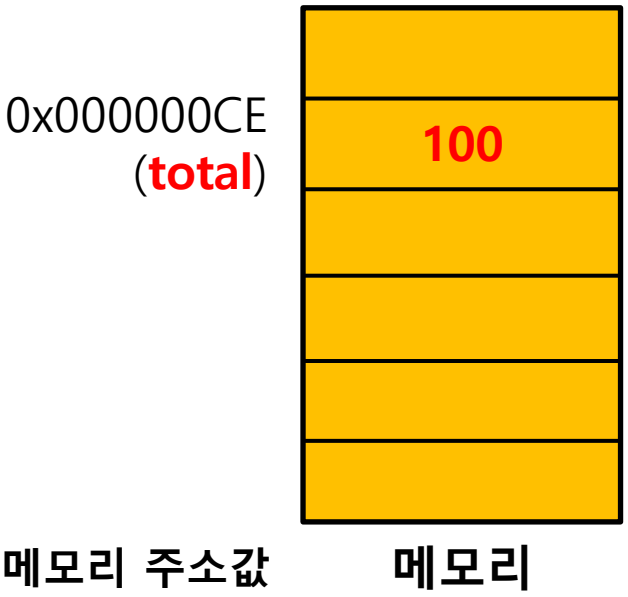
- 변수를 선언만 하고 값을 저장하지 않으면 일단 정의되지 않았다는 뜻을 가진 값인 undefined가 할당됩니다.
- 특정 값을 저장하고 싶으면 할당(대입)연산자 = 을 사용하여 값을 할당해야 합니다.
- 변수의 선언과 함께 동시에 값을 할당할 수도 있습니다.
  ex) var age = 27;

< 변수 할당 방법>

var 변수이름;
변수이름 = 값;

ex) var total;
    total = 100;





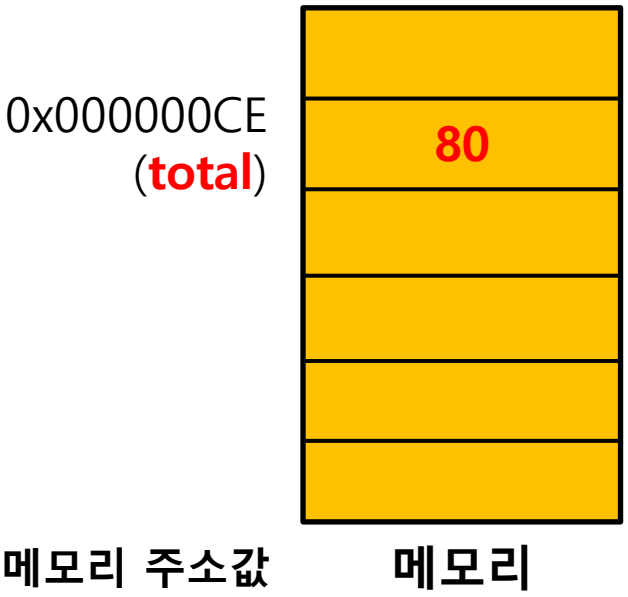
* 변수의 재할당

- 변수(variable)는 이름에서 알 수 있듯이 변경 가능한 데이터를 말합니다.
- 따라서 언제든지 저장된 값을 변경할 수 있습니다. 이를 변수의 재할당이라고 합니다.
- 이에 반대되는 개념으로 한번 할당된 값을 변경할 수 없다면 해당 개념을 상수(constant)라고 합니다.

< 변수 재할당 방법>

var 변수이름 = 값;
변수이름 = 변경값;

ex) var total = 100;
total = 80;



* 식별자 이름 규칙

1. 식별자 이름은 중복을 허용하지 않고 대/소문자를 구분!

2. 식별자 이름은 숫자만으로 구성하거나 숫자로 시작하면 안됨!

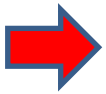
3. 식별자 이름에 공백을 포함할 수 없음!

4. 식별자 이름에는 특수문자를 사용하면 안됨(예외: _, $)

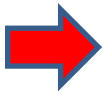
5. 키워드(예약어)는 식별자 이름으로 사용 불가능!

* 네이밍 컨벤션(관례)

```
//네이밍 컨벤션(관례)

//카멜 케이스(camelCase)
var userPassword;

//스네이크 케이스(snake_case)
var user_password;

//파스칼 케이스(PascalCase)
var UserPassword;

//헝가리안 케이스(type+HungarianCase)
var strUserPassword; //str : 문자형
var intLuckyNumber;  //int : 정수형
var $button = document.getElementById('my-btn'); //$ : DOM노드
```

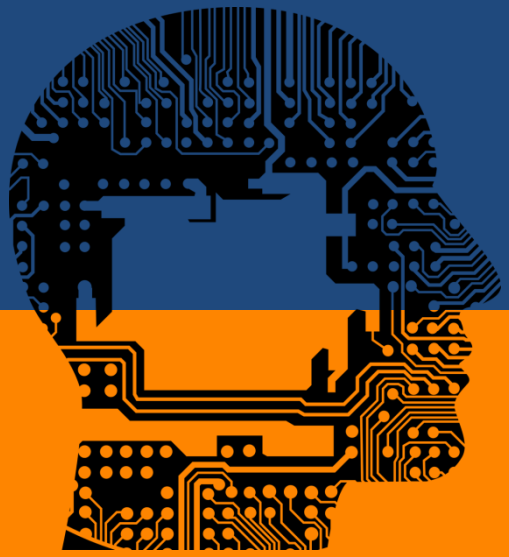


변수, 함수 이름에 사용



클래스, 생성자 이름에 사용

한국IT진흥부설
정보보호교육학원 아이섹



JAVA 웹 개발자 양성과정

Javascript

2. 표현식과 문

* 값(value)

- 값(value)이란 표현식(expression)이 평가(evaluate)되어 생성된 결과입니다.

```
//10 + 20이라는 표현식은 평가되어 숫자 값 30을 생성함
10 + 20; //30
```

- 모든 값은 데이터 타입(data type)을 가지며 메모리에 2진수 비트의 나열로 저장됩니다.
- 예를 들어, 메모리에 저장된 값 01000001을 숫자로 해석하면 10진수로 65이지만 문자로 해석하면 A입니다.

* 리터럴(literal)

- 리터럴은 사람이 이해할 수 있는 문자 또는 약속된 기호를 사용해 값을 생성하는 표기법을 말합니다.

리터럴	예시	비고
정수 리터럴	100	
실수 리터럴	54.12	
2진수 리터럴	0b00001010	
8진수 리터럴	0o74	ES6 도입
16진수 리터럴	0xAC02	ES6 도입
문자열 리터럴	'안녕', "메롱"	
논리 리터럴	true, false	
객체 리터럴	{ name: 'Hong', age: 20 }	
배열 리터럴	[10, 20, 30, 40]	
함수 리터럴	function() {}	

* 표현식(expression)

- 표현식은 값으로 평가될 수 있는 문(statement)입니다.
- 즉, 표현식이 평가되면 새로운 값을 생성하거나 기존 값을 참조합니다.
- 앞 장에서 살펴본 리터럴은 값으로 평가됩니다.
- 따라서 리터럴도 표현식입니다.

```
//리터럴 표현식
10
'안녕'

//식별자 표현식
total
pet.owner
scores[3]

//연산자 표현식
10 + 20
age > 19

//함수 호출 표현식
alert()
confirm()
```

* 문(statement)

- 문(statement)은 프로그램을 구성하는 기본 단위이자 최소 실행 단위입니다.

- 문의 집합으로 이뤄진 것을 프로그램이라고 하며, 문을 작성하고 순서에 맞게 나열하는 것을 프로그래밍이라 합니다.

```
//변수 선언문
var age;

//할당문
age = 10;

//함수 선언문
function hello() {}

//조건문
if (age > 19) {
    console.log('성인입니다.');
}

//반복문
for (var i = 0; i < 5; i++) {
    console.log(i);
}
```

* 세미콜론(;)과 세미콜론 자동 삽입 기능(ASI)

- 세미콜론(;)은 문의 종료를 나타냅니다.
- 자바스크립트 엔진은 세미콜론으로 문이 종료한 위치를 파악하고 순차적으로 하나씩 문을 실행합니다.
- 단, 문들을 중괄호로 묶은 코드 블록({ ... }) 뒤에는 세미콜론을 붙이지 않습니다.
- 자바스크립트에서 세미콜론은 옵션입니다. 엔진 자체적으로 문의 끝 지점에 세미콜론을 넣어주는 세미 콜론 자동 삽입 기능(ASI: automatic semicolon insertion)이 있기 때문입니다.
- 하지만 ASI기능이 잘못 삽입하는 경우도 발생하므로 항상 붙이는 것을 권장합니다.

```
//ASI가 잘못 동작한 예시
function hello() {
    return
      {}

    //개발자의 예측 => return {};
    //ASI 동작 결과 => return; {};
}
```

감사합니다

THANK YOU