

# javascript개념정리

## ❖ JAVASCRIPT(코어언어) - 객체지향언어 핵심

### ◆ 1. 인터프리터 방식

- 작성된 소스 코드를 한 줄씩 읽어 즉시 실행하는 프로그램
- 실행 속도는 상대적으로 느림 (매번 해석해야 하니까)
- 대신 실행 결과를 즉시 볼 수 있어 개발/디버깅에 편리함
- 대표 사례: JavaScript(브라우저 엔진), Python, PHP, Ruby

예시 흐름

소스코드 → [인터프리터] → 즉시 실행

### ◆ 2. 컴파일러 방식

- 작성된 소스 코드를 컴퓨터가 이해할 수 있는 기계어(다른 언어)로 변환하는 프로그램
- 실행 속도는 빠름 (이미 기계어로 변환되어 있음)
- 대신 코드 변경 시 다시 컴파일해야 함
- 대표 사례: C, C++, Go, Rust

예시 흐름

소스코드 → [컴파일러] → 기계어(.exe) → 실행

### ◆ 3. 혼합형 (인터프리터 + 컴파일러)

현대 언어들은 둘을 섞어서 사용합니다.

- **Java** → 소스코드를 \*\*바이트코드(.class)\*\*로 컴파일 → JVM이 인터프리터/즉시 컴파일(JIT) 실행

- **JavaScript (V8 엔진)** → 인터프리팅 + JIT 컴파일 혼합으로 최적화
- 

## ✓ 정리

- **인터프리터** → 실행 빠르게 시작, 속도는 느림
- **컴파일러** → 실행 전 변환 필요, 실행 속도는 빠름
- **현대 언어** → 두 방식을 섞어 "빠른 실행 + 개발 편의성" 모두 잡음

## ❖ JS의 라이브러리, 프레임워크 개념잡기

### ◆ 라이브러리 (Library) - 유연함

- 도서관에서 필요한 책만 빌려오는 것
  - 개발자가 필요할 때 불러서 쓰는 도구 모음
  - 제어권이 개발자에게 있음
  - 예: jQuery, Lodash, Chart.js
- 

### ◆ 프레임워크 (Framework) - 유지보수

- 집 전체를 빌려주는 것
- 이미 짜여진 집(구조, 틀) 안에서 방을 꾸미고 살아야 함
- 제어권이 **프레임워크에 있음** (개발자가 그 안에 맞춰 코드를 작성)
- 예: React, Angular, Django, Spring

## EX) 시계를 만든다고 가정했을때

- 조립하는 사람은 개발자로 통칭

- 객체 = 부품(시계의 시침, 분침, 초침)
- 메서드 = 부품의 동작(돌아가기, 움직이기)

- 개발자 = 부품을 조립해 시계를 만드는 사람

## ◆ **window** 객체 - (카멜표기법)

- 브라우저에서 가장 최상위 객체(전역 객체)
- 브라우저 창(탭) 자체를 의미
- `alert()`, `setTimeout()`, `console.log()` 같은 전역 함수도 사실 `window` 의 메서드
- 예:

```
window.alert("Hello"); // alert 실행
console.log(window.innerWidth); // 브라우저 창 가로 크기
```

## ◆ **document** 객체 - (단일 식별자)

- `window` 안에 포함된 객체 중 하나
- \*현재 열려 있는 웹 문서(HTML 문서)\*\*를 가리킴
- HTML 요소에 접근하거나 수정할 때 사용
- 예:

```
document.title = "새 제목";
document.getElementById("app").innerText = "변경된 내용";
```

## 네이밍 규칙 비교와 이해

- 카멜 표기법(**camelCase**) → `innerHTML`, `getElementById` 처럼 단어를 이어쓸 때 첫 단어는 소문자, 뒤 단어는 대문자 시작
- 파스칼 표기법(**PascalCase**) → `Date`, `ArrayBuffer` 처럼 클래스/생성자 이름에 주로 사용
- 스네이크 표기법(**snake\_case**) → `max_value` 처럼 언더스코어로 구분

- **document** → 단어가 하나뿐이라 그냥 소문자로 쓰인 **단일 식별자**로 구분

## ❖ **Function = 함수(기능) 호출**

### ◆ 개념

- \*매개변수(Parameter)\*\*는 함수가 호출될 때 외부에서 값을 전달받아 사용하는 변수예요.
- 함수의 "입구"에서 값을 받아주는 그릇이라고 보면 된다.

### ◆ 예시 (자바스크립트)

```
function greet(name) { // ← name이 '매개변수(Parameter)'  
  console.log("안녕하세요, " + name + "님!");  
}  
  
greet("철수"); // ← "철수"는 '인자(Argument)'
```

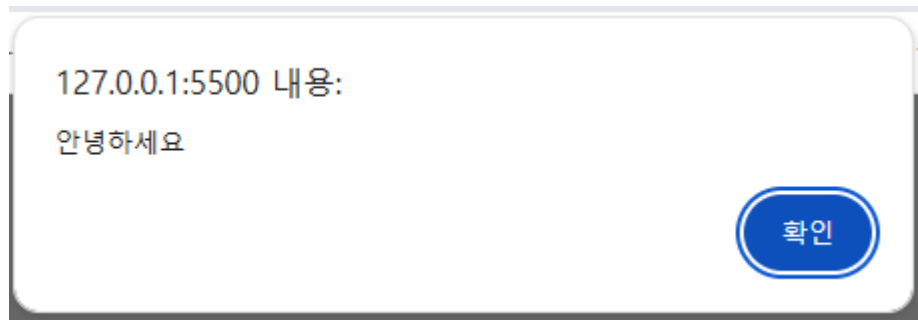
### ◆ 쉽게 비유

- 함수(function) = 자판기
- 매개변수 (Parameter)= 투입구 (돈 넣는 구멍)
- 인자(Argument) = 내가 넣은 동전
- 리턴값(Return value) = 자판기에서 나온 음료수

**결과값 (Result)** → 함수 실행의 최종 산출물

- 리턴값이 될 수도 있고
- 기획 의도라고 말할수도있다.

- 판매자가 음료수를 팔아 얻는 수익이라 \*칭할수도있음 - 판매하는이유 - 수익



//window 객체 = 최상위 객체 = 객체명 생략 가능하여

```
//window.alert("안녕하세요");
alert("안녕하세요");
```

\*같은결과물이 나온다.

alert은 1회성 document위에있는 경고창 메서드이다.  
윈도우내에서 제공하는것 이라고 생각하면 편함

## ❖ 변수(var) : 변할수 있는 데이터

### ◆ 1. 기본적으로 변수에는 하나의 값만 저장

```
let x = 10; // 숫자 1개
let name = "철수"; // 문자열 1개
```

let은 블록 범위(block scope)를 가지는 변수 선언 키워드

왼쪽(name) → 변수 이름(Key, Label)

오른쪽("철수") → 값(Value, Data)

key = value 구조

## ◆ 2. 여러 개를 넣고 싶을 때 → 배열(Array), 객체(Object) 활용

// 배열: 여러 개의 데이터 순서대로 저장

```
let numbers = [1, 2, 3, 4, 5];
```

// 객체: 여러 개의 데이터(키-값 쌍) 저장

```
let user = { name: "철수", age: 20, city: "서울" };
```

배열과객체는 결국 변수에 “하나의 값”으로 저장되지만, 그 값 안에 여러 데이터가 구조적으로 포함되는 것.

많이들 “변수는 데이터를 담는 상자”라고 배우지만, 실제로는 값 자체는 메모리에 저장되고, 변수는

그 메모리 주소를 가리키는 이름표라고 말한다.

## 1. 숫자(Number, 숫자열이라고도 함)

- 숫자로 된 데이터
- 연산(+, -, \*, / 등)이 가능
- 프로그래밍에서 보통 `Number` 타입으로 분류

예시 (JavaScript)

```
let age = 25; // 숫자
```

```
let price = 19.99; // 실수
```

```
console.log(age + 5); // 30 (덧셈 가능)
```

## ◆ 2. 문자열(String)


- 문자들의 집합(텍스트 데이터)
- `'` 또는 `"` 또는 백틱(````)으로 감싼 값
- 글자이므로 수학적 연산은 안 되지만, 이어붙이기(+) 가능

예시 (JavaScript)


```
let name = "철수";
let greet = "안녕, " + name; // 문자열 결합
console.log(greet); // "안녕, 철수"
```

### ◆ 3. 차이 정리

구분	숫자(Number)	문자열(String)
형태	10, 3.14	"10", "철수", 'Hello'
연산	사칙연산 가능	+로 이어붙이기 가능
용도	계산, 수치 표현	텍스트, 메시지 표현



## JavaScript (ES6) code snippets

[charalampos karypidis](#) |  19,982,543 installs | ★★★★★ (45) | Free

Code snippets for JavaScript in ES6 syntax

[Install](#)
[Trouble Installing?](#)

**vscode - 확장 - JavaScript (ES6) code (snippets / 자동완성)**

```

<script>
  //a.변수선언
  var box;
  //b. 변수 초기값
  box = 10; // Type : number
  //c. 초기값 치환 대입
  box = "200"; // Type : string
  //d.변수 사용
  console.log(box);
  console.log(typeof (box));
</script>

```

console.log(typeof box);

typeof = 이것의 타입은 뭐야?

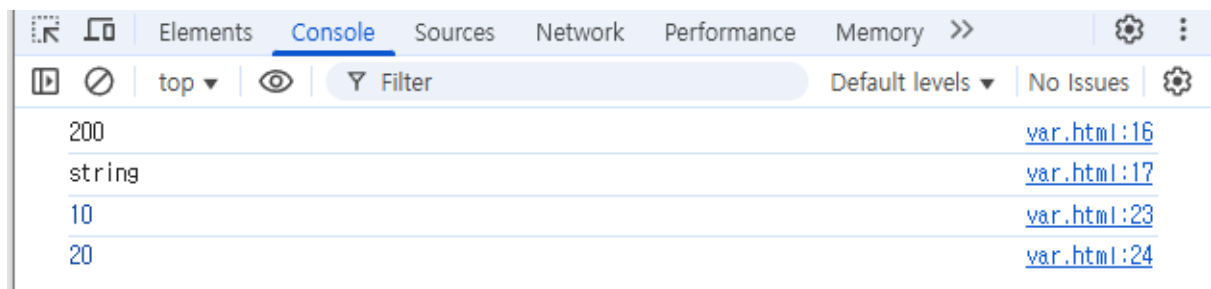
변수 초기값은 number였다가

"" 인용구 를 추가하며 string으로 type을 변환한것을 알수가 있다.

```

//실습1 변수명 num1, num2에 각각 숫자 10 과 20을 할당하세요. 콘솔 출력
var num1,num2;
num1 = 10;
num2 = 20;
console.log(num1, num2);

```



## ❖자바스크립트의 기본 자료형(Primitive Types)



## ◆ 1. **undefined**

- 값이 아직 정의되지 않음을 의미
- 변수를 선언했지만, 초기화하지 않았을 때 자동으로 들어가는 값
- 자바스크립트 엔진이 기본으로 넣어주는 "없음"

```
let a;  
console.log(a); // undefined
```

👉 "값을 아직 안 넣었다"라는 상태

## ◆ 2. **null**

- 개발자가 의도적으로 "값 없음"을 명시하는 값
- 즉, "여기에 값이 없다는 걸 내가 일부러 지정"
- 객체를 담을 자리에 "지금은 비어있다"라고 표현할 때 자주 사용

```
let b = null;  
console.log(b); // null
```

👉 "여기는 비워둔다"라는 개발자의 의도

```

//실습1. 변수명 num1, num2에 각각 숫자 10 과 20을 할당하세요. 콘솔 출력
var num1,num2;
num1 = 10;
num2 = 20;
console.log(num1, num2);

var a = null; // null : 값이 없다
var b = undefined; // undefined : 값이 정의되지 않음
var c = true; // boolean : 논리값 (true, false)

//document는 body 태그 나오는곳 웹브라우저는 기본적으로 다 문자열이다.

// string : 문자열
var str = "HelloJavascript + 자바스크립트";
// string : 숫자열

var str2 = 1000;
document.write(str);
console.log(typeof(str));
console.log(typeof(str2));

var num3 = 100;
var num4 = 200;
var result2 = num3 + num4; // + : 산술연산자 (더하기)

document.write("<br><br>" + result2);

// 문자열 결합 "" + : 연결연산자
document.write("<br><br>두합의 결괏값은 : " + result2 + "입니다!!");

//실습2. 변수명 number1, number2에 숫자형 데이터 50,80 할당 result3 변수에
// 두수의 합을 저장하고 결과를 출력하기
var number1, number2 ,result3;
number1 = 50;
number2 = 80;
result3 = number1 + number2;
document.write("<br><br>계산의 결괏값은 : " + result3 + "입니다.");

```

HelloJavascript + 자바스크립트

300

두합의 결괏값은 : 300입니다!!

계산의 결괏값은 : 130입니다.

### ◆ 3. Boolean (불린)

- 논리형 데이터 타입
- 값은 오직 두 가지 → `true` (참) 또는 `false` (거짓)
- 주로 조건문(`if`, `while`)에서 사용

#### ✓ 기본 예시

```
let isAdult = true;  
let isAdmin = false;  
  
console.log(isAdult); // true  
console.log(typeof isAdmin); // "boolean"
```

#### ✓ 비교 연산의 결과는 Boolean

```
console.log(10 > 5); // true  
console.log(10 < 5); // false  
console.log(10 === 10); // true
```

#### ✓ 논리 연산자와 함께 사용

```
let a = true;
let b = false;

console.log(a && b); // false (AND)
console.log(a || b); // true (OR)
console.log(!a);    // false (NOT)
```

## ✅ Truthy / Falsy 개념

자바스크립트에서는 **boolean**이 아닌 값도 조건문에서 자동으로 **true** / **false** 로 평가된다

- **Falsy (false로 취급되는 값):** 0, "", null, undefined, NaN, false
- **Truthy (true로 취급되는 값):** 그 외 모든 값

```
if ("Hello") {
  console.log("실행됨"); // "Hello"는 Truthy → 실행
}
if (0) {
  console.log("실행 안 됨"); // 0은 Falsy → 실행 안 됨
}
```

## ✅ 정리

- Boolean은 참/거짓을 표현하는 자료형
- 조건문/반복문에서 핵심적 역할
- 실제 코드에서는 비교/논리 연산으로 많이 만들어짐

## ◆ Boolean (불리언) 활용 예시

```
let age = 20;
let isAdult = (age >= 18); // true
console.log(isAdult);      // true

if(isAdult) {
```

```
console.log("성인입니다.");  
} else {  
  console.log("미성년자입니다.");  
}
```

## ✅ 정리

- 숫자(Number) = 수치 데이터
- 문자열(String) = 문자 데이터
- 불리언(Boolean) = 참/거짓 논리값 (조건 처리에 필수)
- null / undefined = 값이 없음 (의도적 / 자동)

## ◆ 1. = (할당 연산자, Assignment)

- 오른쪽 값을 왼쪽 변수에 저장
- 프로그래밍에서 = 는 수학의 같다(=) 와 다르게 할당 의미입니다.

```
let x = 10; // x는 10이다
```

## ◆ 2. == (동등 비교, Equality)

- 값이 같은지만 비교 (타입은 강제 변환 후 비교)

```
console.log(10 == "10"); // true (숫자 10과 문자열 "10"을 같은 값으로 취급)
```

## ◆ 3. === (일치 비교, Strict Equality)

- 값과 타입까지 모두 같아야 true

```
console.log(10 === "10"); // false (숫자와 문자열 타입이 다름)  
console.log(10 === 10); // true
```

## ✅ 정리

- `=` → 할당(Assign)
- `==` → 값만 같으면 true (느슨한 비교)
- `===` → 값 + 타입 모두 같아야 true (엄격한 비교)

## ◆ 현재 변수 상태

```
var com1 = 10;    // number
var com2 = 20;    // number
var com3 = "30";  // string
var com4 = "40";  // string
var com5 = true;  // boolean
var com6 = false; // boolean
```

## ◆ 형변환 예시

### 1) 숫자 ↔ 문자열

```
String(com1); // "10"
Number(com3); // 30
```

### 2) 불리언 ↔ 숫자

```
Number(com5); // 1
Number(com6); // 0
Boolean(0);    // false
Boolean(123);  // true
```

### 3) 불리언 ↔ 문자열

```
String(com5); // "true"
String(com6); // "false"
Boolean("Hello"); // true
Boolean(""); // false
```

### ◆ 암시적 형변환 (자동 변환)

```
console.log(com1 + com3); // "1030" (숫자 + 문자열 → 문자열)
console.log(com1 + com2); // 30 (숫자 + 숫자 → 숫자)
console.log(com3 * 2); // 60 (문자열 "30" → 숫자 변환)
console.log(com4 - 10); // 30 (문자열 "40" → 숫자 변환)
```

#### ✓ 정리

- 명시적 변환 → `Number()`, `String()`, `Boolean()`
- 암시적 변환 → JS가 상황에 맞춰 자동 변환 (`+`, `,`, 비교 연산 등)

## 1. 변수 이해하기

```
//실습3. 데이터 형변환 : type casting
var com1 = 10; // string
var com2 = 20; // number
var com3 = "30";
var com4 = "40";
var com5 = true;
var com6 = false;

String(com3); // String() : 문자열로 형변환
console.log( "com3의 데이터타입은 " + typeof(com3) + "입니다.");
document.write("com1 과 com2의 합은 " + (com1 + com2) + "입니다.<br><br>" ); //30

// (중요) + 연산시 데이터형이 (하나라도 문자형)일경우 [무조건] 문자열로 결합이 된다.
var result7 = com2 + com4;
document.write("문자열 결합의 결과 =>" + result7 + "입니다.<br><br>"); //2040
```

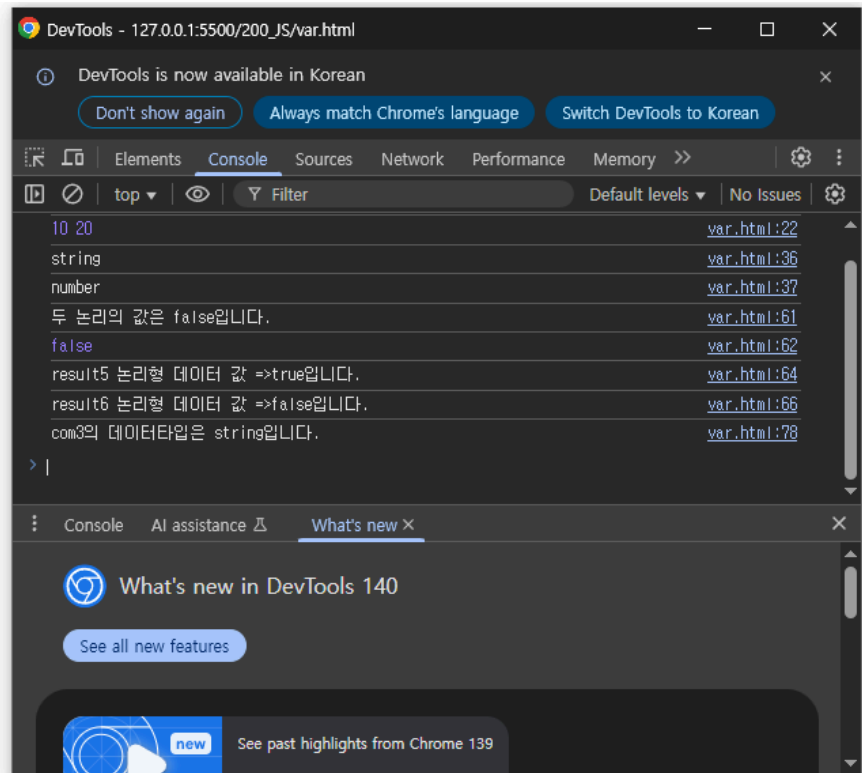
HelloJavascript + 자바스크립트

300두합의 결괏값은 : 300입니다!!

계산의 결괏값은 : 130입니다.

com1 과 com2의 합은30입니다.

문자열 결합의 결과 =>2040입니다.



## 2. 변수 재사용성



```

<!DOCTYPE html>
<html lang="ko">

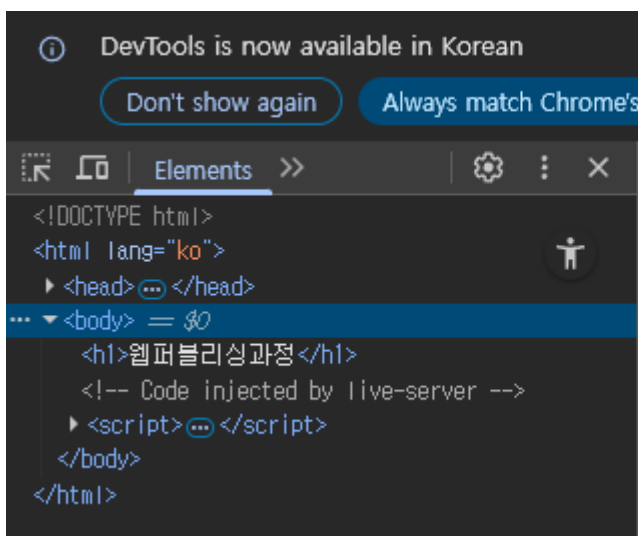
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>변수의 재사용성</title>
  <script>
    var aa = "<"
    var bb = "h"
    var cc = "1"
    var dd = ">"
    var ee = "/"
    var result = null;

    result = aa + bb + cc + dd + "웹퍼블리싱과정" + aa + ee + bb + cc + dd;
    document.write(result);

  </script>
</head>
<body>

</body>
</html>

```



웹퍼블리싱과정

### 3. 변수 실습

```

<!DOCTYPE html>
<html lang="ko">

<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <title>변수 실습</title>
  <script>
    var user = "해당 사용자의 키는 ";
    var tall = 174;
    var Height = "cm";
    var Weight = "kg";

    var totalweight = (tall - 100) * 0.9;
    var messageHeight = user + tall + Height + "입니다.";
    var messageWeight = "적정 몸무게는 " + totalweight + "입니다.";

    document.write("<h1>" + messageHeight + "</h1>");
    document.write("<br>" + messageWeight);
  </script>
</head>

<body>

</body>

</html>

```

**해당 사용자의 키는 174cm입니다.**

적정 몸무게는 66.60000000000001kg입니다.