



# JavaScript 101 - (1) 변수와 데이터 타입

## JavaScript란 무엇인가?

JavaScript는 웹 페이지에서 복잡한 기능을 구현할 수 있도록 하는 스크립팅 언어 또는 프로그래밍 언어입니다. JavaScript로 작성된 프로그램을 스크립트라고 하며, 컴파일이 필요하지 않습니다. 그냥 HTML 웹 페이지에 스크립트를 삽입하기만 하면 동작하며 최신 웹 브라우저에서 모두 동작합니다.

JavaScript는 웹을 풍성하고 동적으로 만들어주는 작고 가벼운 언어입니다. 예를 들어, JavaScript로 다음과 같은 일들을 할 수 있습니다.

- 웹 페이지의 내용이 주기적으로 갱신되거나 사용자와 상호작용이 가능하게 만들 수 있습니다.
- 애니메이션이 적용된 2D나 3D 그래픽을 그리거나 비디오 클립을 재생할 수 있습니다.
- 사용자의 위치 정보나 가속도계 데이터와 같은 장치의 센서 데이터를 읽어올 수 있습니다.
- 서버와 통신하여 데이터를 전송하거나 받아올 수 있습니다.

JavaScript는 오늘날 가장 널리 사용되는 프로그래밍 언어 중 하나이며, 웹 개발뿐만 아니라 다양한 분야에서 활용됩니다. 예를 들어, JavaScript로 서버 사이드 애플리케이션을 개발할 수 있는 Node.js와 같은 플랫폼이 있으며, JavaScript로 모바일 앱이나 데스크탑 앱도 만들 수 있습니다.

JavaScript를 배우면 웹 개발에 필수적인 기술을 습득할 수 있으며, 창의적이고 멋진 웹 사이트를 만들 수 있는 능력을 갖출 수 있습니다. 이제부터 JavaScript의 기본 문법과 특징에 대해 자세히 알아보겠습니다.

## JavaScript의 변수

JavaScript에서 데이터를 저장하는 방법은 여러 가지가 있습니다. 그 중에서 가장 일반적으로 사용되는 방법은 변수(variables)입니다. 변수란 데이터를 저장하는 저장소(공간)이라고 생각하면 됩니다. 변수를 선언하고 값을 할당할 수 있으며, 저장한 데이터를 변경하거나 추가할 수도 있습니다.

변수를 선언하기 위해서는 `var`, `let`, `const` 키워드 중 하나를 사용합니다. `var`는 구 방식으로 현재는 `var`보다 `let`과 `const`을 주로 사용합니다. 예를 들어, 다음과 같이 키워드로 변수나 상수를 선언하고 값을 할당할 수 있습니다.

```
var name = "Kim";
let age = 20;
const PI = 3.14;
```

위의 예시에서 `name`은 `var`로 선언된 변수이며, `age`는 `let`으로 선언된 변수입니다. `PI`는 `const`로 선언된 상수입니다. `name`과 `age`는 값이 변경될 수 있지만 `PI`는 값이 변경될 수 없습니다.

```
PI = 3.15; // TypeError: Assignment to constant variable.
```

위의 예시에서 `PI`는 `const`로 선언되었기 때문에 값을 변경하려고 하면 오류가 발생합니다. `const`는 `let`과 마찬가지로 블록 스코프(block scope)를 가지며, `var`와 달리 호이스팅(hoisting)이 일어나지 않습니다. 스코프와 호이스팅에 대한 내용은 다른 문서에서 설명하겠습니다.

## JavaScript의 데이터 타입

변수와 상수에 할당된 값들은 각각 문자열(string), 숫자(number), 불린(boolean), 널(null), 언디파인드(undefined)라는 원시형 데이터 타입과 객체(object), 배열(array), 함수(function)라는 참조형 데이터 타입을 가지고 있습니다.

### 원시형 데이터 타입

원시형 데이터 타입은 숫자(number), 문자열(string), 불리언(boolean), null, undefined 등이 있으며, 값 자체가 메모리에 저장됩니다. 기본 타입은 불변성을 가지며, 한 번 생성된 값은 변경할 수 없습니다. 예를 들어, 문자열을 변경하려고 하면 새로운 문자열이 생성되고 기존의 문자열은 그대로 남아있습니다.

- Number: 숫자를 의미합니다. 정수와 소수, 지수 모두 Number라는 데이터 타입으로 저장됩니다. `typeof`를 통해 확인하면 `"number"`로 표현됩니다.

- String: 문자열을 의미합니다. 따옴표(“ 또는 ”)로 묶여진 텍스트입니다. `typeof` 를 통해 확인하면 `"string"` 로 표현됩니다.
- Boolean: 참(true) 또는 거짓(false)을 의미합니다. 조건문이나 반복문 등에서 자주 사용됩니다. `typeof` 를 통해 확인하면 `"boolean"` 로 표현됩니다.
- Null: 값이 없음을 의미합니다. null이라고 적어야 합니다. `typeof` 를 통해 확인하면 `"object"` 로 표현됩니다. 비교문을 사용할 때에는 `(null === null)` 과 같이 비교합니다.
- Undefined: 값이 할당되지 않았음을 의미합니다. `undefined` 라고 적어야 합니다. `typeof` 를 통해 확인하면 `"undefined"` 로 표현됩니다.

## 참조형 데이터 타입

• 참조형 데이터 타입은 객체(object), 배열(array), 함수(function) 등이 있으며, 값이 저장된 메모리 주소가 변수에 저장됩니다. 참조 타입은 가변성을 가지며, 한 번 생성된 값도 변경할 수 있습니다. 예를 들어, 객체의 속성을 변경하려고 하면 같은 객체를 가리키는 변수들도 영향을 받습니다.

기본 타입과 참조 타입의 차이점을 이해하는 것은 JavaScript 프로그래밍에 중요합니다. 다음 예제들을 통해 확인해보세요.

```
//기본 타입
var x = 10; //x에 10이라는 숫자 값을 할당합니다.
var y = x; //y에 x값을 할당합니다.
x = 20; //x값을 20으로 변경합니다.
console.log(x); //20
console.log(y); //10 (y값은 변하지 않습니다.)

var s1 = 'hello'; //s1에 'hello'라는 문자열 값을 할당합니다.
var s2 = s1; //s2에 s1값을 할당합니다.
s1 = 'world'; //s1값을 'world'로 변경합니다.
console.log(s1); //'world'
console.log(s2); //'hello' (s2값은 변하지 않습니다.)

//참조 타입
var obj1 = {name: 'Alice', age: 20}; //obj1에 객체 리터럴로 객체 값을 할당합니다.
var obj2 = obj1; //obj2에 obj1값(메모리 주소)를 할당합니다.
obj1.name = 'Bob'; //obj1의 name 속성 값을 'Bob'으로 변경합니다.
console.log(obj1.name); //'Bob'
console.log(obj2.name); //'Bob' (obj2의 name 속성 값도 변했습니다.)

var arr1 = [1, 2, 3]; //arr1에 배열 리터럴로 배열 값을 할당합니다.
var arr2 = arr1; //arr2에 arr1값(메모리 주소)를 할당합니다.
arr1[0] = 4; //arr1의 첫 번째 요소 값을 4로 변경합니다.
console.log(arr1[0]); //4
console.log(arr2[0]); //4 (arr2의 첫 번째 요소 값도 변했습니다.)
```

데이터의 타입을 확인하기 위해서는 `typeof` 연산자를 사용할 수 있습니다. `typeof` 연산자는 피연산자의 타입을 나타내는 문자열을 리턴합니다. 예를 들면 다음과 같습니다.

```
typeof name; // "string"
typeof age; // "number"
typeof PI; // "number"
```

`typeof` 연산자는 다음 문서에서 추가로 확인할 수 있습니다.

JavaScript에서 변수와 데이터 타입은 웹 페이지에 다양한 정보와 기능을 표현하기 위해 필요한 요소입니다.