Javascript 기초

연산자와 조건문

김태민

저번에 우리는

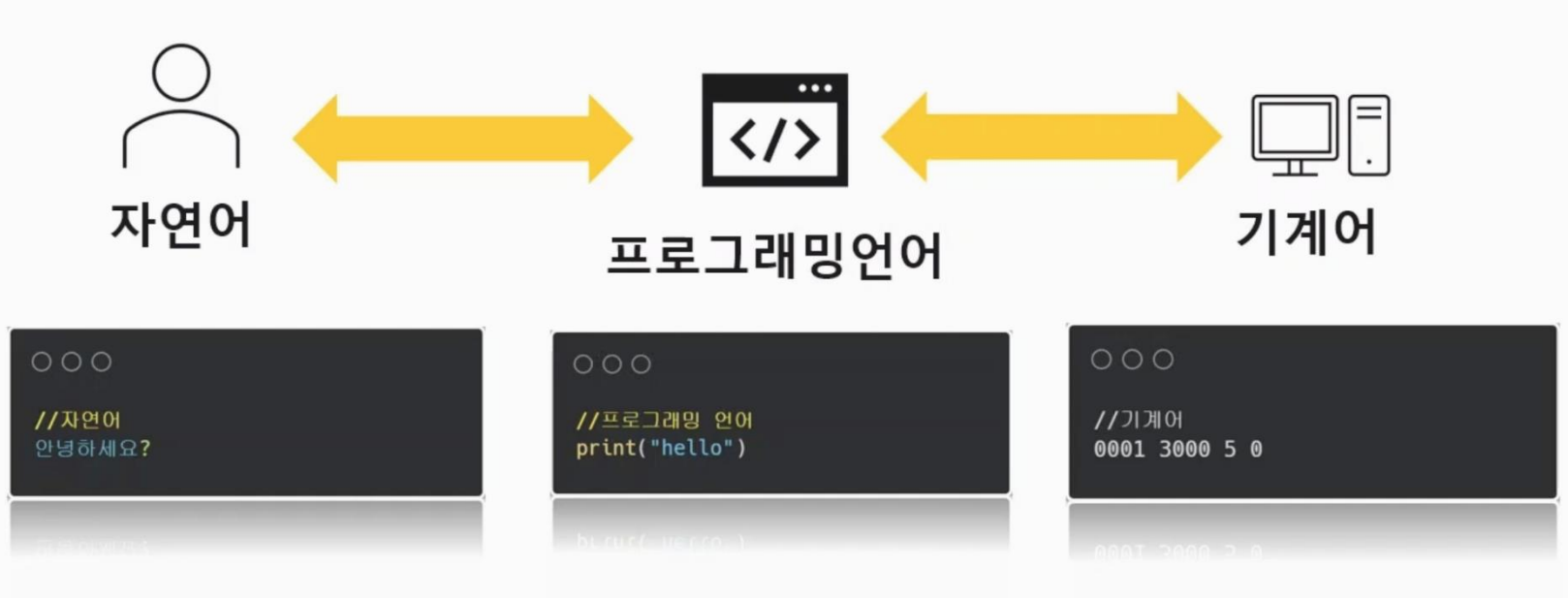
Javascript 개요

프로그래밍

- 컴퓨터가 특정 작업을
- 인간의 언어(한국어,
- 예: 웹사이트에서 버튼

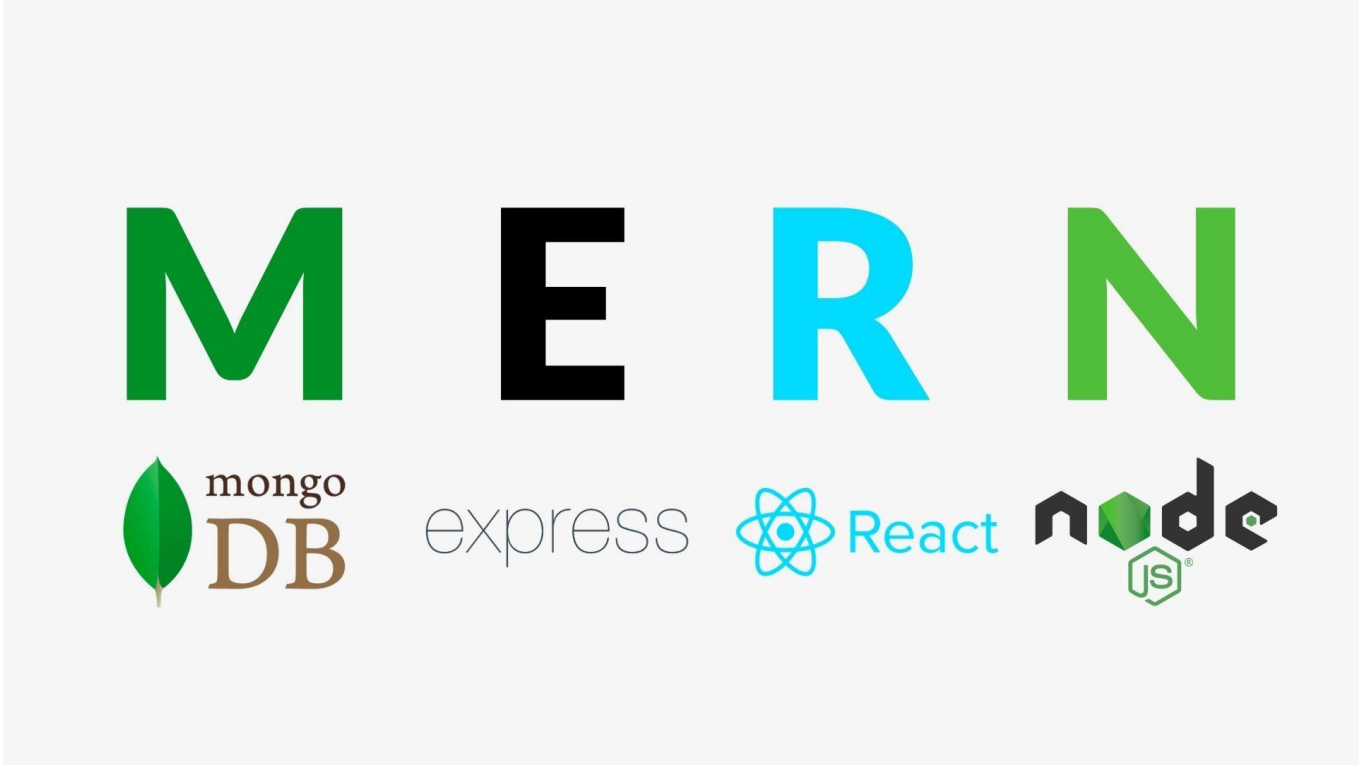
기계어를 번역하는 방식에 따라
컴파일드 언어(**Compiled Language**)와 인터프리티드 언어(**Interpreted Language**)

컴파일드 언어는 다수의 명령어로 이뤄진 소스코드를 한 번에 기계어로 번역해서 실행 파일을 생성. **[C, C++, C#, GO, Java]**
반면, 인터프리티드 언어는 소스코드를 한 줄씩 기계어로 번역해서 실행 결과를 보여줌. 이때문에 인터프리티드 언어는 스크립트(**Script**)언어라고도 한다.
[Javascript, Python, Ruby, PHP, shell script]



Javascript로 시작하는 프로그래밍

- 웹 개발의 핵심 언어, 브라우저에서 바로 실행
- 프론트엔드(React, Vue)와 백엔드(Node.js) 모두 가능
- 초보자 친화적: 간단한 코드로 즉각적인 결과 확인



Javascript 개요

Client	Server	DB
PC Browser Mobile Chrome App Mobile App	RESTful Api Oauth RAG / LLM Api	UserInfo ProductInfo Payment ...
NextJS	NodeJS Express	MongoDB / MySQL

AWS / GCP / Azure / Cafe24 ...

변수 선언, 할당, 초기화

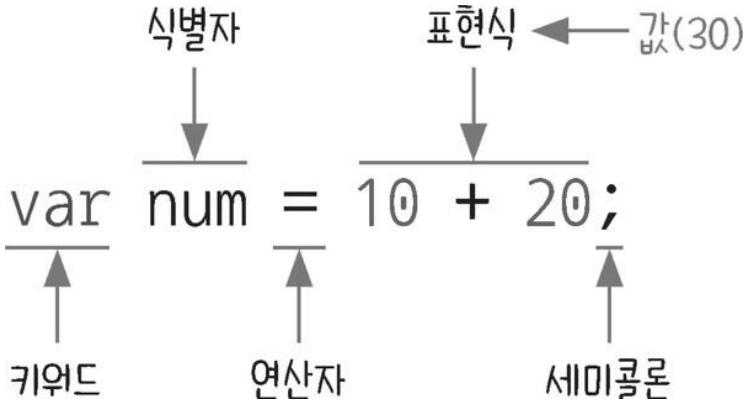
변수를 생성하고 값을 저장하는 문법에서 var 키워드나 이후에 배우는 let, const 키워드를 사용

'변수의 식별자를 지정하는 행위를 **변수를 선언한다**고 부름.

할당 연산자인 = 기호로 우변에 있는 값을 변수 공간에 대입(저장)하는 것을 **값을 할당한다**고 함.

변수는 초기에 값을 할당하지 않고 선언만 할 수 있음.

var num;



자료형

자료형(data type)이란 자바스크립트에서 사용할 수 있는 데이터의 종류를 의미.
자바스크립트의 자료형은 **기본 자료형**과 **참조 자료형**으로 구분.

기본(primitive) 자료형으로는 문자(string), 숫자(number), 논리(boolean), undefined, null, Symbol 자료형이 있고, 참조(reference) 자료형에는 객체(object)가 있음.

```
console.log(typeof 변수명);
```

실습 환경

실습환경

OS : Window / Mac

브라우저 : Chrome

에디터 : VS Code https://code.visualstudio.com/

VS Code 익스텐션 : ESLint, Prettier, HTML CSS Support, HTML to CSS autocompletion, Auto Rename Tag, Auto Close Tag, htmltagwrap

Git : git bash, github 가입

연산자

연산자

산술 연산자

덧셈, 뺄셈, 곱셈, 나눗셈과 같은 수학 연산을 수행하는 연산자

구분	연산자	예	설명
이항 산술	+	x + y	x에 y를 더합니다.
	-	x - y	x에서 y를 뺍니다.
	*	x * y	x에 y를 곱합니다.
	/	x / y	x를 y로 나눕니다.
	%	x % y	x를 y로 나누어 나머지를 구합니다.
	**	x ** y	x의 y 거듭제곱을 구합니다.
단항 산술	++	x++ (후치 연산) ++x (전치 연산)	x를 1 증가시킵니다.
	--	x-- (후치 연산) --x (전치 연산)	x를 1 감소시킵니다.
단항 부정	-	-x	x의 부호를 부정합니다(음수 → 양수, 양수 → 음수)

연산자

산술 연산자 – 이항 산술 연산자

피연산자가 2개 필요한 연산자

```
let sum = 10 + 20; // 30
let sub = 20 - 10; // 10
let multi = 10 * 20; // 200
let div = 10 / 2; // 5
let remain = 10 % 3; // 1
let expon = 2 ** 3; // 8(2의 3승)
```

산술 연산자 – 단항 산술 연산자

피연산자가 1개 필요한 연산자

```
let increment = 10;
increment++;
let decrement = 10;
decrement--;
console.log(increment); // 11
console.log(decrement); // 9

let increment = 10++; // 불가
```

산술 연산자 – 단항 산술 연산자 : 전치연산 / 후치연산

앞에 사용하면 **전치 연산**, 뒤에 사용하면 **후치 연산**

```
let num = 10;
let subNum = ++num; // 앞에 사용했으므로 전치 연산 방식
console.log(subNum); // 11
```

```
let num = 10;
let subNum = num++; // 뒤에 사용했으므로 후치 연산 방식
console.log(subNum); // 10
```

값을 먼저 **할당한 후에** 증감 연산을 합니다.
그래서 코드를 실행해 보면 subNum 변수에
num 변수 값인 10을 먼저 할당한 후에 1을
증가시키므로 subNum 변수를 출력하면 10이 나옴

연산자

산술 연산자 – 단항 부정 연산자

항상 피연산자 앞에 위치하며 피연산자의 부호를 부정하는 연산을 수행

음수는 양수로 변환하고 양수는 음수로 변환하는 연산

숫자형 데이터가 할당된 변수에만 사용

```
let num = -10;
num = -num;
console.log(num); // 10
```

연산자

대입 연산자

데이터를 대입(할당)하는 연산

복합대입 연산자

산술 연산자와 대입 연산자를 함께 사용해 산술과 할당을 한 번에 수행

연산자	예	설명
=	x = y	x에 y를 대입합니다.
+=	x += y	x에 x + y를 대입합니다.
-=	x -= y	x에 x - y를 대입합니다.
*=	x *= y	x에 x * y를 대입합니다.
/=	x /= y	x에 x / y를 대입합니다.
%=	x %= y	x에 x % y를 대입합니다.
**=	x **= y	x에 x ** y를 대입합니다.

대입 연산자 – 복합 대입 연산자

```
let x = 10;
x += 5; // 15(x = x + 5)
let y = 10;
y -= 5; // 5(y = y - 5)
let z = 10;
z *= 5; // 50(z = z * 5)
let u = 10;
u /= 5; // 2(u = u / 5)
let v = 10;
v %= 3; // 1(v = v % 3)
let t = 10;
t **= 2; // 100(t = t ** 2)
```

연산자

비교 연산자

피연산자를 비교한 뒤,
논리형 값인 참(true),
거짓(false)을 반환

```
10 == '10'; // true
10 === '10'; // false
10 != '10'; // false
10 !== '10'; // true
10 < 10; // false
10 <= 10; // true
10 > 10; // false
10 >= 10; // true
```

연산자	예	설명
==	x == y	x와 y의 값이 같으면 true를 반환합니다.
===	x === y	x와 y의 값과 자료형이 같으면 true를 반환합니다.
!=	x != y	x와 y의 값이 다르면 true를 반환합니다.
!==	x !== y	x와 y의 값과 자료형이 다르면 true를 반환합니다.
〈	x 〈 y	x가 y보다 작으면 true를 반환합니다.
〈=	x 〈= y	x가 y보다 작거나 같으면 true를 반환합니다.
〉	x 〉 y	x가 y보다 크면 true를 반환합니다.
〉=	x 〉= y	x가 y보다 크거나 같으면 true를 반환합니다.

논리 연산자

피연산자를 논리적으로 평가한 뒤, 조건에 맞는 피연산자를 반환

연산자	예	설명
&&	x && y	x가 참이면 y를 반환하고, 거짓이면 x를 반환합니다.
||	x || y	x가 참이면 x를 반환하고, 거짓이면 y를 반환합니다.
!	!x	x가 참이면 false를 반환하고, 거짓이면 true를 반환합니다.

논리 연산자 – And 연산자 (&&)

피연산자를 왼쪽부터 평가해 평가 결과가 거짓이면 거짓이 나온 피연산자를 즉시 반환

거짓이 아니면 마지막에 평가되는 피연산자를 반환

```
true && true; // true
true && false && true; // false
```

숫자형을 쓰거나 문자열을 쓰거나 전부 논리 값으로 평가

""(빈 문자열), undefined, 0, null만 거짓으로 평가되고 나머지는 참으로 평가

AND 연산자는 연산 결과가 거짓으로 평가되면 거짓으로 평가된 피연산자를 반환

```
"" && "cat"; // ""
undefined && "cat"; // undefined
0 && "cat"; // 0
null && "cat"; // null
```

```
"cat" && "dog"; // "dog"
"cat" && "dog" && "bird"; // "bird"
```

논리 연산자 – Or 연산자 (||)

피연산자를 왼쪽부터 평가해 참으로 평가된 피연산자를 즉시 반환

모든 피연산자가 참으로 평가되지 않으면 마지막에 평가된 피연산자를 반환

```
false || true || false; // true
false || false; // false
false || "cat"; // "cat"
"" || "cat"; // "cat"
"dog" || "cat"; // "dog"
```

논리 연산자 – Not 연산자 (!)

피연산자나 식을 평가한 논리 값의 반대 값(true → false, false → true)을 반환

괄호로 식을 어떻게 묶는지에 따라 결과가 다름

!true && false와 !(true && false)는 결과가 다릅

```
!false; // true
!(10 < 20); // false
!(10 < 20 && 20 < 10); // true
```

삼항 연산자

세 항 중 가장 왼쪽에 있는 피연산자의 참, 거짓에 따라
나머지 두 항에 있는 피연산자를 선택적으로 반환하는 연산

연산자	예	설명
?:	x ? y : z	x가 참이면 y를 반환하고, x가 거짓이면 z를 반환합니다.

```
let score = 90;
let grade = score >= 90 ? 'A+' : 'B';
console.log(grade); // A+
```

연산자

연산 우선순위

그룹연산자() > 산술-곱하기,나누기 * / > 산술-더하기,빼기 + -

```
let sum = 10 + 20 * 3;
console.log(sum); // 70
let sum = (10 + 20) * 3;
console.log(sum); // 90
```

조건문

If 문

```
If (조건식) {
        // 조건식이 참이면 블록문 실행
}
```

```
let num = 10;
If (num % 2 === 0){
        console.log("변수 num에 할당된 숫자는 짝수입니다.");
}
```

else 문

```
if(조건식){
        // 조건식이 참이면 블록문 실행
}else{
        // 조건식이 거짓이면 블록문 실행
}
```

```
let num = 5;
if(num % 2 === 0){
        console.log("변수 num에 할당된 숫자는 짝수입니다.");
}else{
        console.log("변수 num에 할당된 숫자는 홀수입니다.");
}
```

else if 문

```
if(조건식1){
        // 조건식1이 참이면 블록문 실행
}else if(조건식2){
        // 조건식2가 참이면 블록문 실행
}else{
        // 조건식이 모두 거짓이면 블록문 실행
}
```

```
let num = 0;
if(num > 0){
        console.log("양수");
}else if(num < 0){
        console.log("음수");
}else{
        console.log("0");
}
```

조건문

중첩 if 문

```
if(조건식){
        if(조건식){
                // if 안쪽 if문
        } else {
                // if 안쪽 else문
        }
}else{
        if(조건식){
                // else 안쪽 if문
        } else {
                // else 안쪽 else문
        }
}
```

조건문

switch 문

```
switch(key){
        case value1:
                // key가 value1일 때 실행할 블록문
                break;
        case value2:
                // key가 value2일 때 실행할 블록문
                break;
        default:
                // 아무것도 일치하지 않을 때 실행할 블록
문
                break;
}
```

```
let food = "melon";
switch(food){
        case "melon":
                console.log("fruit");
                break;
        case "apple":
                console.log("fruit");
                break;
        case "carrot":
                console.log("vegetable");
                break;
        default:
                console.log("It's not fruits and vegetables.");
                break;
}
```

조건문

switch 문

switch 문은 소괄호 안의 값과
일치하는 case 문을 실행

해당 case 문의 블록문과 break 문 없음

break 문을 만날 때까지 case 문을 연속 실행

break 문을 만나 조건문이 종료

```
let food = "melon";
switch(food){
        case "melon":
        case "apple":
                console.log("fruit");
                break;
        case "carrot":
                console.log("vegetable");
                break;
        default:
                console.log("It's not fruits and vegetables.");
                break;
}
```

조건문

If문과 조건식

앞에서 배운 논리 연산자나 비교 연산자를 식에 이용
점수가 90점 이상이면 A++ 학점이라고 출력하는 조건문

```
let score = 90;
if(score >= 90){
        console.log("A++ 학점");
}
```

100점 이하라는 조건을 추가
AND 연산자는 피연산자를 평가해 모두 참이면
마지막에 평가되는 피연산자를 반환

```
let score = 90;
if(score >= 90 && score <= 100){
        console.log("A++ 학점");
}
```

조건문

If문 vs switch문

if 문은 조건에 식(statement)을 사용하고,

switch 문은 조건에 값(value)을 사용

90부터 99까지 'A++ 학점'이라고 출력하는 코드

```
let score = 90;
if(score >= 90 && score < 100){
        console.log("A++ 학점");
}
```

```
let score = 90;
switch(score){
        case 90:
        (중략)
        case 98:
        case 99:
                console.log("A++ 학점");
                break;
        default:
                break;
}
```

오늘 우리는

연산자

산술 연산자

덧셈, 뺄셈, 곱셈, 나눗셈과 같은 수학 연산을 수행하는 연산자

구분	연산자	예	설명
이항 산술	+	x + y	x에 y를 더합니다.
	-	x - y	x에서 y를 뺍니다.
	*	x * y	x에 y를 곱합니다.
	/	x / y	x를 y로 나눕니다.
	%	x % y	x를 y로 나누어 나머지를 구합니다.
	**	x ** y	x의 y 거듭제곱을 구합니다.
단항 산술	++	x++ (후치 연산) ++x (전치 연산)	x를 1 증가시킵니다.
	--	x-- (후치 연산) --x (전치 연산)	x를 1 감소시킵니다.
단항 부정	-	-x	x의 부호를 부정합니다(음수 → 양수, 양수 → 음수)

연산자

대입 연산자

데이터를 대입(할당)하는 연산

복합대입 연산자

산술 연산자와 대입 연산자를 함께 사용해 산술과 할당을 한 번에 수행

연산자	예	설명
=	x = y	x에 y를 대입합니다.
+=	x += y	x에 x + y를 대입합니다.
-=	x -= y	x에 x - y를 대입합니다.
*=	x *= y	x에 x * y를 대입합니다.
/=	x /= y	x에 x / y를 대입합니다.
%=	x %= y	x에 x % y를 대입합니다.
**=	x **= y	x에 x ** y를 대입합니다.

연산자

비교 연산자

피연산자를 비교한 뒤,
논리형 값인 참(true),
거짓(false)을 반환

```
10 == '10'; // true
10 === '10'; // false
10 != '10'; // false
10 !== '10'; // true
10 < 10; // false
10 <= 10; // true
10 > 10; // false
10 >= 10; // true
```

연산자	예	설명
==	x == y	x와 y의 값이 같으면 true를 반환합니다.
===	x === y	x와 y의 값과 자료형이 같으면 true를 반환합니다.
!=	x != y	x와 y의 값이 다르면 true를 반환합니다.
!==	x !== y	x와 y의 값과 자료형이 다르면 true를 반환합니다.
〈	x 〈 y	x가 y보다 작으면 true를 반환합니다.
〈=	x 〈= y	x가 y보다 작거나 같으면 true를 반환합니다.
〉	x 〉 y	x가 y보다 크면 true를 반환합니다.
〉=	x 〉= y	x가 y보다 크거나 같으면 true를 반환합니다.

논리 연산자

피연산자를 논리적으로 평가한 뒤, 조건에 맞는 피연산자를 반환

연산자	예	설명
&&	x && y	x가 참이면 y를 반환하고, 거짓이면 x를 반환합니다.
||	x || y	x가 참이면 x를 반환하고, 거짓이면 y를 반환합니다.
!	!x	x가 참이면 false를 반환하고, 거짓이면 true를 반환합니다.

삼항 연산자

세 항 중 가장 왼쪽에 있는 피연산자의 참, 거짓에 따라
나머지 두 항에 있는 피연산자를 선택적으로 반환하는 연산

연산자	예	설명
?:	x ? y : z	x가 참이면 y를 반환하고, x가 거짓이면 z를 반환합니다.

```
let score = 90;
let grade = score >= 90 ? 'A+' : 'B';
console.log(grade); // A+
```

조건문

If문 vs switch문

if 문은 조건에 식(statement)을 사용하고,

switch 문은 조건에 값(value)을 사용

90부터 99까지 'A++ 학점'이라고 출력하는 코드

```
let score = 90;
if(score >= 90 && score < 100){
        console.log("A++ 학점");
}
```

```
let score = 90;
switch(score){
        case 90:
        (중략)
        case 98:
        case 99:
                console.log("A++ 학점");
                break;
        default:
                break;
}
```

감사합니다