03

다양한 연산자

학습내용

01 연산자란

02 연산자의 종류

03 연산자의 사용 예

학습목표

- JavaScript에서 사용되는 다양한 연산자의 종류를 분류하고, 각 연산자의 의미와 특징을 설명할 수 있다.
- 각 연산자를 실제 코드에 적용하여 연산 결과를 예측하거나 설명할 수 있다.
- 연산자의 우선순위와 결합 규칙을 이해하고, 복합 표현식에서의 실행 순서를 설명할 수 있다.

01

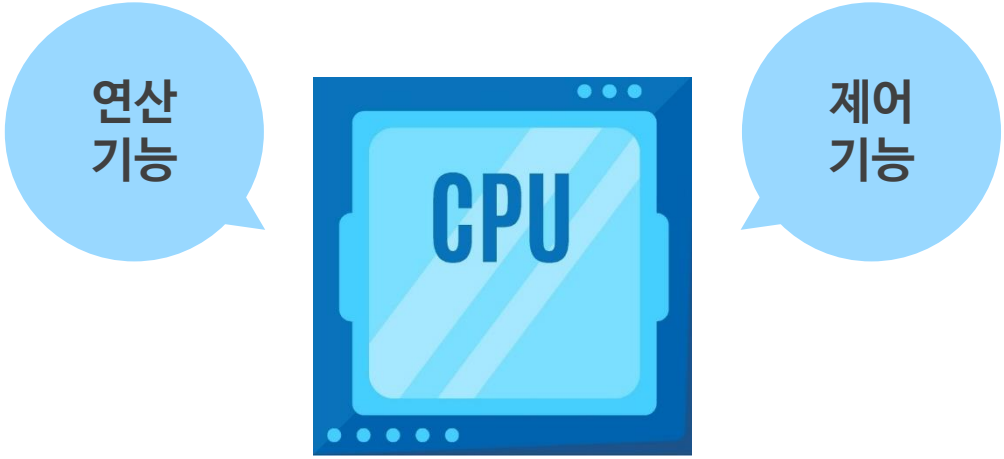
연산자란

1) 연산자의 일반적 정의

“ 연산자 ”

- CPU가 실행할 수 있는 명령어 집합(Instructions) 중 하나
- 피연산자에 대해 산술, 논리, 비교, 이동 등의 연산을 수행하는 기능 단위

2) CPU의 내부 구성과 역할에 기반한 연산자의 정의



이 기능 중 연산기능을 수행하도록 하는 연산자는
CPU의 ALU가 직접 처리할 수 있는 연산 명령어 집합에 해당함

02

연산자의 종류

1) JavaScript 연산자 종류 및 ECMAScript 버전별 지원

구분	연산자 종류	예시	설명	도입 버전
산술 연산자	+, −, *, /, %, ++, −−	a + b, a++	수학 연산을 수행	ES1
	** (거듭제곱)	2 ** 3	거듭제곱	ES2016(ES7)
대입 연산자	=, +=, −=, *=, /=, %=	x += 3	변수에 값을 할당	ES1
	**=	x **= 2	거듭제곱 후 대입	ES2016(ES7)
비교 연산자	==, !=, 〉, 〈, 〉=, 〈=	x == 5	느슨한 비교	ES1
	===, !==	x === 5	엄격한 비교	ES3
논리 연산자	&&, ||, !	console.log(!'hello'); // false		a && b
비트 연산자	&, |(OR), ^(XOR), ~(NOT), 〈〈, 〉〉, 〉〉〉	5^3→6(두 비트가 다름)	0101^0011→0110 XOR은 같으면 0, 다르면 1	비트 단위 연산
삼항 연산자	조건 ? 참 : 거짓	x〉10 ? '크다' : '작다'	조건문 간소화	ES1
타입 연산자	Typeof	typeof 3	데이터 타입 반환	ES1
	Instanceof	obj instanceof Class	인스턴스 확인	ES3
null 병합 연산자	??	a ?? b	a가 null 또는 undefined면 b 반환	ES2020(ES11)
전개 연산자(Spread)	...	[...arr]	배열/객체 전개	ES2015(ES6)
구조 분해 할당	{} 또는 []	const {a} = obj	값 추출	ES2015(ES6)
쉼표 연산자	,	(a = 1, b = 2)	여러 표현식 중 마지막 반환	ES1

03

연산자의 사용 예

1) 연산자 사용 예시

연산자 종류	예시
산술 연산자	let sum = 3 + 2;, let power = 2 ** 3;
대입 연산자	x *= 2;, y **= 3;
비교 연산자	5 === '5'; // false, 7 >= 4
논리 연산자	true && false, !isReady
비트 연산자	5 & 3, ~2, 4 << 1
삼항 연산자	let msg = age > 18 ? '성인' : '미성년자';
타입 연산자	typeof 'hello' // 'string', arr instanceof Array // true
null 병합 연산자	let name = inputName ?? '기본값';
전개 연산자	let copy = [...originalArray];, let newObj = {...oldObj}
구조 분해 할당	let [a, b] = [1, 2];, const {x, y} = point;
쉼표 연산자	let result = (x = 1, y = 2, x + y); // 3

일시정지 버튼을 누른 후, 아래의 학습활동에 참여하세요.

Q

변수 & 연산자 실습

웹 브라우저에서 다음 코드를 실행하여 결과를 확인해 보세요.

```
let a = 5, b = 3;
console.log('산술:', a + b);                    // 산술 : 8 → 산술연산자
a *= 2;                                         // 대입 연산자: a = a * 2
console.log('대입 후:', a);                     // 10
console.log('비교:', a > b);                    // 비교 연산자: true
console.log('논리 AND:', a > 0 && b > 0);       // 논리 연산자: true
console.log('논리 NOT:', !false);               // true
let max = a > b ? a : b;                        // 삼항 연산자
console.log('삼항 결과:', max);                 // 10
let name = null;
console.log('null 병합:', name ?? '기본값');    // '기본값'
```

학습 평가

Q1

다음 중 JavaScript의 기본형(Primitive Type)이 아닌 것은 무엇인가?

1. Number
2. String
3. Object
4. Boolean

학습 평가

Q1

다음 중 JavaScript의 기본형(Primitive Type)이 아닌 것은 무엇인가?

1. Number
2. String
3. Object
4. Boolean

정답 3

해설 Object는 참조형(Reference Type)입니다.
나머지(Number, String, Boolean)는 기본형입니다.

학습 평가

Q2

다음 중 const 키워드에 대한 설명으로 옳은 것은 무엇인가?

1. 재선언과 재할당 모두 가능하다.
2. 블록 스코프를 따르지 않는다.
3. 반드시 초기화가 필요하다.
4. 값이 변경되어도 에러가 발생하지 않는다.

학습 평가

Q2

다음 중 const 키워드에 대한 설명으로 옳은 것은 무엇인가?

1. 재선언과 재할당 모두 가능하다.
2. 블록 스코프를 따르지 않는다.
3. 반드시 초기화가 필요하다.
4. 값이 변경되어도 에러가 발생하지 않는다.

정답 3

해설 const는 선언 시 반드시 초기화를 해야 하며,
이후 재할당이 불가능합니다. 블록 스코프도 따릅니다.

학습 평가

Q3 다음 코드의 실행 결과는 무엇인가?

```
console.log("5" - true);
```

학습 평가

Q3 다음 코드의 실행 결과는 무엇인가?

```
console.log("5" - true);
```

4

정답 4

해설 "5"는 숫자 5로, true는 숫자 1로 toNumber 변환된 후,
5 - 1 = 4를 수행해서 4가 정답입니다.

학습 평가

Q4 다음 표현식은 true를 출력한다.

```
[ ] == 0
```

O X

학습 평가

Q4 다음 표현식은 true를 출력한다.

```
[ ] == 0
```

X

정답 O

해설 [] → “” (toPrimitive) → 0 (toNumber)
0 == 0 → true

학습 평가

Q5 다음 코드의 출력 결과는 어떻게 되는가?

```
var y;
console.log(y);
```

학습 평가

Q5 다음 코드의 출력 결과는 어떻게 되는가?

```
var  y;
console.log(y);
```

Undefined

정답 Undefined

해설 변수 y는 선언되었지만 값이 없으므로
typeof y는 “undefined”를 반환합니다.

학습 평가

Q6 다음 코드의 출력 결과는 어떻게 되는가?

```
let a = null;
let b = "대체값";
console.log(a ?? b);
```

학습 평가

Q6 다음 코드의 출력 결과는 어떻게 되는가?

```
let a = null;
let b = "대체값";
console.log(a ?? b);
```

대체값

정답 대체값

해설 ?? 연산자는 a가 null 또는 undefined일 때만 오른쪽 값을 반환합니다. 따라서 "대체값"이 출력됩니다.

학습정리

변수와 연산자

1교시

프로그래밍 실행 원리 & 메모리

- 프로그램의 실행 원리
- 메모리 구조
- 변수란?
- 변수 선언법
- var/let/cons

2교시

자료형

- 자료형의 필요성
- 자료형의 종류
- 자료형 변환

3교시

연산자

- 종류
  - 산술, 대입, 비교, 논리, 비트, ??, 삼항연산자 등
- 사용법
  - 우선순위, 결합규칙

세종사이버대학교

03주. 흐름을 제어한다! 제어문

01

조건문