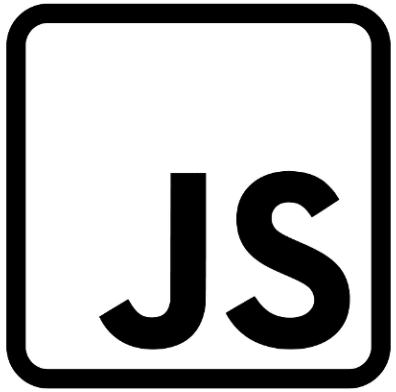


JavaScript Basic

Why JavaScript?

- 앞으로 학습할 **React**는 JavaScript 프레임워크

- Python, Java에 이은 전세계 인기 프로그래밍 언어.. **JavaScript!**

- 지금까지 학습한 Html, Css는 정적인 페이지를 만들 수 있고, 이 Html, Css에 JavaScript를 더해 **동적**인 페이지를 제작할 수 있다.

변수 자료형 연산자 조건문 반복문 함수 배열

변수(let)

let

중복선언 X

값 재할당 O

const

중복선언 X

값 재할당 X

변수(선언과 할당)

프로그래밍에서 '='은 "같다"의 의미가 아닌 값을 "삽입"한다는 의미이다.

let name; (선언)

name = "재오"; (할당)

name = "시현"; (재할당)

const name; (선언)

name = "재오"; (할당)

name = "시현"; (재할당X)

자료형

숫자 관련 자료형 int / float

문자 관련 자료형 string

참 거짓 자료형 True / False

값이 없다 null (어떤 값이 의도적으로 비어있음을 나타냄)

무슨 값인지 모를 때 undefined (값이 지정되지 않았음을 나타냄)

조건문(if)

```
if(조건문) {
    실행할 내용
}
```

```
if(조건문 a) {
    a조건이 True일 때 실행할 내용
}
else if(조건문 b) {
    a조건이 False이고 b조건이 True일 때
}
else{
    a, b 조건이 모두 False일 때
}
```

반복문(for)

```
for(초기문; 조건문; 증감문;){
    실행할 내용
}
```

1. 초기문을 실행한다.
2. 조건문을 확인한다.
3. 조건문이 True일 경우 실행할 내용을 시행한다.
4. 증감문을 실행한다.

ex)

```
for(let i=0; i<5; i++;){
    document.write("동령");
    document.write(i);
}
```

결과값

동령 0

동령 1

동령 2

동령 3

동령 4

반복문(while)

```
while(조건문) {
    실행할 내용
}
```

1. 조건문을 확인한다.
2. 조건문이 True일 경우 실행할 내용을 시행한다.
3. 조건문이 False일 경우 넘어간다.

ex)

```
let i = 0;
while(i < 5){
    document.write("세현");
    document.write(i);
    i = i + 1;
}
```

결과값

세현 0
세현 1
세현 2
세현 3
세현 4

연산자(기본)

기본 연산자: +, -, *, /, % (/는 나눗셈 후 몫을 의미하고, %는 나눗셈 후 나머지를 의미한다.)

x += y 이 식은 x = x+y 랑 같은 식이다. 뺄셈, 나눗셈, 곱셈도 마찬가지이다.

x의 값에 y를 더하고 이를 다시 x에 삽입하는 의미이다.

```
alert(4 / 2) // 2출력
alert(5 % 2) // 1 출력

let n = 2;
n = n+2; // n = 4
n += 2; // n = 6
```

연산자(논리)

논리 연산자: && || !

&&는 and와 같은 의미이고 비교하는 두 값이 모두 True일 때 True 이다.

true && true는 true이고, true && false는 false, false && false는 false이다.

||는 or와 같은 의미이고 비교하는 두 값 중 한개라도 True일 때 True 이다.

true || true는 true이고, true || false는 true, false && false는 false이다.

!는 not의 의미이고 !true는 false 이다.

연산자(비교)

비교 연산자: === !== > >= < <=

===(일치)

두 피연산자의 값과 타입이 모두 같으면 true를 반환한다.

"상혁" === "지환" // false

!==(불일치)

두 피연산자의 값 또는 타입이 다른 경우 true를 반환한다.

삼항 연산자

bool형 또는 조건문 ? True일 때 실행할 부분 : False일 때 실행할 부분

```
let 학번 = 18;
let 동령 = (학번 < 19) ? "화석" : "응애";
document.write(동령) // 화석
```

함수

일들을 단순화 시키고 일률화 하기 위해 사용 + 재사용

함수 적용 전

```
document.write("가희")
document.write(1)
document.write("홍석")
document.write(2)
document.write("서아")
document.write(3)
document.write("유선")
document.write(4)
document.write("주용")
document.write(5)
```

함수 적용 후

```
function name_num(name, num){
    document.write(name);
    document.write(num);
}
name_num("가희", 1);
name_num("홍석", 2);
name_num("서아", 3);
name_num("유선", 4);
name_num("주용", 5);
```

함수(제작 방법)

```
function 함수명 (입력받을 매개변수, 매개변수 2 ...) {
    실행할 내용
    return 반환 할 내용
}
```

ex)

```
function addNum(x,y){
    return x+y;
}
document.write(addNum(3,4)); // 7
```

함수(익명 함수)

함수의 이름이 없다는 것을 의미한다.

```
function addNum(x,y){
    return x+y;
}
```

위 함수의 이름은 addNum이다.

```
let addNum = (x,y) => x+y
```

위 함수의 이름은 없다.
대신 addNum이라는 변수에 함수를 담았다.

함수(Array Function)

```
let func = function(arg1, arg2, arg3, ··· argN){
    return expression;
};
```

= 일반 함수

```
let func = (arg1, arg2, arg3, ··· argN) => expression
```

= 화살표 함수

위와 같이 화살표(=>)를 이용한 함수를 의미한다.
화살표 함수를 이용할 때에는 화살표를 기준으로 좌측에는 인자를, 우측에는 표현식을 평가하고 반환한다.
Array Function은 항상 익명함수를 사용한다.
expression 부분에 중괄호를 사용했다면 반드시 return 값을 명시해주어야 한다.

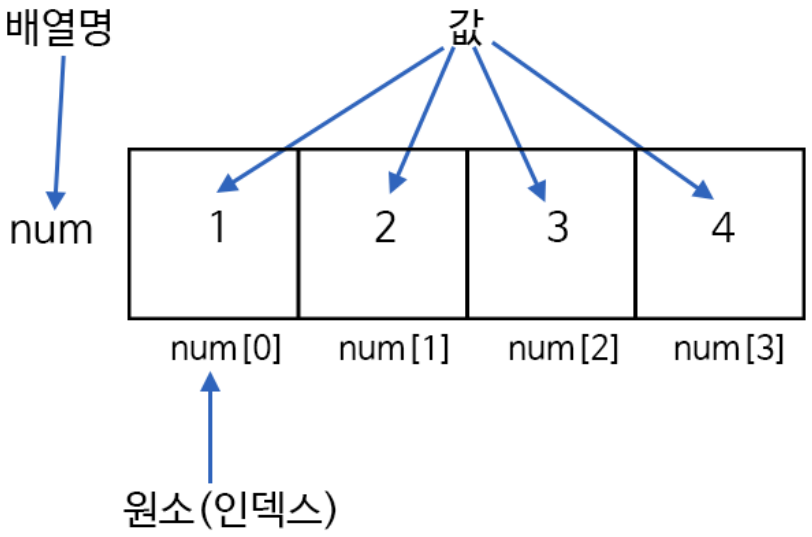
함수(Array Function + map메소드)

배열의 map() 메소드란?

배열 속 모든 요소를 인자로 받아서 map() 내부의 익명함수를 통해 만들어진 새로운 배열을 반환해준다.(반복문과 비슷)

```
let arr = [1,2,3,4,5];
let arr2 = arr.map((num) => {
    document.write(num);
    num * 2;
})
// document.write(num)에 의해 1,2,3,4,5 순차적으로 출력한다
document.write(arr2); // [2,4,6,8,10]
```

배열(Array)



let num = [1,2,3,4];

num[0] -> 1

num[1] -> 2

num[2] -> 3

num[3] -> 4

배열은 0번째 부터!

자바스크립트 배열은 서로 다른 자료형을 담을 수 있다. ex) [1998, "진아", "정보통신공학과"]

배열(Array)

배열의 생성: let array = [ ];

push: 배열 가장 끝에 값을 삽입

```
let frontEnd = ["은서","성"];
frontEnd.push("재오"); // ["은서", "성", "재오"]
```

pop: 배열 가장 끝에 값을 삭제하고 반환

```
let frontEnd = ["은서","성"];
frontEnd.pop(); // 성
alert(frontEnd); // 은서
```

https://ko.javascript.info/array