**JavaScript**

**● Js 실행문은 ; 로 구분한다. 대소문자를 지켜 써줘야 한다.**

**● 식별자: 영문자(대소문자), 숫자, 언더스코어(\_) 또는 달러($)**

**보통 첫단어 소문자+첫문자 대문자로 작성(CamelCase)**

**● 주석은 // 또는 /\* \*/ 사용**

**● Js를 html에 출력하고 싶다면**

1. window.alert() 메소드

2. HTML DOM 요소를 이용한 innerHTML 프로퍼티

3. document.write() 메소드

4. console.log() 메소드

**● Js를 html에 적용하려면**

1. <head>나 <body>에 직접 삽입하기

<head(또는 body)>

<script> function printDate() {document.getElementById("date").innerHTML = Date();}</script>

</head(또는 body)>

**JavaScript는 보통 <head> 또는 <body>의 맨 하단에 넣는다.**

<head>에 넣게 되면 데이터를 읽느라 시간이 오래 걸려 페이지 화면이 백색으로 뜨게되어 ‘느리다’는 인식을 줄 수 있다. Js를 처리하는 동안 html의 텍스트 요소들을 화면에 표시하기 위해 후자를 택하기도 한다.

2. 새로운 js 파일을 만든 후 <script> 안에 엮기.

<head>

<script src="JSC/jspractice.js"></script>

</head>

**● 타입은 숫자(number), 문자열(string)-”/’, 불리언(boolean)-t/f, 심볼(symbol), undefined, 객체(object)가 있다.**

**● 묵시적/명시적으로 타입을 변환할 수 있다.**

**3. 연산자**

●**산술 연산자**: 사칙연산, 피연산자 두 개, 왼쪽에서 오른쪽

우선순위: () → \*\* \* / % → + -

(같은 우선순위인 경우 결합 순서에 따라 연산)

●**대입 연산자**: 변수 값 대입, 피연산자 두 개, 오른쪽에서 왼쪽

복합 대입 연산자 존재함: +=, -=, \*=, /=, %=

Ex: x = 10; x += 5 -> x = x+5 = 15

+, -를 부호로 잘못 쓰지 않도록 주의 -> y-=5와 y=-5

\*const 상수 (X) let (O)

●**증감 연산자**: 피연산자 한 개

++ (1씩 증가) / --(1씩 감소)

연산자 위치에 따라 결과 달라짐 주의

Q. y++ - 3 = 7 : 10-3=7로 연산 끝, y=11로 초기화된 것?

●**비교 연산자**: 피연산자 두 개, 왼쪽에서 오른쪽

숫자끼리/문자열끼리 비교 (O) 숫자와 문자열 혼합 비교는 항상 false

동등/부등 연산자(==, !=)는 값만 비교(타입 강제변환)

일치/불일치 연산자(===, !==)는 타입까지 비교(별다른 변환X)

그 외 >, >=, <, <= 등 값 비교하는 연산자

●**논리 연산자**: &&와 ||는 피연산자 두 개, 왼쪽에서 오른쪽

! 는 피연산자 한 개, 단항 연산자, 오른쪽에서 왼쪽

&&(AND, 둘다 참), ||(OR둘중 하나라도 참이면 참), !(NOT, 참이면 거짓, 거짓이면 참)

●**비트 연산자**≒논리연산자: 비트 단위의 논리연산

&(둘다 1), |(둘중 하나 1), ^(XOR, 서로다름), ~(NOT, 반대)

<<(left shift), >>(right shift), >>> 등이 있음.

**●기타 연산자**

**문자열 결합 연산자**: +는 산술 연산자로 피연산자 2개

둘다 숫자면 덧셈, 둘다 문자열이면 결합,

둘 중 하나라도 문자열이면 결합(문자열로 강제변환)

**삼항 연산자**: 피연산자 세 개

* (반환값 1.2가 포함된 식)? (반환값1) : (반환값2)
* Ex: (3 > 5)? 3 : 5; => result=5

**쉼표 연산자**: for문에서 사용, 여러 변수 갱신

**delete 연산자**: 피연산자 한 개, 오른쪽에서 왼쪽

객체, 객체의 프로퍼티, 배열의 요소 등 삭제

삭제 성공 시 true, 실패 시 false

**typeof 연산자**: 피연산자 한 개, 오른쪽에서 왼쪽

**instance of 연산자**: 피연산자 두 개, 왼쪽에서 오른쪽

인스턴스인지 아닌지 확인(true/false)

**void 연산자**: 피연산자 한 개, 오른쪽에서 왼쪽

어떤 타입의 값이든 언제나 undefined

**4. 제어문**

**●조건문:** 표현식의 결과에 따라 명령 수행

1. if 문: 참일때 실행, 거짓이면 실행X  
2. if / else 문: 참일때 if 실행, 거짓일때 else 실행  
3. if / else if / else 문: 중첩된 조건문 표현, 삼항 연산자로 표현 가능.  
4. switch 문: case에 해당되는 값이 있으면 실행, 없으면 default

Break 키워드 반드시 포함.

**●반복문**: 동일한 명령을 일정 횟수 반복하여 수행

1. while 문: 조건 만족할 때까지 실행, 참이어도 다시 돌아와 판단

Loop: 검사를 통해 반복 실행되는 반복문을 칭함.

Infinite loop: 표현식의 결과를 변경하지 못해 루프를 영원히 반복.

=> 어느순간에는 false가 되도록 하는 실행문 필히 포함

2. do / while문: 무조건 루프 한 번 실행 후 표현식 검사

3. for 문: 초기식, 표현식, 증감식(각각 생략 가능, 쉼표로 도이 사용 가능)

=> while문보다 간결함.

4. for / in 문: 객체에 존재하는 프로퍼티 순회

루프마다 프로퍼티 이름을 변수에 대입하여 순차적 접근

=> 열거할 수 있는 프로퍼티 반환

5. for / of 문: 반복할 수 있는 객체( Array, Map, Set, arguments)를 순회

루프마다 프로퍼티의 값을 변수에 대입

●**기타 제어문**

1. label문: 특정영역 식별

2. continue문: 나머지 건너뛰고 다음 표현식으로 이동

continue; 또는 continue 라벨이름;

3. break문: 반복문 종료, 그 다음 실행문으로 이동

break; 또는 break 라벨이름;

**5. 배열**

**●배열(array 객체):** 배열 요소(값), 인덱스(배열 위치)

배열 요소끼리 타입 다를 수 있음.

배열 요소의 인덱스 연속되지 않을 수 있음

=> 홀을 참조하면 undefined 반환

**●배열의 생성**: [배열요소1, 배열요소2, …]

**●배열의 참조**: 배열이름[인덱스]

인덱스는 0부터 양의 정수만 사용

배열요소의 개수 = 배열의 길이(length 프로퍼티)

**●배열요소의 추가**: 배열의 최후에 새로운 요소 추가

**●배열의 순회**: 반복문을 사용하여 모든 배열요소에 차례대로 접근

**●배열의 활용**

희소 배열: 배열요소의 위치가 연속적 X

배열요소 개수 < 배열길이

다차원 배열: n차원 배열 = 요소가 (n-1)차원

연관 배열: 인덱스(0, 양의 정수) 대신 키(문자열)를 사용

array 객체가 아닌 기본 객체 즉 배열 X

문자열을 배열처럼 접근하기: split() 메소드 등을 이용해 먼저 배열로 변환

자바스크립트에서 배열 여부 확인: 1. Array.isArray() 메소드

2. instanceof 연산자

3. constructor 프로퍼티

**6. 함수**

●함수: 특별한 목적의 작업 수행하는 블록 { }

타입임. 변수 대입, 프로퍼티 지정 가능

값이기도 함. 변수에 대입될 수도 타 함수의 인수로 전달될 수도 있음.

함수 식별자: function name

●반환문: 함수 실행 중단, return 다음의 표현식 값(모든 타입O)을 반환

●변수의 유효 범위(variable scope): 변수가 접근할 수 있는 변수, 객체, 함수의 집합

지역 변수: 함수 내에서 선언

전역 변수: 함수 외부에서 선언

●함수의 유효 범위: 자신이 정의된 범위 안에서 정의된 모든 변수와 함수에 접근 가능

전역 함수: 모든 전역 변수와 전역 함수에 접근

내부 함수: 부모 함수에서 정의된 모든 변수 및 부모 함수가 접근할 수 있는 모든 다른 변수까지도 접근

함수 호이스팅: 유효 범위 적용이 변수 선언되기 전에도 적용

●매개변수와 인수

매개변수: 함수 호출 시 인수로 전달된 값을 함수 내부에서 사용가능토록 함.인수:  함수가 호출될 때 함수로 값을 전달해주는 변수나 상수

Argument 객체: 함수로 전달된 인수의 총 개수를 확인하거나, 각각의 인수에도 바로 접근 가능. 함수가 호출될 때 전달된 인수를 배열의 형태로 저장

●미리 정의된 전역 함수

1. eval(“문자열”)

2. isFinite(검사할값)

3. isNaN(검사할값)

4. parseFloat(“문자열”)

5. parseInt(“문자열”)

6. decodeURI(해독할URI)

7. decodeURIComponent(해독할URI)

8. encodeURI(부호화할URI)

9. encodeURIComponent(부호화할URI)

10. escape(변환할문자열")

11. unescape("원래대로변환할문자열")

12. Number(객체)

13. String(객체)