변수(타입)

Array(배열)

```
형태 : ['하나', '둘', '셋'],[100, 200, 300]

호출 : 변수명[0], 변수명[0][0] (변수명[index]) 이것을  indexing  되었다라고  말한다.

let  a = [[1,2,3,]

          [4,5,6,]

          [7,8,9,]]

a [0] = [1,2,3,]

a [0][1] = [2]
```

String(문자열)

```
형태 : 'abcde', "abcde", `abcde`  세가지  방법으로  사용  작은,큰따움표,벡틱

호출 : 변수명, 변수명[0]
```

Number(숫자)

```
형태 : 10, 12.123

호출 : 변수명
```

Boolean(논리값)

```
형태 : true, false

호출 : 변수명
```

Object(객체)

```
 //배열안에 오브젝트(객체)를 json 이라고 부름

   *형태 : {

   "지역이름": "전국",

   "확진자수": 24889,

   "격리해제수": 23030,

   "사망자수": 438,

   "십만명당발생율": 48.0

   }

*호출 : 변수명, 변수명.지역이름,변수명['지역이름'] (변수명.key , 변수명[key])
```

---

undefined

- undefined = undefined

null

- mull = object

NaN

- NaN = Number

산술연산

```
( +, -, /, *, **, % )
```

논리연산

```
( !, &&, ||) / 부정, 앤드(곱) 또는(합)
```

비교연산

```
( ==, !=, >, >=, <, <=, !==) / === 타입까지 비교(실무에서 많이씀),
```

조건문

```
( if, else  if, else, switch )

조건에  따라서  실행할지  말지를  결정
```

반복문

```
( for, for  in, for  of, while, do  while, forEach )
```

1번형태

```
for (let  i = 0; i < 10; i++) {
console.log(i);
}
```

2번 for of

```
let  a = [10, 20, 30, 40]

for (let  i  of  a) {
console.log(i)
}
```

3번 for in

```
let  a = [10, 20, 30, 40]

for (let  i  in  a) {
console.log(i)
}
```

4번 while

```
let  x = 0;

wihle (x < 10) {
console.log(x);
x++;
}
```

5번 do while

```
let  x =0;

do {
console.log(x);
x++;
} wihle (x < 10)
```

6번 for each

```
let  a = [10, 20, 30, 40];

a.forEach( e  =>  console.log(e**2));
```

7번 break

```
for (let  i = 0; i <10; i++) {
if (i == 5){
break;
}
console.log(i);
}
```

8번 continue *continue 는 해당 조건은 넘어가고 진행됨 건너뜀*

```
for (let  i = 0; i <10; i++) {
if (i == 5){
continue;
}
console.log(i);
}
```

함수

함수

```
형태

// 여기서 x, y 를 보통 한국에서는 인자라고 부른다.

// 매개변수(인자, 파라미터:para)(x, y)

// 전달인자 (아규먼트:argument) (3, 5)

function  add(x, y){
return(x+y);
}
add(3,5) //괄호안에 값이 전달이자(아규먼트)
```

콜백함수

함수를 인자값으로 들어오게 해서 사용한다.

```
function  add(x, y){
return(x+y);
}
add(3,5)

function  mul(x, y) {
return  x*y;
}

function  cal(a, b){
return  a(10, 10) + b(10, 10)
}
cal(add, mul);
```

화살표함수

일반 함수

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

화살표함수

```
let  addArrow = (x, y) =>  x + y;
매개변수  =>  리턴값
```

익명함수

```
//기명
var  aa = function  sum (x, y){
return  x + y
}

//익명 아님
var  bb = function (x, y){
return  x + y
}

// ES5 에서는 빈 문자열이었는데 ES6 에서 name 값을 가지는 것으로 바뀜.

//익명 같지만 바뀜
let  cc = (x, y) =>  x + y;
```

선언

```
let  c

c 를 선언한다.= undefined
let c = 10;
c 에 10 을 할당해  준다.
console.log(c); = 10

var (전역에서 사용 실무X)

let (변수로 사용한다.)

const (변하지 않는 상수값)
```

전개표현식 사용

```
function  add(...x) {
return(x);
}
add(1,2,3,4,5)
```

블록스코프

```
if (true) {
let  x =100; //지역변수는 해당 스코프 에서 사용한다.
}
console.log(x); //지역변수는 블록에서 나오면 사라져서 출력되지 않는다. var 제외
```

https://velog.io/@fromzoo/%ED%95%A8%EC%88%98%EC%8A%A4%EC%BD%94%ED%94%84-vs-%EB%B8%94%EB%A1%9D%EC%8A%A4%EC%BD%94%ED%94%84

구문

하나의 문장이라고 생각하면 된다.

---

console.log

window.console 자바스크립트의 내장 함수.

변수명으로써 변경해서 사용 가능하다.

리터럴

참고 사이트.

감성블로그

공식문서

프로그래머스

```
리터럴: 사람들이  이해할수있는  문자 (아라비아  숫자, 알파벳, 한글  등)
또는  미리  약속된  기호(“”, [], {},//)로 표기한 코드다

자바스크립트  엔진은  코드가  실행되는  시점인  런타임에  리터럴을  평가해  값을
생성한다.

즉, 리터럴은  값을  생성하기  위해  미리  약속한  표기법이라고  할  수  있다.
리터럴을  사용하면  다음과  같이  다양한  종류의  값을  생성할  수  있다.

Ex)
문자열  리터럴 “hello”
객체  리터럴 {name: “Lee”, address: “seoul”}
배열  리터럴 [1,2,3]
```

DOM

문서를 조작할때 DOM 을 활용한다. DOM 은 문서를 찍어낼수 있는 프린트 공장이다.

```
문서 객체 모델(The Document Object Model, 이하 DOM) 은 HTML, XML 문서의 프로그래밍
interface 이다. DOM은 문서의 구조화된 표현(structured representation)을 제공하며
프로그래밍 언어가 DOM 구조에 접근할 수 있는 방법을 제공하여 그들이 문서 구조, 스타일, 내용 등을
변경할 수 있게 돕는다.
```

DOM참고영상 DOM공식문서

BOM

//정규표현식

[0-9]{3}[-.*][0-9]{4}[-.*][0-9]{4}

0-9숫자 3자리

[-.*]

하이픈 닷 별 띄어쓰기

[0-9]{4}

0-9숫자 4자리

(|)|(([가-R]|(\d{1,5}(~|-)\d{1,5})|\d{1,5})+(로|길))