OPEN YEAR ROUND

Java script

2 조

CONTENTS

01 변수

- 변수
- 상수
- 데이터 타입

02 연산자

- 산술 연산자
- 할당 연산자
- 비교 연산자
- 논리 연산자

03 함수

- 익명 함수
- 화살표 함수
- 즉시실행 함수
- 콜백 함수
- 과제

01

변수

- 변수란?
- :바뀔 수 있는 값 =>한번 선언 후 변경 O
- 변수선언
- :let =>한번 선언했으면 같은 이름 선언 X

```
1  let value = 1;
2  console.log(value);
3  value = 2;
4  console.log(value);
```

```
1  let value = 1;
2  let value = 2;
```

01

변수의 이름

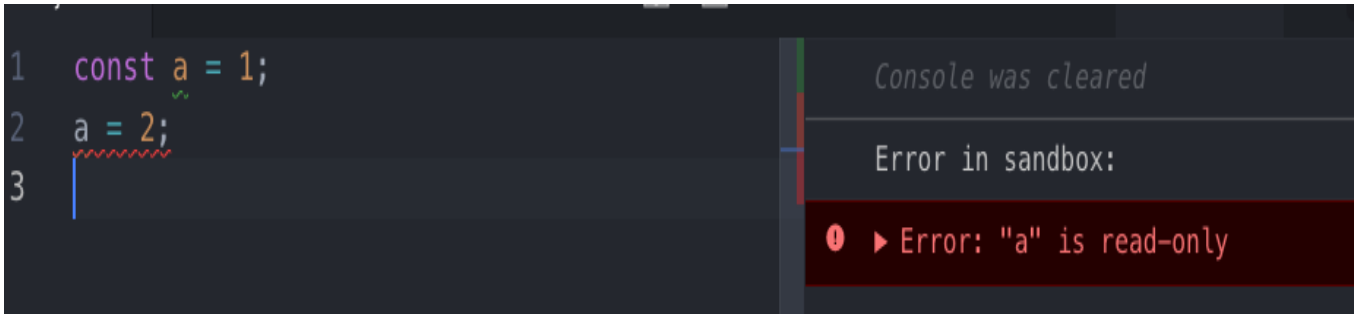
- -알파벳(A~Z/a~z), 숫자(0~9), 언더바(_), 달러($)기호 사용 가능
- -첫 글자로는 숫자 사용 X
- -예약어를 식별자로 사용 X
- ex) break/ case/ continue...

01

상수

- 상수란?
- :값이 바뀌지 않는 값(고정적인 값)
- 상수 선언
- :const =>한번 선언했으면 같은 이름 선언 X

#var
:모던 자바스크립트에서는
더 이상 사용 X



01

데이터 타입

숫자

:바로 값에 대입

```
let value = 1;
```

문자열

:작은 따옴표, 큰 따옴표로 감싸서 선언

=>큰 차이 X

```
let text = "Hello"
```

참/거짓(Boolean)

:참/거짓 2개 값만 존재

```
let a = true;
let b = false;
```

01

데이터 타입

null/ undefined

-null

:값이 없다

```
const abc = null;
```

-undefined

:아직 값이 설정되지 않았다

```
let a;
console.log(a);
```

02

연산자

연산자란?

:산술, 할당, 비교, 논리 등의 연산을 수행해서 값을 만든다

피연산자란?

:연산이 되는 대상자

```
x + y
// x, y가 피연산자
// + 가 연산자
```

02

산술연산자

- **이항 산술 연산자**

2개의 피연산자를 대상으로 연산
*부수효과가 없음: 피연산자의 값이 변경되지 않음

+ 덧셈

- 뺄셈

***** 곱셈

/ 나눗셈

% 나머지

- **단항 산술 연산자**

1개의 피연산자를 대상으로 연산
부수효과 있음

++ 1씩 증가 **--** 1씩 감소

```
let x = 3, result;

result = ++x;
// result: 4
// x: 4

result = x++;
// result: 4
// x: 5
```

02

할당 연산자&문자열 연결 연산자

- **할당 연산자**

우항에 있는 결과 값을 좌항의 변수에 대입

= 대입
+= x += y x = x + y
-= x -= y x = x - y
***=** x *= y x = x * y
/= x /= y x = x / y
%= x %= y x = x % y

- **문자열 연결 연산자**

+ 는 피연산자 중 하나 이상이 문자열인 경우 문자열 연산자

```
'1' + '2' // '12'

'1' + 2 // '12'
```

02

비교 연산자

- **동등, 일치 비교 연산자**

== 동등 비교

피연산자들의 값이 같으면 true

!= 반대 개념
피연산자들의 값이 다르면 true

=== 일치 비교
피연산들의 타입과 값이 같으면 true
!== 반대 개념
피연산자들의 타입과 값이 다르면 true

- **대소 관계 비교 연산자**

피연산자들의 크기를 비교

> x > y x가 y보다 클 때 true

< x < y x가 y보다 작을 때 true

>= x >= y x가 y보다 크거나 같을 때 true

<= x <= y x가 y보다 작거나 같을 때 true

02

논리 연산자&다양한 연산자들

```
true || true   // true
true || false  // true
false || true  // true
false || false // false


true && true   // true
true && false  // false
false && true  // false
false && false // false


!true  // false
!false // true
```

논리 연산자

|| 논리합(OR)

&& 논리곱(AND)

! 부정(NOT)

typeof 연산자
피연산자의 데이터 타입을 문자열

그룹 연산자
그룹 내의 표현식을 최우선으로 평가

```
10 * 2 + 3 // 23

10 * (2 + 3) // 50
```

03

함수

- **함수란?**

특정 코드를 하나의 명령으로 실행할 수 있게 해주는 기능

함수를 사용하는 이유

재사용

다른 인자를 사용하여 동일한 코드를 여러 번 사용

가독성

불필요한 코드들을 줄여줌

```
function add(a,b){
  let result=a+b;
  return result;
  console.log('hi');  //실행되지않음
}

add(1,2);
console.log(result);  //Error
```

(a, b): 파라미터(매개변수)

return으로 함수의 결과값을 반환

*return시 함수 끝, 이후의 코드들은 실행되지 않음

함수안에서 정의된 변수는 외부에서 접근 불가능

03

익명함수

함수명을 사용하지 않고 변수에 함수의 코드를 저장

선언 이후에만 호출가능

```
let 변수명 =
function(매개변수){
        실행문;
};
변수명();
```

03

화살표 함수

function 대신 **=>** 를 사용

```
let add = (a, b) => {
  return a+b;
};
add(1, 2);
```

```
let add = (a, b) => a+b;
add(1, 2);
```

03

즉시실행 함수

함수를 정의하자마자 바로 호출해 실행

익명함수를 ()로 묶고 뒤에 인수를 작성

```
(function(a, b){
  console.log(a+b);
})(3, 4);
```

03

콜백 함수

매개변수로 전달되어 특정시점에 호출

어떤 이벤트가 발생했거나 특정 시점에 도달했을 때 시스템에서 호출하는 함수

```
function returnName(callback){
  callback("sejong");
  console.log("만나서 반갑습니다.");
}

returnName(function(name){  //익명함수를 인자로 전달
  console.log("안녕하세요 "+name);
});
```

```
안녕하세요 sejong
만나서 반갑습니다.
```

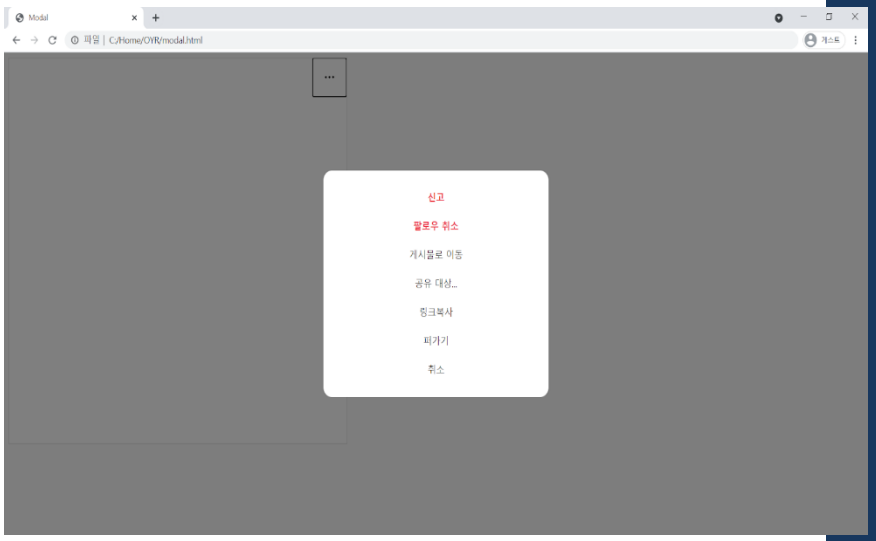
03

과제- 모달창 띄우기

```
let modal = document.getElementById("modalBox");
let open = document.getElementById("openBtn");
let close = document.getElementById("closeBtn");

open.onclick = function(){
  modal.style.display = "block";
}

close.onclick = function(){
  modal.style.display = "none";
}
```



THANK YOU