javascript 자료형

2020-01-19

이승진

학습목표

약타입 언어

javascript 기본 자료형

boolean

숫자형

Math 클래스

목차

[1. 약타입 언어(week typed language) 2](#_Toc36223435)

[1) 약타입 언어와 강타입 언어 2](#_Toc36223436)

[2) === 연산자, !== 연산자 3](#_Toc36223437)

[3) typeof 연산자 3](#_Toc36223438)

[4) undefined 4](#_Toc36223439)

[5) null 5](#_Toc36223440)

[2. javascript 자료형 6](#_Toc36223441)

[1) 기본 자료형 (primitive type) 6](#_Toc36223442)

[2) 복잡 자료형 (complex type) 6](#_Toc36223443)

[3. boolean 자료형 7](#_Toc36223444)

[1) true, false 값 7](#_Toc36223445)

[2) false 에 해당하는 값 7](#_Toc36223446)

[3) lazy evaluation 8](#_Toc36223447)

[4. number 자료형 9](#_Toc36223448)

[1) 정수 실수 구분이 없다 9](#_Toc36223449)

[2) 형변환 (string -> number) 9](#_Toc36223450)

[3) 형변환 (number -> string) 10](#_Toc36223451)

[5. Math 클래스 11](#_Toc36223452)

[1) PI - 원주율 11](#_Toc36223453)

[2) round - 반올림 메소드 11](#_Toc36223454)

[3) pow - 거듭제곱 메소드 11](#_Toc36223455)

[4) sqrt - 제곱근 메소드 12](#_Toc36223456)

[5) abs - 절대값 메소드 12](#_Toc36223457)

[6) floor/ceil - 내림/올림 메소드 12](#_Toc36223458)

[7) sin, cos, tan - 삼각함수 메소드 13](#_Toc36223459)

[8) max, min - 최대 최소 메소드 13](#_Toc36223460)

[9) random - 난수 메소드 13](#_Toc36223461)

[6. 요약 15](#_Toc36223462)

[7. 부록 17](#_Toc36223463)

[1) 최대값과 최소값 17](#_Toc36223464)

[2) 유효숫자 자릿수 17](#_Toc36223465)

[3) Infinity 18](#_Toc36223466)

[4) NaN 19](#_Toc36223467)

[8. 과제 20](#_Toc36223468)

[1) pages.js 20](#_Toc36223469)

[2) random.js 20](#_Toc36223470)

# 약타입 언어(week typed language)

## 약타입 언어와 강타입 언어

(weak typed langauge, strong typed language)

javascript는 약타입 언어이다.

약타입 언어는 변수의 자료형(type)을 엄격하게 구분하지 않는다.

java 언어는 강타입 언어이다.

java는 변수의 자료형을 엄격하게 구분한다.

### operator1.js

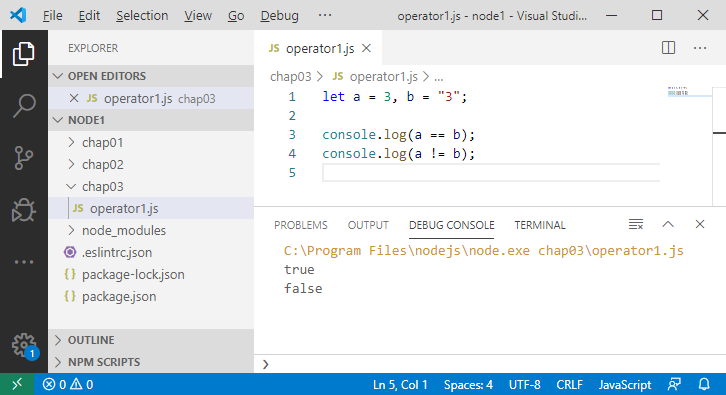
|  |
| --- |
| let a = 3, b = "3";  console.log(a == b);  console.log(a != b); |

위 코드에서 변수 a 값은 숫자 3 이고, 변수 b 값은 문자열 "3" 이다.

javascript는 숫자 타입(number type)과 문자열 타입(string type)을 엄격하게 구분하지 않기 때문에,

숫자 3 과 문자열 "3" 이 같은지 비교하면, 같다고 나온다.

a == b 의 결과는 true 이다.



## === 연산자, !== 연산자

=== 연산자는 == 연산자와 기능이 같고,

!== 연산자는 != 연산자와 기능이 같다.

== 연산자나 != 연산자로 비교할 때는 타입을 엄격하게 비교하지 않지만,

=== 연산자나 !== 연산자로 비교할 때는 타입을 엄격하게 비교한다.

### operator2.js

|  |
| --- |
| let a = 3, b = "3";  console.log(a === b);  console.log(a !== b); |

a와 b는 타입이 다르기 때문에,

a === b 의 결과는 false 이고, a !== b 의 결과는 true 이다.

## typeof 연산자

typeof 연산자를 이용하여, 값의 타입을 확인할 수 있다.

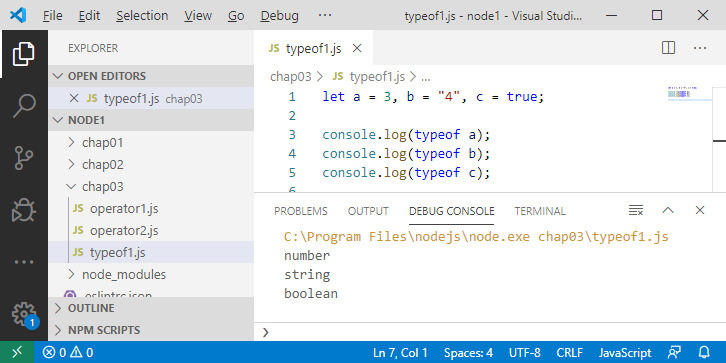
### typeof1.js

|  |
| --- |
| let a = 3, b = "4", c = true;  console.log(typeof a);  console.log(typeof b);  console.log(typeof c); |

위 예제에서 변수 a의 타입은 number 이고, typeof a 표현식의 값은 "number" 이다.

위 예제에서 변수 b의 타입은 string 이고, typeof b 표현식의 값은 "string" 이다.

위 예제에서 변수 c의 타입은 boolean 이고, typeof c 표현식의 값은 "boolean" 이다.



### typeof2.js

|  |
| --- |
| let a = 3;  console.log(typeof a == "number");  a = "hello";  console.log(typeof a == "string");  a = true;  console.log(typeof a == "boolean"); |

강타입 언어인 java 에서는, int 변수에 int 값만 대입할 수 있다.

약타입 언어인 javascript 에서는, 어떤 변수에 number, string, boolean 값을 번갈아 대입해도 된다.

|  |
| --- |
|  |

## undefined

javascript 언어에 undefined 키워드가 있다.

아직 값이 대입되지 않은 변수의 값은 undefined 이다.

### undefined1.js

|  |
| --- |
| let a;  console.log(a);  console.log(typeof a);  console.log(a == undefined);  console.log(typeof a == "undefined"); |

변수 a 에 아무 값도 대입되지 않았기 때문에, a 의 값은 undefined 이고, a 의 타입도 undefined 이다.

|  |
| --- |
|  |

## null

javascript 에도 java 처럼 null 키워드가 있다.

null 값의 의미는 java와 같다.

### null.js

|  |
| --- |
| let s = null;  console.log(s);  console.log(s == null); |

변수에 null 값을 대입할 수 있고, 변수 값이 null 인지 비교할 수 있다.

|  |
| --- |
|  |

### undefined vs null

undefined 와 null 은 둘 다 변수 값이 없음을 의미한다.

즉 undefined 와 null 의 의미는 거의 비슷하다.

약간의 차이점은 다음과 같다.

null : 값이 있고, 그 값이 null 이다.

undefined : 값이 없다.

|  |  |  |
| --- | --- | --- |
| **not null**    휴지가 있다 | **null**    휴지가 없다 | **undefined**    아무것도 없다 |

# javascript 자료형

## 기본 자료형 (primitive type)

javascript 언어의 기본 자료형은 다음과 같다.

- number

- string

- boolean

- undefined

정수(int)형과 실수형을 구분하지 않음에 주목하자.

undefined 도 자료형임에 주목하자.

## 복잡 자료형 (complex type)

javascript 언어의 복합 자료형은 다음과 같다.

- function

- object

- javascript 언어에서는 함수(function)가 자료형임에 주목하자.

javascript 언어에서는 함수가 값이라는 말이다.

함수를 변수에 대입할 수 있고, 파라미터로 함수를 전달할 수도 있다.

어떤 변수 값의 자료형이 함수(function)일 수 있다.

# boolean 자료형

## true, false 값

### boolean1.js

|  |
| --- |
| let a = false;  let b = (4 > 3);  console.log(a);  console.log(b);  console.log(typeof a);  console.log(typeof b); |

출력

|  |
| --- |
| false  true  boolean  boolean |

## false 에 해당하는 값

java 언어에서는. if 문이나 while 문의 조건식에, 반드시 boolean 타입의 조건식만 나올 수 있다.

javascript 언어에서는 그렇지 않다.

javascript 언어에서는 boolean 타입의 아닌 표현식도,

if 문이나 while 문의 조건식에 사용될 수 있다.

### 조건식에 사용된 값이 boolean 타입이 아닌 경우

그 값이 NaN, 0, null, undefined 이면, false로 취급된다.

그외 나머지 값은 true로 취급된다.

### boolean2.js

|  |
| --- |
| let a = undefined, b = null, c = 0, d = NaN;  console.log(a ? "true" : "false");  console.log(b ? "true" : "false");  console.log(c ? "true" : "false");  console.log(d ? "true" : "false"); |

출력

|  |
| --- |
| false  false  false  false |

### 조건식 ? 값1 : 값2

조건식이 true 이면, 결과는 값1 이고,

조건식이 false 이면, 결과는 값2 이다.

## lazy evaluation

&& || 연산자 표현식에서 실행할 필요가 없는 부분을 실행하지 않는 것을 lazy evaluation 이라고 한다.

C, Java, Javascript 등 대부분의 언어는 lazy evalution 한다.

|| 연산자 (OR 연산자)

&& 연산자 (&& 연산자)

### 표현식1 || 표현식2

표현식1 값이 true 이면, 결과는 true 이다. 이때 표현식2는 실행되지 않는다.

표현식1 값이 false 이면, 결과는 표현식2의 값이다. 이때는 당연히 표현식2가 실행된다.

### 표현식1 && 표현식2

표현식1 값이 flase 이면, 결과는 false 이다. 이때 표현식2는 실행되지 않는다.

표현식1 값이 true 이면, 결과는 표현식2의 값이다. 이때는 당연히 표현식2가 실행된다.

# number 자료형

## 정수 실수 구분이 없다

javascript 언어에서 숫자는, 정수(int)형과 실수형을 구분하지 않고, 그냥 number 타입이다.

숫자 타입 변수의 크기는 64bit 이고, 실수 형태로 저장된다.

### number1.js

|  |
| --- |
| let a = 100;  console.log(a);  a = 3.14159;  console.log(a); |

출력

|  |
| --- |
| 100  3.14159 |

## 형변환 (string -> number)

string 타입을 number 타입으로 변환하기 위한 함수는 다음과 같다.

- parseInt

- parseFloat

- Number

parseInt 함수는 소숫점 아래 값은 버린다.

Number 함수 이름의 첫문자가 대문자임에 주의하자.

### parse.js

|  |
| --- |
| let s = "123.456";  let a = parseInt(s);  console.log(a);  let b = parseFloat(s);  console.log(b);  let c = Number(s);  console.log(c); |

출력

|  |
| --- |
| 123  123.456  123.456 |

## 형변환 (number -> string)

number 타입을 string 타입으로 변환하기 위한 함수는 다음과 같다.

- toString

- String

String 함수 이름의 첫문자가 대문자임에 주의하자.

### toString.js

|  |
| --- |
| let a = 123.456;  let s = a.toString();  console.log(s);  console.log(typeof s);  s = String(a);  console.log(s);  console.log(typeof s); |

출력

|  |
| --- |
| 123.456  string  123.456  string |

# Math 클래스

## PI - 원주율

### pi.js

|  |
| --- |
| console.log(Math.PI); |

출력

|  |
| --- |
| 3.141592653589793 |

## round - 반올림 메소드

### round.js

|  |
| --- |
| console.log(10 / 3);  console.log(Math.round(10 / 3));  console.log(10 / 4);  console.log(Math.round(10 / 4)); |

출력

|  |
| --- |
| 3.3333333333333335  3  2.5  3 |

## pow - 거듭제곱 메소드

### pow.js

|  |
| --- |
| console.log(Math.pow(2, 3));  console.log(Math.pow(2, 8));  console.log(Math.pow(2, 10)); |

출력

|  |
| --- |
| 8  256  1024 |

## sqrt - 제곱근 메소드

### sqrt.js

|  |
| --- |
| console.log(Math.sqrt(9));  console.log(Math.sqrt(16)); |

출력

|  |
| --- |
| 3  4 |

## abs - 절대값 메소드

### abs.js

|  |
| --- |
| console.log(Math.abs(-3)); |

출력

|  |
| --- |
| 3 |

## floor/ceil - 내림/올림 메소드

### ceil.js

|  |
| --- |
| console.log(Math.floor(4.01));  console.log(Math.floor(4.99));  console.log(Math.ceil(4.01));  console.log(Math.ceil(4.99)); |

출력

|  |
| --- |
| 4  4  5  5 |

내림 : 소수점 아래 값을 버린다.

올림 : 소수점 아래 값이 있으면, 그 값을 버리고 1을 올린다.

## sin, cos, tan - 삼각함수 메소드

### sin.js

|  |
| --- |
| console.log(Math.sin(Math.PI / 2)); // sin 90도  console.log(Math.cos(Math.PI));     // cos 180도 |

출력

|  |
| --- |
| 1  -1 |

## max, min - 최대 최소 메소드

### max.js

|  |
| --- |
| console.log(Math.max(2, 3));  console.log(Math.max(2, 3, 5));  console.log(Math.min(2, 3));  console.log(Math.min(2, 3, 5)); |

출력

|  |
| --- |
| 3  5  2  2 |

max 메소드: 파라미터 목록에서 최대값을 리턴한다.

min 메소드: 파라미터 목록에서 최소값을 리턴한다.

## random - 난수 메소드

### random.js

|  |
| --- |
| for (let i = 0; i < 20; ++i) {    let a = Math.floor(Math.random() \* 6 + 1);    console.log(a);  } |

Math.random 메소드는, 0 보다 크거나 같고, 1보다 작은 난수를 리턴한다. (0 <= 리턴값 < 1)

Math.floor(Math.random() \* 6 + 1)

이 계산식의 값은 1, 2, 3, 4, 5, 6 중의 하나이다.

### random2.js

|  |
| --- |
| const MAX = 5, MIN = 2;  for (let i = 0; i < 20; ++i) {    let a = Math.floor(Math.random() \* (MAX - MIN + 1) + MIN);    console.log(a);  } |

Math.random() \* (MAX - MIN + 1) + MIN

이 계산식의 값은 MIN 보다 크거나 같고, MAX 보다 작거나 같은 정수들 중의 하나이다.

# 요약

== 연산자나 != 연산자로 비교할 때는 타입을 엄격하게 비교하지 않지만,

=== 연산자나 !== 연산자로 비교할 때는 타입을 엄격하게 비교한다.

(3 == "3") 표현식의 값은 true 이다.

(3 === "3") 표현식의 값은 false 이다.

typeof 연산자를 이용하여, 값의 타입을 확인할 수 있다.

typeof 3 표현식의 값은 "number" 이다.

typeof "hello" 표현식의 값은 "string" 이다.

typeof true 표현식의 값은 "boolean" 이다.

아직 값이 대입되지 않은 변수의 값은 undefined 이다.

javascript 언어의 기본 자료형은 다음과 같다.

number, string, boolean, undefined

&& || 연산자 표현식에서 실행할 필요가 없는 부분을 실행하지 않는 것을 lazy evaluation 이라고 한다.

C, Java, Javascript 등 대부분의 언어는 lazy evalution 한다.

표현식1 || 표현식2

표현식1 값이 true 이면, 결과는 true 이다. 이때 표현식2는 실행되지 않는다.

표현식1 값이 false 이면, 결과는 표현식2의 값이다. 이때는 당연히 표현식2가 실행된다.

표현식1 && 표현식2

표현식1 값이 flase 이면, 결과는 false 이다. 이때 표현식2는 실행되지 않는다.

표현식1 값이 true 이면, 결과는 표현식2의 값이다. 이때는 당연히 표현식2가 실행된다.

javascript 언어의 복합 자료형은 다음과 같다.

function, object

조건식에 사용된 값이 boolean 타입이 아닌 경우

그 값이 NaN, 0, null, undefined 이면, false로 취급된다.

그외 나머지 값은 true로 취급된다.

javascript 언어에서 숫자는, 정수(int)형과 실수형을 구분하지 않고, 그냥 number 타입이다.

string 타입을 number 타입으로 변환하기 위한 함수는 다음과 같다.

parseInt, parseFloat, Number

parseInt 함수는 소숫점 아래 값은 버린다.

number 타입을 string 타입으로 변환하기 위한 함수는 다음과 같다.

toString, String

Math 클래스

PI - 원주율

round - 반올림 메소드

pow - 거듭제곱 메소드

sqrt - 제곱근 메소드

abs - 절대값 메소드

floor/ceil - 내림/올림 메소드

sin, cos, tan - 삼각함수 메소드

max, min - 최대 최소 메소드

random - 난수 메소드

# 부록

이 수업의 학습목표에 속하지 않는 내용.

## 최대값과 최소값

### number2.js

|  |
| --- |
| console.log(Number.MAX\_VALUE);  console.log(Number.MIN\_VALUE);  console.log(-Number.MAX\_VALUE);  console.log(-Number.MIN\_VALUE); |

출력

|  |
| --- |
| 1.7976931348623157e+308  5e-324  -1.7976931348623157e+308  -5e-324 |

최대값: 1.7976931348623157 \* 10308

최소값: -1.7976931348623157 \* 10308

최소 절대값: 5 \* 10-324

## 유효숫자 자릿수

숫자 타입 변수의 크기는 64bit 이고, 실수 형태로 저장되기 때문에, 유효숫자 자릿수 문제가 발생한다.

정수는 15자리까지 정확하다.

15자리가 넘는 정수는 유효숫자 자릿수 부족 때문에 오차가 발생한다.

### significantDigits.js

|  |
| --- |
| let a;  a = 999999999999999; // 15 자리  console.log(a);  a = 9999999999999999; // 16 자리  console.log(a);  a = 0.9999999999999999; // 16 자리  console.log(a);  a = 0.99999999999999999; // 17 자리  console.log(a); |

출력

|  |
| --- |
| 999999999999999  10000000000000000  0.9999999999999999  1 |

## Infinity

javascript에서 Infinity 키워드는 무한대에 해당하는 값을 의미한다.

### infinity1.js

|  |
| --- |
| let a = Infinity;  console.log(a);  console.log(typeof a);  let b = 3 / 0;  console.log(b);  console.log(typeof b); |

출력

|  |
| --- |
| Infinity  number  Infinity  number |

3 / 0 계산 결과 값이 Infinity 임에 주목하자.

### infinity2.js

|  |
| --- |
| let a = Infinity;  let b = 3 / a;  console.log(b);  let c = 3 + a;  console.log(c); |

출력

|  |
| --- |
| 0  Infinity |

3 / 무한대 계산 결과는 0 이다.

3 + 무한대 계산 결과는 무한대이다.

## NaN

javascript에서 NaN 키워드는 계산 결과가 숫자가 아님을 의미한다. (Not a Number)

### NaN.js

|  |
| --- |
| let a = 3 \* "ABC";  console.log(a);  console.log(typeof a);  let b = Number("abc");  console.log(b);  console.log(typeof b); |

출력

|  |
| --- |
| NaN  number  NaN  number |

# 과제

## pages.js

한 페이지에 10명의 학생 정보를 출력한다고 할 때,

30명의 정보를 출력하려면, 3 페이지,

31명의 정보를 출력하려면, 4 페이지가 필요하다.

let studentCount = 31;

let pageSize = 10;

위와 같이 학생수와 페이지 크기가 각각 주어졌을 때,

페이지 수를 계산해서 출력하는 코드를 구현하시오.

### 답

|  |
| --- |
| let studentCount = 31;  let pageSize = 10;  let pageCount = Math.ceil(studentCount / pageSize);  console.log(pageCount); |

## random.js

10 이상 20 이하의 정수를 임의로 30개 출력하는 코드를 구현하시오.

### 답

|  |
| --- |
| let min = 10;  let max = 20;  for (let i = 0; i < 30; ++i) {    let n = Math.floor(Math.random() \* (max - min + 1) + min);    console.log(n);  } |