JavaScript

Number – 김근형 강사

Number

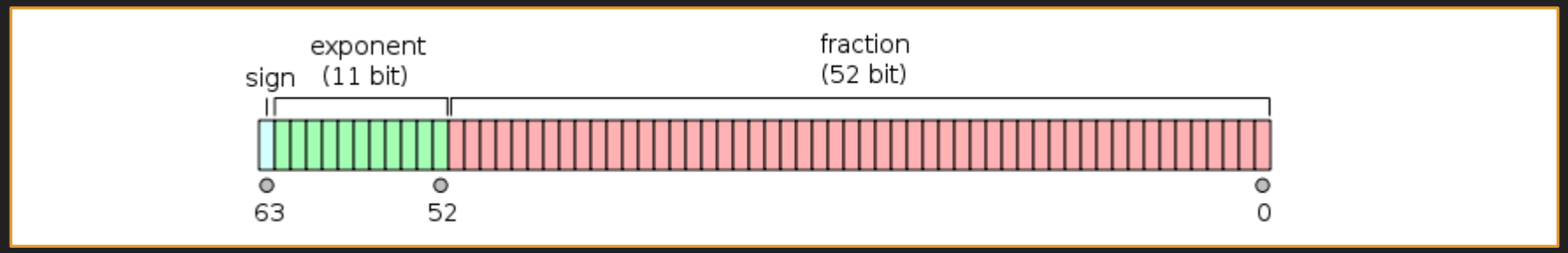
- Number 상수의 추가
  - 아래와 같은 두개의 Number 상수가 추가되었다.

상수이름	값
Number.MAX_SAFE_INTEGER	9007199254740991(2^{53}-1)
Number.MIN_SAFE_INTEGER	-9007199254740991(-(2^{53}-1))

  - 위의 두개의 상수는 안전 정수를 나타내며 안전 정수(safe integer)란 지수(e)를 사용하지 않고 나타낼 수 있는 값을 나타낸다.

Number

- 64비트 유동소수점
  - 자바스크립트는 IEEE(Institute of Electrical and Electronics Engineers) 754에 정의된 “double-precision floating-point format number”로 숫자 값을 표현한다.
  - 이는 64비트 유동 소수점으로 정수와 소수를 포함하여 64개의 비트로 값을 표현한다.



  - 사인 비트 (sign bit) : 양수 음수를 나타내고자 할 때 쓰이는 비트(1이면 음수, 0이면 양수)
  - 지수(exponent) : 숫자가 넘어갈 경우 지수로 표현하기 위해 쓰이는 공간
  - 유효 숫자(fraction) : 최대 나타낼 수 있는 유효 숫자

Number

- 64비트 유동소수점

```
// 9007199254740991
console.log("1:", Number.MAX_SAFE_INTEGER);
console.log("2:", Math.pow(2, 53) - 1);

// -9007199254740991
console.log("3:", Number.MIN_SAFE_INTEGER );
console.log("4:", -(Math.pow(2, 53) - 1));
```

```
1: 9007199254740991
2: 9007199254740991
3: -9007199254740991
4: -9007199254740991
```

Number

- EPSILON
  - 1보다 큰 값에서 최소값과 1과의 차이를 나타낸다.
  - 유동 소수점 처리에서 미세한 값 차이 때문에 일치하지 않을 때 사용한다.

상수 이름	값
Number.EPSILON	$2.220446049250313080847263336181 6 X 10^{-16}$

Number

- EPSILON

```
let total = 0.1 + 0.2;
console.log(total);

let result = (Math.abs(0.1 + 0.2 - 0.3) < Number.EPSILON);
console.log(result);

let value = (Math.pow(10, 1) * 0.1) + (Math.pow(10, 1) * 0.2);
console.log(value / 10 === 0.3);
```

```
0.30000000000000004
true
true
```

Number

- 진수 리터럴
  - ES6에 2진수 표기법이 추가되었으며 8진수 표기법이 재정의 되었다.
    - 2진수 : 첫번째 숫자 0을 작성하고 두번째 소문자에b 또는 대문자B를 작성한다. 세번째부터 값을 0 또는 1로 작성한다.
    - 8진수 : 첫번째에 숫자 0을 작성하고 두번째에 소문자 o 또는 대문자 O를 작성한다. 세번째부터 값을 0에서 7까지 작성한다.

```
// 2진수
let two = 0b0101;
console.log(two);

// 8진수
let eight = 0o0101;
console.log(eight);
```

```
5
65
```

Number

- isNaN(): NaN 여부
  - 파라미터 값이 NaN(Not a Number)이면 true를 반환하고, 아니면 false를 반환한다.

구분	타입	데이터(값)
형태		Number.isNaN()
파라미터	Any	비교대상
반환	Boolean	NaN이면 true, 아니면 false

  - 기존의 체크 결과가 false로 반환되는 점이 있어 ES5에서 글로벌 오브젝트에 isNaN()을 추가하였지만 이또한 완전하지 못해 ES6에서 Number에 추가함.
  - NaN을 체크하는 방법은 1. NaN === NaN / 2. 글로벌 오브젝트의 isNaN() / 3. Number.isNaN()

Number

- isNaN(): NaN 여부

```
// true, true
console.log("1:", Number.isNaN(NaN), isNaN(NaN));
// false, true
console.log("2:", Number.isNaN("NaN"), isNaN("NaN"));
// false, true
console.log("3:", Number.isNaN("ABC"), isNaN("ABC"));
// false, true
console.log("4:", Number.isNaN(undefined), isNaN(undefined));
// false, true
console.log("5:", Number.isNaN({}), isNaN({}));
// true, true
console.log("6:", Number.isNaN(Number.NaN), isNaN(Number.NaN));
// true, true
console.log("7:", Number.isNaN(0 / 0), isNaN(0 / 0));
// false, false
console.log("8:", Number.isNaN(true), isNaN(true));
// false, false
console.log("9:", Number.isNaN(null), isNaN(null));
// false, false
console.log("A:", Number.isNaN(""), isNaN(""));
```

Number

- isInteger() : 정수여부
  - 파라미터 값이 정수이면 true를 반환하고 아니면 false를 반환한다.

구분	타입	데이터(값)
형태		Number.isInteger()
파라미터	Any	비교대상
반환	Boolean	정수이면 true, 아니면 false

```
// 이하 true
console.log("1:", Number.isInteger(0));
console.log("2:", Number.isInteger(1.0));
console.log("3:", Number.isInteger(-123));

// 이하 false
console.log("4:", Number.isInteger("12"));
console.log("5:", Number.isInteger(1.02));
console.log("6:", Number.isInteger(NaN));
console.log("7:", Number.isInteger(true));
```

Number

- isSafeInteger() : 안전정수여부
  - 파라미터 값이 $2^{53}-1$ ~ $-(2^{53}-1)$이면 true를 반환하고 아니면 false를 반환한다.

구분	타입	데이터(값)
형태		Number.isSafeInteger()
파라미터	Any	비교대상
반환	Boolean	$2^{53}-1$ ~ $-(2^{53}-1)$이면 true, 아니면 false

```
// true
console.log("1:", Number.isSafeInteger(7));
console.log("2:", Number.isSafeInteger(7.0));
console.log("3:", Number.isSafeInteger(Number.MAX_SAFE_INTEGER));
console.log("4:", Number.isSafeInteger(Number.MIN_SAFE_INTEGER));

// false
console.log("5:", Number.isSafeInteger(7.1));
console.log("6:", Number.isSafeInteger("123"));
console.log("7:", Number.isSafeInteger(Number.MAX_SAFE_INTEGER + 1));
console.log("8:", Number.isSafeInteger(Number.MIN_SAFE_INTEGER - 1));
```

Number

- isFinite() : 유한 값 여부
  - 파라미터 값이 유한 값 이면 true를 반환하고 아니면 false를 반환한다.

구분	타입	데이터(값)
형태		Number.isFinite()
파라미터	Any	비교대상
반환	Boolean	유한 값 이면 true, 아니면 false

  - 글로벌 오브젝트에도 isFinite()가 있으며 Number.isFinite()와는 차이가 있다.

Number

- isFinite() : 유한 값 여부

```
console.log("1:", Number.isFinite(Infinity), isFinite(Infinity));
console.log("2:", Number.isFinite(-Infinity), isFinite(-Infinity));

console.log("3:", Number.isFinite(0), isFinite(0));
console.log("4:", Number.isFinite("0"), isFinite("0"));

console.log("5:", Number.isFinite(null), isFinite(null));
console.log("6:", Number.isFinite(NaN), isFinite(NaN));

console.log("7:", Number.isFinite(undefined), isFinite(undefined));
console.log("8:", Number.isFinite(true), isFinite(true));
```

```
1: false false
2: false false
3: true true
4: false true
5: false true
6: false false
7: false false
8: false true
```