Number Type In Detail

Problem

Q1: What will be the output of:

typeof -5

Q2: What will be the output of:

typeof -5.2

Q3: What will be the output of:

X = 5
Y = 5.0
console.log(X === Y)

Solution

```
> x = 5
< 5
> y = 5.0
< 5
> console.log(x === y)
  true
```

The ‘===’ checks for the equality of value of the variable and the type of the variable.

Problem

Is there any difference between data type of a floating point number and an integer?

toFixed()

Declaring Floating point value in JavaScript :-

```
var x = 5.7894;
console.log(x.toFixed(2));
```

Output: “5.78”
Returns a String value.

toExponential()

Syntax

```
number.toExponential(x)
```

Parameters

Parameter	Description
x	Optional. An integer between 0 and 20 representing the number of digits in the notation after the decimal point. If omitted, it is set to as many digits as necessary to represent the value

Return Value

A String, representing the number as an exponential notation.

```
let num = 5.56789;
let n = num.toExponential(3);

Out: 5.568e+0

~~~

let num = 100

let n = num.toExponential(3);

n
'1.000e+2'
```

ParseInt() : Part 1

Parse a string and returns the first integer:

Number.parseInt("10")	10
Number.parseInt("10.00")	10
Number.parseInt("10.33")	10
Number.parseInt("34 45 66")	34
Number.parseInt(" 60 ")	60
Number.parseInt("40 years")	40
Number.parseInt("He was 40")	NaN

ParseInt() : Part 2

The Number.parseInt method parses a value as a string and returns the first integer.

A radix parameter specifies the number system to use:

2 = binary, 8 = octal, 10 = decimal, 16 = hexadecimal.

If radix is omitted, JavaScript assumes radix 10. If the value begins with "0x", JavaScript assumes radix 16.

Notes: If the first character cannot be converted, NaN is returned.

Leading and trailing spaces are ignored.

Only the first integer found is returned.

Example with radix

```
let s = "101"

parseInt(s, 2)
5
```

parseFloat()

Parse a string and returns the first number:

Number.parseFloat(10)	10
Number.parseFloat("10")	10
Number.parseFloat("10.33")	10.33
Number.parseFloat("34 45 66")	34
Number.parseFloat("He was 40")	NaN

toString()

How can we convert decimal to binary notation?

```
let num = 15;
let text = num.toString(2);
```

Convert a number to a string using base 2:
1111

Problem

1. Convert 11 to it’s binary representation.

2. Convert the binary ‘1101’ to it’s decimal form.

Solution

Convert an decimal to binary:

```
let num = 17;
num.toString(2);
```

'10001'

Convert a binary to decimal:

```
let x = '1101'
parseInt(x, 2)
```

13

Some Constants

```
> Number.MAX_VALUE
< 1.7976931348623157e+308
> Number.MIN_VALUE
< 5e-324
> Number.POSITIVE_INFINITY
< Infinity
> Number.MAX_SAFE_INTEGER
< 9007199254740991
> Number.MIN_SAFE_INTEGER
< -9007199254740991
```

What is ‘SAFE’ in MAX_SAFE_INTEGER and MIN_SAFE_INTEGER?

Double precision floating point format only has 52 bits to represent the mantissa, so it can only safely represent integers between -(2**53 – 1) and 2**53 – 1. "Safe" in this context refers to the ability to represent integers exactly and to compare them correctly.

For example, Number.MAX_SAFE_INTEGER + 1 === Number.MAX_SAFE_INTEGER + 2 will evaluate to true, which is mathematically incorrect.

See Number.isSafeInteger() for more information.

Because MAX_SAFE_INTEGER is a static property of Number, you always use it as Number.MAX_SAFE_INTEGER, rather than as a property of a number value.

Problem

Can you give one use case for POSITIVE_INFINITY and NEGATIVE_INFINITY?

Answer: During overflow and underflow.

Number.MAX_VALUE + 1
1.7976931348623157e+308

Number.MAX_SAFE_INTEGER + 1 === Number.MAX_SAFE_INTEGER + 2
True (Incorrect because you cannot compare numbers outside of their safe values.)

Number.MAX_VALUE * 2
Infinity

Number.MAX_VALUE * 10
Infinity

Number.MAX_VALUE + Number.MAX_VALUE
Infinity

How many digits are there in the Number.MAX_VALUE?

Number.MAX_VALUE
1.7976931348623157e+308

And we saw that 100 is represented like below:

'1.000e+2'

100 has three digits and powers of 10 in it’s exponential notation is 2.

Answer:

309 Digits

List of ‘Number’ Methods and Constants

Name	Description
constructor	Returns the function that created JavaScript's Number prototype
EPSILON	Returns the difference between 1 and the smallest number greater than 1
isFinite()	Checks whether a value is a finite number
isInteger()	Checks whether a value is an integer
isNaN()	Checks whether a value is Number.NaN
isSafeInteger()	Checks whether a value is a safe integer
MAX_SAFE_INTEGER	Returns the maximum safe integer in JavaScript.
MIN_SAFE_INTEGER	Returns the minimum safe integer in JavaScript
MAX_VALUE	Returns the largest number possible in JavaScript
MIN_VALUE	Returns the smallest number possible in JavaScript

NaN	Represents a "Not-a-Number" value
NEGATIVE_INFINITY	Represents negative infinity (returned on overflow)
POSITIVE_INFINITY	Represents infinity (returned on overflow)
parseFloat()	Parses a string an returns a number
parseInt()	Parses a string an returns a whole number
prototype	Allows you to add properties and methods to an object
toExponential(x)	Converts a number into an exponential notation
toFixed(x)	Formats a number with x numbers of digits after the decimal point
toLocaleString()	Converts a number into a string, based on the locale settings
toPrecision(x)	Formats a number to x length
toString()	Converts a number to a string
valueOf()	Returns the primitive value of a number

Name	Description
abs(x)	Returns the absolute value of x
acos(x)	Returns the arccosine of x, in radians
acosh(x)	Returns the hyperbolic arccosine of x
asin(x)	Returns the arcsine of x, in radians
asinh(x)	Returns the hyperbolic arcsine of x
atan(x)	Returns the arctangent of x as a numeric value between -PI/2 and PI/2 radians
atan2(y, x)	Returns the arctangent of the quotient of its arguments
atanh(x)	Returns the hyperbolic arctangent of x
cbrt(x)	Returns the cubic root of x
ceil(x)	Returns x, rounded upwards to the nearest integer
clz32(x)	Returns the number of leading zeros in a 32-bit binary representation of x
cos(x)	Returns the cosine of x (x is in radians)
cosh(x)	Returns the hyperbolic cosine of x
E	Returns Euler's number (approx. 2.718)
exp(x)	Returns the value of E^x
expm1(x)	Returns the value of E^x minus 1

Math library of JavaScript

Ref: https://www.w3schools.com/jsref/jsref_obj_math.asp