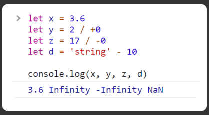
PRIMITIVE DATA TYPES IN JAVASCRIPT

by @eyeshreya

Number

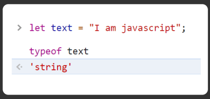
Possible Values

floating-point numbers, +∞, -∞, NaN



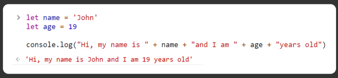
```
> let x = 3.6
  let y = 2 / +0
  let z = 17 / -0
  let d = 'string' - 10

  console.log(x, y, z, d)
  3.6 Infinity -Infinity NaN
```

String



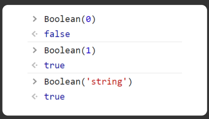
```
> let text = "I am javascript";

  typeof text
<· 'string'
```

String Concatenation in JavaScript



```
> let name = 'John'
  let age = 19

  console.log("Hi, my name is " + name + "and I am " + age + "years old")
<· 'Hi, my name is John and I am 19 years old'
```

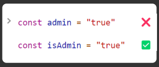
Boolean



```
> Boolean(0)
<· false
> Boolean(1)
<· true
> Boolean('string')
<· true
```

Boolean Tip

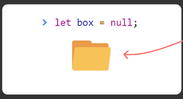
use prefix like "is" while naming variables for a better understanding



```
> const admin = "true"      ✗

  const isAdmin = "true"    ✓
```

Null

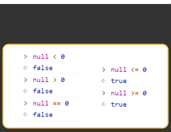
Assigning "null" value to a variable simply means at the moment this variable is empty but does exist, and it will be having a value in future.



```
> let box = null;
```

an empty folder exists, that can store a value in future

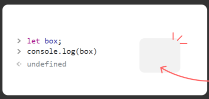
Value of null?



```
> null < 0
<· false             > null <= 0
> null > 0           <· true
<· false             > null >= 0
> null == 0          <· true
<· false
```

Undefined

Assigning "undefined" value to a variable means that this variable has not been defined yet.



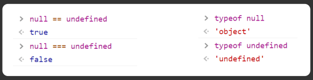
```
> let box;
> console.log(box)
<· undefined
```

where's the folder?

Null vs Undefined

source: Reddit

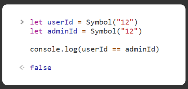
Are both equal?



```
> null == undefined          > typeof null
<· true                      <· 'object'
> null === undefined         > typeof undefined
<· false                     <· 'undefined'
```

Symbol

A unique identifier.
Can't be changed so they can also be used as a key in objects.

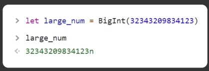


```
> let userId = Symbol("12")
  let adminId = Symbol("12")

  console.log(userId == adminId)

<· false
```

Even when we've assigned same content to userId and adminId, both are not equal to each other.

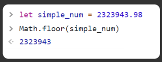
BigInt

Used to store big values.



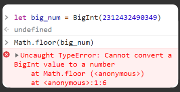
```
> let large_num = BigInt(32343209834123)
> large_num
<· 32343209834123n
```

Number vs BigInt



```
> let simple_num = 2323943.98
> Math.floor(simple_num)
<· 2323943
```

Built-in Math objects can be used.



```
> let big_num = BigInt(2312432490349)
<· undefined
> Math.floor(big_num)
✖ ▶Uncaught TypeError: Cannot convert a
  BigInt value to a number
      at Math.floor (<anonymous>)
      at <anonymous>:1:6
```

Built-in Math objects can not be used.