<https://www.toptal.com/javascript/interview-questions>

<https://www.ajaymatharu.com/javascript-difference-between-undefined-and-null/>

var a=null;

>>undefined

if(a){console.log("True")} else{console.log("False")}

>> False

var f;

>>undefined

if(f){console.log("True")} else{console.log("False")}

>>False

Typeof  => returns string

undefined

>>undefined

typeof undefined

>>"undefined"

null

>>null

typeof null

>>"object"

NULL

>> NULL is not defined

3<2   => False => 0

3>2  => True => 1

**== Vs ===**

<https://codeburst.io/javascript-double-equals-vs-triple-equals-61d4ce5a121a>

Strict equality === both type and value

Loose equality ==  type coercion and then checking

There are only six falsy values in JavaScript

false, 0, “”, null, undefined NaN

NaN === NaN // false

**& (bitwise and) &&(logical AND):**

<https://stackoverflow.com/questions/7310109/whats-the-difference-between-and-in-javascript>

Operator chaining:

It expects two arguments and returns:

* First term that evaluates to false
* Last term otherwise (if all are true-y)

Here are some examples:

0 && false 0 (both are false-y, but 0 is the first) true && false false (second one is false-y) true && true true (both are true-y) true && 20 20 (both are true-y)

This definition conforms to the definition of logical and from the field of mathematics.

“”/2  gives 0

**object constructor function**

<https://www.w3schools.com/js/js_object_constructors.asp>

**NAN**

<https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/isNaN#Confusing_special-case_behavior>

Number.isNAN()- use this

isNaN() - does a type coercion before checking for equality

The **Number.EPSILON** property represents the difference between 1 and the smallest floating point number greater than 1.

x = 0.2;  
y = 0.3;  
z = 0.1;  
equal = (Math.abs(x - y + z) < Number.EPSILON);

function isInteger(x) { return (x ^ 0) === x; }

function isInteger(x) { return Math.round(x) === x; }

function isInteger(x) { return (typeof x === 'number') && (x % 1 === 0); }