1. What is the value of typeof typeof true?

Ans :

let a=10;

console.log(typeof typeof a);

output: string

Reason: here, typeof a value gives as “number” first, again typeof is became number, therefore when we typeof for 2nd time. It will be gives string

So, output will be string

1. What is the result of null == undefined?

Ans:

console.log(null == undefined);

output: true

reason: here both null and undefined are belongd to primitive data type.

“==” operator will tries to compare them into common type. By this they belong to same datatype. Hence, it will gives output as true

**“==”**operator (also known as the "loose equality" operator)

1. What is the value of +''?

Here, unary plus oprationis performed to an empty string. So, here generally these operation applied to a string it converts to a number. But here string is empty it returns “0”

// What is the value of +''?

console.log(+'');

output: 0

1. What is the result of [] == false?

[] is empty array, false is Boolean value, when compare them with “==” operator initially these operator converts to Boolean value before comparing it with boolean value.

So, when you compare [] with false using the == operator,

the empty array is converted to true,

and the comparison returns false. By using toboolean abstract operation.

False==false gives comparison “true”

// What is the result of [] == false?

console.log([]==false);

output: true

1. 5. What is the value of {} + {}?

{} {} are the objects with concadination operater “+”.

“+” operator it will try to convert hem into strings and add them (concadinate them)

Here, return “[object object]” converted using ‘tostring()’ method

Finally, we get [object object] [object object]

//What is the value of {} + {}?

console.log({}+{});

output: [object Object][object Object]

1. What is the value of typeof 42n?

Ans: here 42n is the n might be bigger or smaller number but must be belongs to biginter wereas 42n is bigint bilateral consider comes to large integer.

let b=42;

console.log(typeof b);

output: Bigint

1. What is the result of "" == false?

Return output as true because both belong to same family of primitive datatypes

Along it is empty string and other is Boolean value”false” here, == converts and compare to boolen value. Here, empty string retuns “true” and when compare with “==” gives result false. False==false is “true”

note: empty considered as falsy value in simple

// What is the result of "" == false?

console.log("" == false)

output: true

1. What is the value of Boolean("0")?

In a simple way string is not an empty by this we can consider it a “true” as Boolean value beside that empty string is “false” value

// What is the value of Boolean("0")?

console.log(Boolean("0"));

output: true

1. What is the result of [] === []?

When assign two empty array which will by default consider two different memory address, So, here “===” opertoer striclyt compares the empty strings both values and type with their memory address infomtion too.

Results , is false

And it’s an error too!

// What is the result of [] === []?

console.log([]===[]);

output: false

1. What is the value of typeof Infinity?

// What is the value of typeof Infinity?

let c=Infinity;

console.log(typeof c);

output: number

it might be infinity, but comes to the number category only right!

1. What is the result of {} === {}?

Ans:

the overall expression console.log(typeof {} === {}) evaluates to false because the strict equality operator (===) in the console.log statement checks if the result (true in this case) is the exact same value in memory as false. Since they are different values, the final result is false.

// What is the result of {} === {}?

console.log(typeof {}==={})

output: false

1. What is the value of typeof NaN?

Ans: NaN: not an number, the meaning itself suggest number which elevate. So, when checks with type of NaN, We get output as Number.

// What is the value of typeof NaN?

console.log(typeof NaN)

output: number