What is a string?

What is a string?

A string is a sequence of characters. (Here each character has a position associated with it.)

When we write “hello world!” what computer sees is this:

h	e	l	l	o		w	o	r	l	d	!
0	1	2	3	4	5	6	7	8	9	10	11
j	a	c	k		s	m	i	t	h		
0	1	2	3	4	5	6	7	8	9		

String Indexing

Just a string	f	o	o	b	a	r
position	0	1	2	3	4	5
string	b	o	b			
position	0	1	2			
string	a	l	i	c	e	
position	0	1	2	3	4	
string	h	e	l	l	o	
position	0	1	2	3	4	

Through the lens of programming

JavaScript strings are for storing and manipulating text.

A JavaScript string is zero or more characters written inside quotes.
You can use single or double quotes:
You can use quotes inside a string, as long as they don't match the quotes surrounding the string:

```
let answer1 = "It's alright";
let answer2 = "He is called 'Johnny'";
let answer3 = 'He is called "Johnny"';
```

String Length

To find the length of a string, use the built-in length property:

```
let text = "ABCDEFGHIJKLMNOPQRSTUVWXYZ";
let length = text.length;
```

```
let name = 'Abhisneha'
name.length
9
```

JavaScript Strings as Objects

Normally, JavaScript strings are primitive values, created from literals:

let x = "John";

But strings can also be defined as objects with the keyword new:

let y = new String("John");

Word of Caution:
Do not create Strings objects.
The new keyword complicates the code and slows down execution speed.
String objects can produce unexpected results:

let x = "John";

X is string literal.

let y = new String("John");

Y is a string object.

Equality Between String Literal and String Object

When using the == operator, x and y are equal:

```
let x = "John";
let y = new String("John");
x==y
true
```

When using the === operator, x and y are not equal:

```
let x = "John";
let y = new String("John");
x===y
False
```

This is because “==” checks only for value while “===” checks also for the data type.

Note the difference between (x==y) and (x===y)

(x == y) true or false?

```
let x = new String("John");
let y = new String("John");
```

Returns true.

(x === y) true or false?

```
let x = new String("John");
let y = new String("John");
```

Returns false.

Comparing two JavaScript objects always returns false.

Why JavaScript objects cannot be compared?

“Comparing two objects like this results in false even if they have the same data. It is because those are two different object instances, they are referring to two different objects. There is no direct method in javascript to check whether two objects have the same data or not.”
Ref:
https://www.geeksforgeeks.org/how-to-check-two-objects-have-same-data-using-javascript/

You have two brand new pencils from the same box. They are both identical pencils but can we say:
pencil_1 === pencil_2?

That is what GeeksForGeeks is trying to explain.

The two pencils are still two independent objects, that’s why.