Objectives

1. *Introduce 5 primitive data types*
2. *Work with numbers and numeric operators*
3. *Work with string and common string methods*

One of the big ideas at the core of every programming languages is the language's ability to differentiate between different categories of data. So, a language can differentiate between a number and a word, or it can differentiate between a whole number and a fractional number, or between a positive number and a negative number. So, these vary from language to language. In JavaScript there are 5 primitive data types. There are 5 low level basic types of data, and we are going to do a quick overview of the 5 of them first and then we are going to dive into details afterwards.

//Numbers

4

9.3

-10

The first one is number 4, a whole number, 9.3, a decimal number and -10, a negative number. JavaScript does not care if a number is a whole number, a decimal number or a negative number, they are all treated as numbers. It's a broad category, other languages do differentiate between numbers by int, double etc. data-types but JavaScript does not. So, Numbers is just a category for whole numbers, decimal numbers or negative numbers.

//Strings

"Hello World"

"43"

Strings are basically texts, so the important thing is that they are inside of quotes. So, as you can see above Hello World, it is inside of quotes and 43 though it is a number but as it is inside of quotes, it is counted as a String because we can have any character inside of a string.

//Booleans

True

False

The next character type is Boolean. Booleans have two options they are either true or false. There are no quotes, there's no numbers, it is just that the word true or the word false and that is it. Booleans exist as true or false, yes or no.

//null and undefined

Null

Undefined

There are two more types Null and Undefined. These are values, they are not a category. There are no multiple types of null or multiple types of undefined, like there are for numbers or strings. There is only one null and only one undefined. There are just values.

We will learn more about them in the next topics, but right now we must know that they exist. So, Numbers, Strings, Booleans and Undefined.

Now, we will be coding on Chrome's developer's console.

Console coding:

505

➔ 505

-99

-> -99

These are very basic stuffs. So, we will be using these numbers to do more interesting things. So, the first thing we can do with these numbers are simple mathematical operations.

Console coding:

4 + 100

➔ 104

We can also work with no spaces and that works too.

3+7

➔ *10*

6/2

➔ *3*

1/3

➔ *0.333333333*

1-54

➔ *-53*

Then we also have multiplication

*2*5*

➔ *10*

Another important concept is that JavaScript follows the order of operations that all regular math follows as well.

*(3-8) * 24*

➔ *-120*

We have another important operator call Modulo. Its often call the remainder operator.

It uses the percentage sign as its symbol.

So for example if we perform 10 mod 3, then what it will do is, it will take 3 as many times it can be taken as a whole number so that will be 3 times and then it will take the remainder, so 3 goes to 10 three times, which is 9 and the remainder is 1.

10 % 3

➔ *1*

20 % 5

➔ *0*

The next data-type we are going to focus on is the string. Strings are texts, they are quotes, numbers, and characters, inside of quotes. And those quotes can be single or double quotes, so as an example,

Console coding:

"hello class!"

➔ *"hello class!"*

'dogs are awesome'

➔ *"dogs are awesome"*

As you can note that though we put our characters in single quotes we get the output in double quotes because it treats the characters same as before.

We can also find the length of a string by using the *.length* property

"HELLO".length

➔ *5*

We can also retrieve individual characters like the first character or the fifth character using the square bracket notation. So, we write the string and after that put the number in square brackets and it gives us the position of the character according to the number that we have put in the brackets.

"The Beatles"[0]

➔ *"T"*

So, as T is the first character whose position is zero, that is why we get "T".

"The Beatles"[4]

➔ *"B"*

"The Beatles"[10]

➔ *"s"*

We can see that the position of the last character is 10, where the position of the first character starts from 0, but if we want to find the length of the string it will give us 11 because it starts counting from 1.

"The Beatles".length

➔ *11*