*At the end of the day, a programming language is a tool developed by humans and, humans are weird!*

*- Dennis Ritchie*

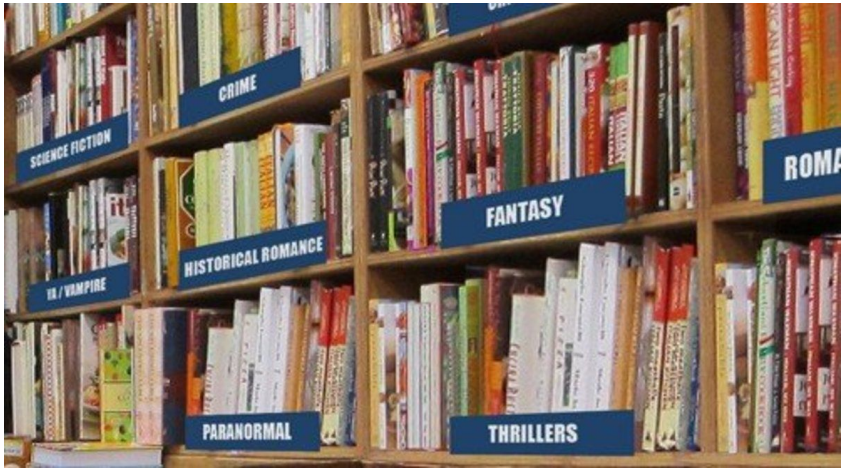
# Advanced Javascript

Day 1 - Js Data Types

Inside the computer’s world. There is only data.

What would be your first step to understand and work with such data without getting lost?

Categorization. Separating things into Types.



I have a question for you!

Programming language is the one who understands whether we want 2+5 or “two”+”five”.

Let’s understand what type of data Javascript understands and more fun stuff it does with the data.

There are 5 primitive data types in Js and 3 Non-primitive.

# What is primitive?

being the first or earliest of the kind or in existence

# Primitive DataType

In computer science, primitive data type is  a basic type provided by a programming language as a basic building block.

*Like a brick is to a building.*

**String**

As the name suggests, the string is for a sequence of characters, for example, “demo”, “Hi”, etc.

**Boolean**

It has two values i.e True or False.

**Number**

Number represents numerical values such as 5, 20, 500, etc.

**Undefined**

Undefined value

**Null**

Null value

# Numbers

Nothing fancy here. Numbers represent Numeric values.

# Numbers and Js

JavaScript has a built-in Math methods which contains a small library of mathematical functions.

**`Math.floor`**

Rounds down: `3.1` becomes `3`, and `-1.1` becomes `-2`.

**`Math.ceil`**

Rounds up: `3.1` becomes `4`, and `-1.1` becomes `-1`.

**`Math.round`**

Rounds to the nearest integer: `3.1` becomes `3`, `3.6` becomes `4`, the middle case: `3.5` rounds up to `4` too.

**`Math.trunc` (not supported by Internet Explorer)**

Removes anything after the decimal point without rounding: `3.1` becomes `3`, `-1.1` becomes `-1`.

`Math.random()`

`Math.max(a, b, c...)` / `Math.min(a, b, c...)`

`Math.pow(n, power)`

# String

In JavaScript, the textual data is stored as strings.

Strings can be enclosed within either ‘single quotes’, “double quotes” or `backticks`.

# Strings and Js

1. Finding length of a String
2. Accessing Characters (charAt)
3. Changing the case
4. Substring
5. Includes, startsWith, endsWith
6. split

For more - https://developer.mozilla.org/en-US/docs/Web/JavaScript/Reference/Global_Objects/String

# Boolean

Boolean has only two values in any programming language *True & False*.

It decides if the expression is true or false in *if-else*, *while*, *for-loop* blocks and according to that certain part of code gets executed.

# Null

*NULL* is a value.

You assign this value to the variable or object to wipe out all the data in it.

It’s just a special value which represents “nothing”, “empty” or “value unknown”.

# Undefined

The special value undefined also stands apart. It makes a type of its own, just like null.

The meaning of undefined is “value is not assigned”.

If a variable is declared, but not assigned, then its value is undefined:

# Non- Primitive DataTypes

**Object**

Collection of key value pairs.

**Array**

Storing multiple values as a long list.

**function**

Executes a block of code.

# By Value and By Reference

# Assignments in Js

Primitive Data Type  ( String, Number, Boolean etc)

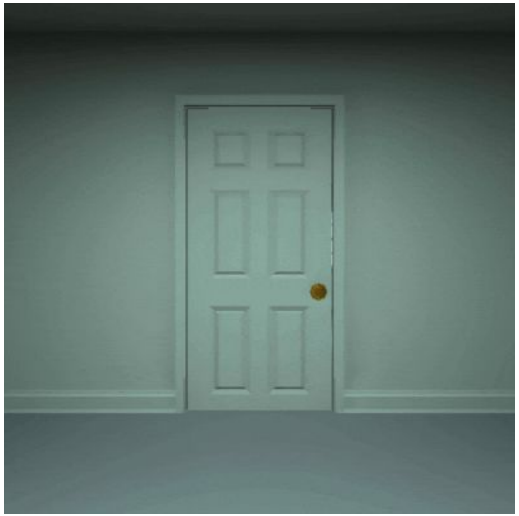
They hold the actual value.

Non-primitive Data type ( Objects, Array, functions )

They hold the location of value.

Consider assignments in JS to understand this mental model.

Suppose you have a multi-room house and multiple people living inside it.



There are TWO Primary ways those people would live there.

What if we have more people joining in?

It’s all about how Js handles the data it is has to store.

It’s called Memory Management.

In Js,

Primitive Datatypes: By Value

Non-Primitive Datatypes: By reference

Let’s see some examples...

# Reassignments

## By value

The original variable is not modified on changes in other variables.

## By reference

The original variable gets modified on changes in other variables.

# Data Mutability

# What is Mutability()?

To be able to change or modify something.

In Js,

Primitive Data Types : Immutable ( value cannot be changed in memory )

Non- Primitive Data Types: Mutable ( value can be changed in memory )

Remember:

Mutability refers to changing the value itself and not the value of variable.

Variable by its name and nature is supposed to be changed throughout a computer program.

What are the benefits of Immutability?

# Predictability

Predictable code is code where you can anticipate the outcome with a single read

# Tracking Mutations

Immutable data gives you a clear history of the program, because you're creating a new reference based on the source.

What are the drawbacks of Immutability?

Well, it depends....

Immutability isn’t free, every time we have to reallocate a new value, that's consuming CPU time and memory, including the garbage collection process for values that are no longer referenced.

But if you compare it with the fact that you have to write more code, create more copies and things can get really hard to know who changed who.

You will appreciate Immutability.

# Conclusion

If your app is small, doesn’t change a lot of values and have less reference to each other, you can avoid Immutability.

If your app is big, changes a lot of values and have complex references amongst them, Immutability can save your life!

# Let’s Recap

By Value - 

( a=b)

By Reference - a 

b

Mutability -

Immutability -