Cypress-JavaScript Knowledge Transformation

Javascript Intro - Basics to Know About

What is JavaScript?

JavaScript was initially created to “make web pages alive”.
Scripts (codes) can be written right in a web page’s HTML and run automatically as the page loads.

- Scripts are provided and executed as plain text. They don’t need special preparation or compilation to run.
- In this aspect, JavaScript is very different from another language called Java.

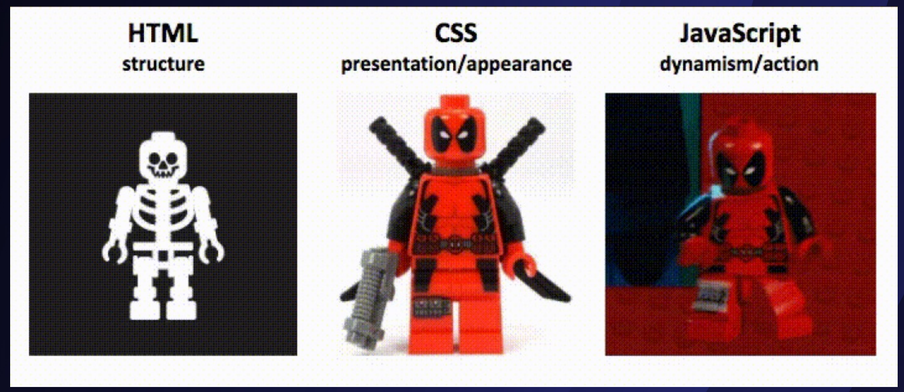
JavaScript Run Environments

Today, JavaScript can execute not only in the browser, but also on the server, or actually on any device that has a special program called the JavaScript engine.

- The browser has an embedded engine sometimes called a “JavaScript virtual machine”.
- Different engines have different “codenames”. For example: V8 – in Chrome, Opera and Edge.

The Role of JavaScript

JavaScript is one of the three core technologies of web development. For instance, in-browser JavaScript is able to:



- React to user actions, run on mouse clicks, pointer movements, key presses.
- Send requests over the network to remote servers, download and upload files.
- Get and set cookies, ask questions to the visitor, show messages.

Where do we put JS code?

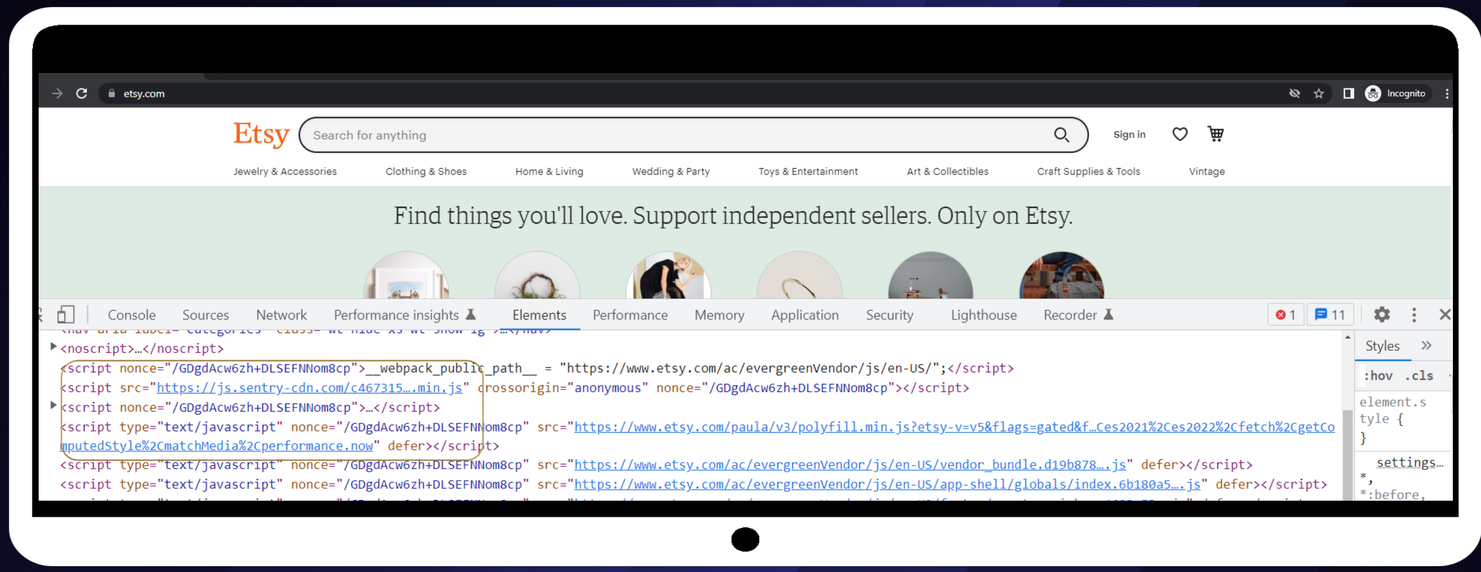
The <script> tag contains JavaScript code which is automatically executed when the browser processes the tag.

```
// Script files are attached to HTML with the src attribute


<script src="/path/to/script.js"></script>
```

How do we see JS code on a Browser Page?

Let’s learn how to open Developer console of Google Chrome: Press F12 or, if you’re on Mac, then Cmd+Opt+J.



Code Structure and Running Codes

Hello World!

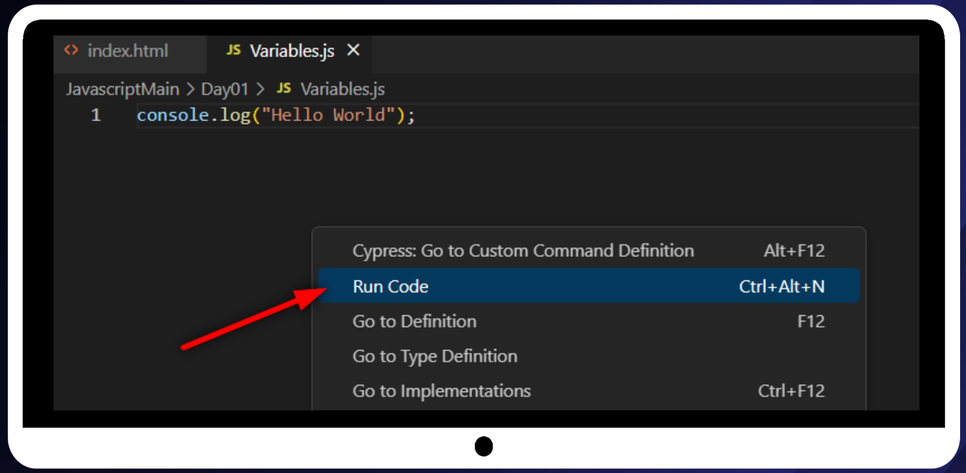
Let’s write our first JavaScript code:

Assuming we have installed VSC, Node.js, and extension (CodeRunner)

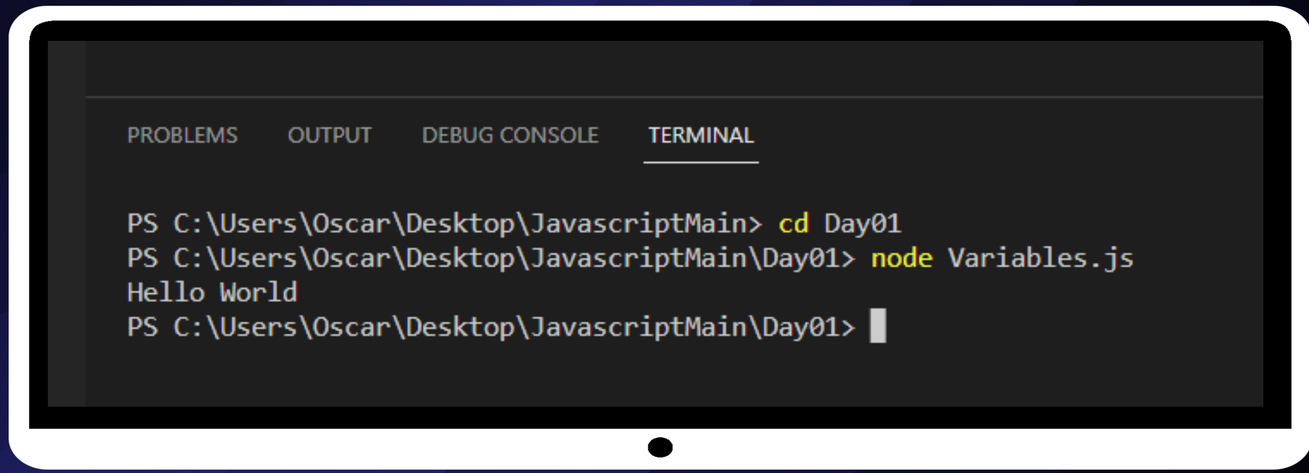
- Create a folder named JavaScriptProgramming on Desktop
- Open folder in VSC and Create Variables.js file

Hello World!

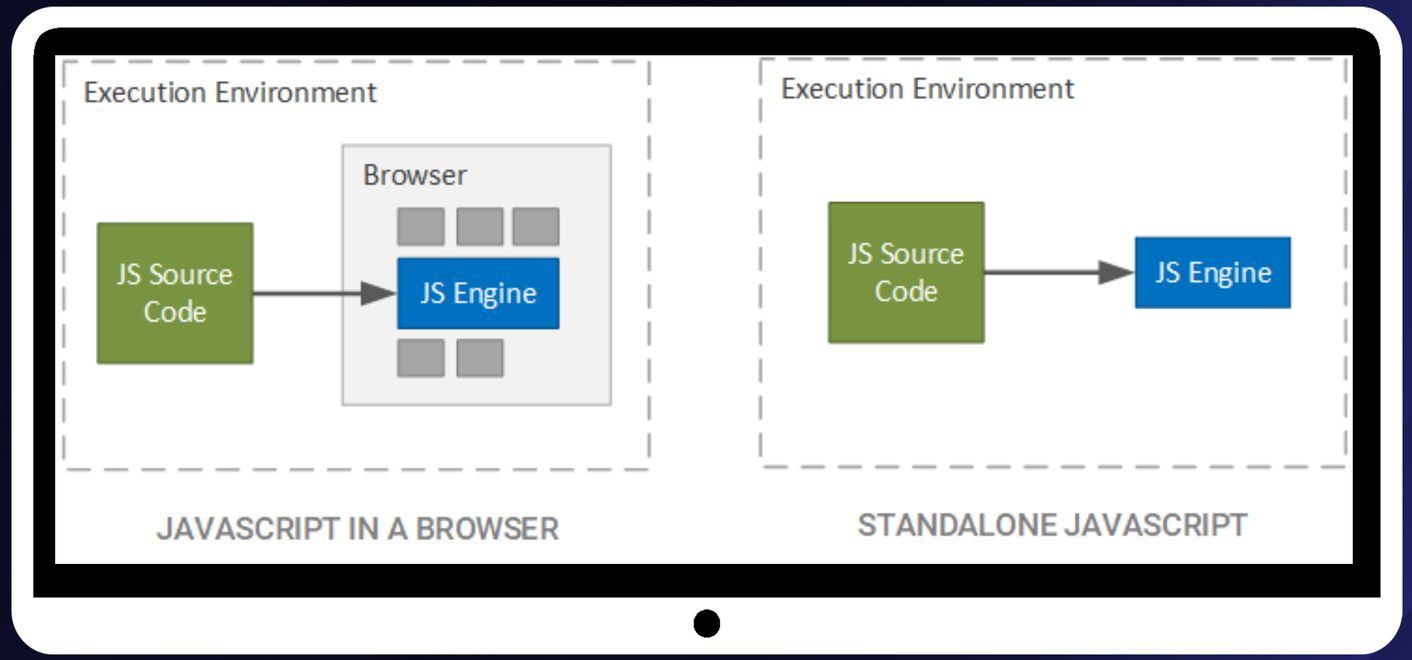
Run your codes with Node.js engine in your computer with Code Runner extension



Or running your code from Terminal



Javascript Virtual Machine Browser vs Node.js



HTML file needed: Runs on browser JS Virtual Machine

No HTML file: Runs on Node.js Machine

Code structure

Statements : are syntax constructs and commands that perform actions.

```
// A semicolon may be omitted in most cases when a line break exists.

console.log('Hello World');
```

Variables and Data Types in JS

Variables

Variable is a named container for a particular type of data used to hold a value.

```
// The statement below creates (in other words: declares) a variable with the name “message”:
   let message;
// store the string 'Hello' in the variable named message
   message = 'Hello';
```

- Storing data
- Manipulate Values
- Writing Flexible Code
- Enhance Code Readability

Task-1 Working with variables

Declare two variables: admin and firstName.

Assign the value "John" to firstName.

Copy the value from firstName to admin.

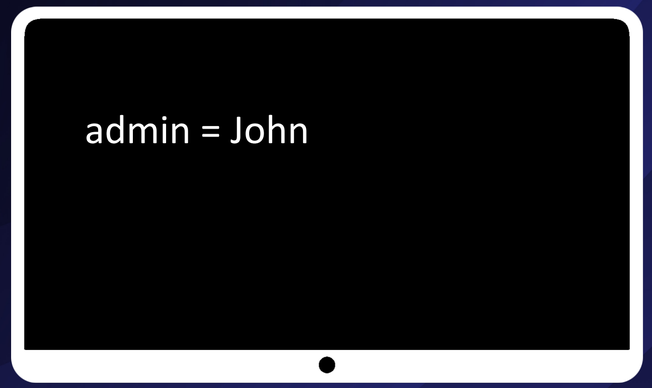
Show the value of admin using console.log()

Example:

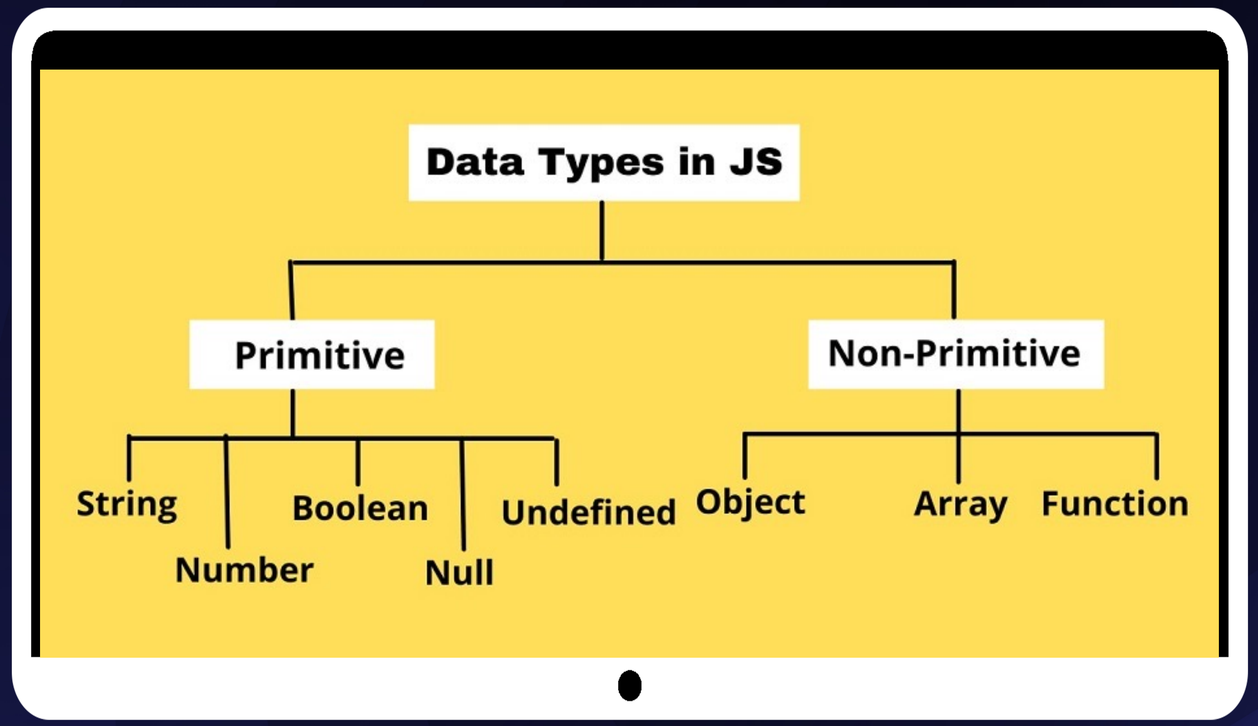
Input:

firstName = John

Output:



Data Types

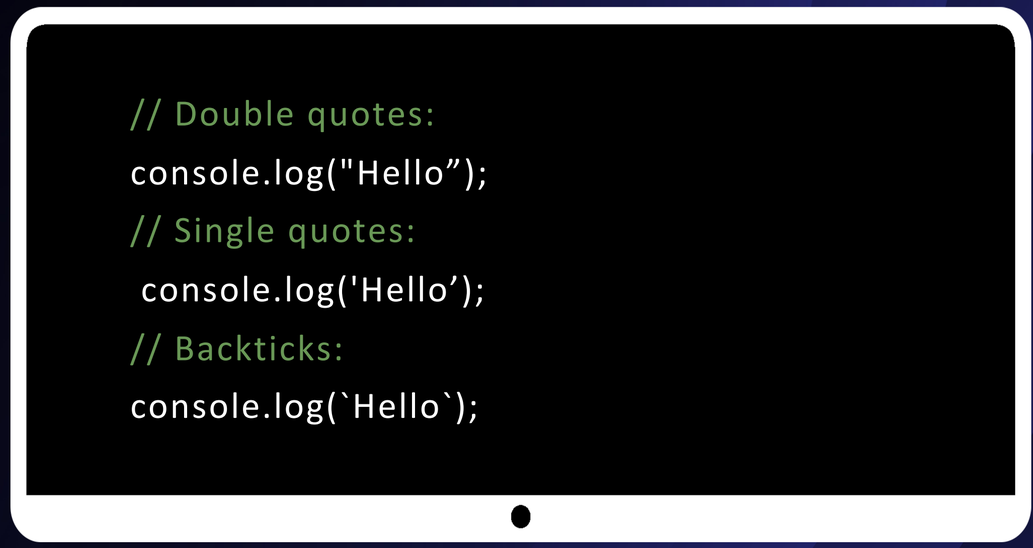


- Primitive: String, Boolean, Number
- Undefined and Null
- Object
- Array and Function

Strings in JS

Strings

In JavaScript, there are 3 types of quotes.



- Double and single quotes are “simple” quotes. There’s practically no difference between them in JavaScript.

- Backticks are “extended functionality” quotes. They allow us to embed variables and expressions into a string by wrapping them in ${…}

Operators in JS

Basic operators, maths

Math operations are supported in JavaScript

- Addition + and Subtraction -
- Multiplication * and Division /
- Remainder %
- Exponentiation **

Increment/decrement

Increasing or decreasing a number by one is among the most common numerical operations. So, there are special operators for it:

Increment ++ increases a variable by 1
Decrement -- decreases a variable by 1

The operators ++ and -- can be placed either before or after a variable.

When the operator goes after the variable, it is in “postfix form”: counter++.
The “prefix form” is when the operator goes before the variable: ++counter.

The prefix form returns the new value while the postfix form returns the old value (prior to increment/decrement).

Comparison Operators

Boolean is the result

All comparison operators return a boolean value

Description	Operator
Equality	==
Inequality	!=
Identity/Strict Equality	===
Non-identity/Strict Inequality	!==
Greater than	>
Greater than or equal	>=
Less than	<
Less than or equal	<=

- true – means “yes”, “correct” or “the truth”.
- false – means “no”, “wrong” or “not the truth”.

Logical operators

|| (OR)

The “OR” operator is represented with two vertical line symbols

&& (AND)

The AND operator is represented with two ampersands &&

Precedence of AND && is higher than OR ||
So the code a && b || c && d is essentially the same as if the && expressions were in parentheses: (a && b) || (c && d).

! (NOT)

The boolean NOT operator is represented with an exclamation sign !.

Conditional Branching in JS

Conditional branching: if, if else

Sometimes, we need to perform different actions based on different conditions. To do that, we can use the if statement

```
// if condition is true code block is executed

    if(score >= 50){
        congratulate();
}
```

“question mark” operator

The so-called “conditional” or “question mark” operator lets us do that in a shorter and simpler way.

The operator is represented by a question mark ?. Sometimes it’s called “ternary”, because the operator has three operands. It is actually the one and only operator in JavaScript which has that many.

The syntax is:

```
let result = condition ? value1 : value2;
```

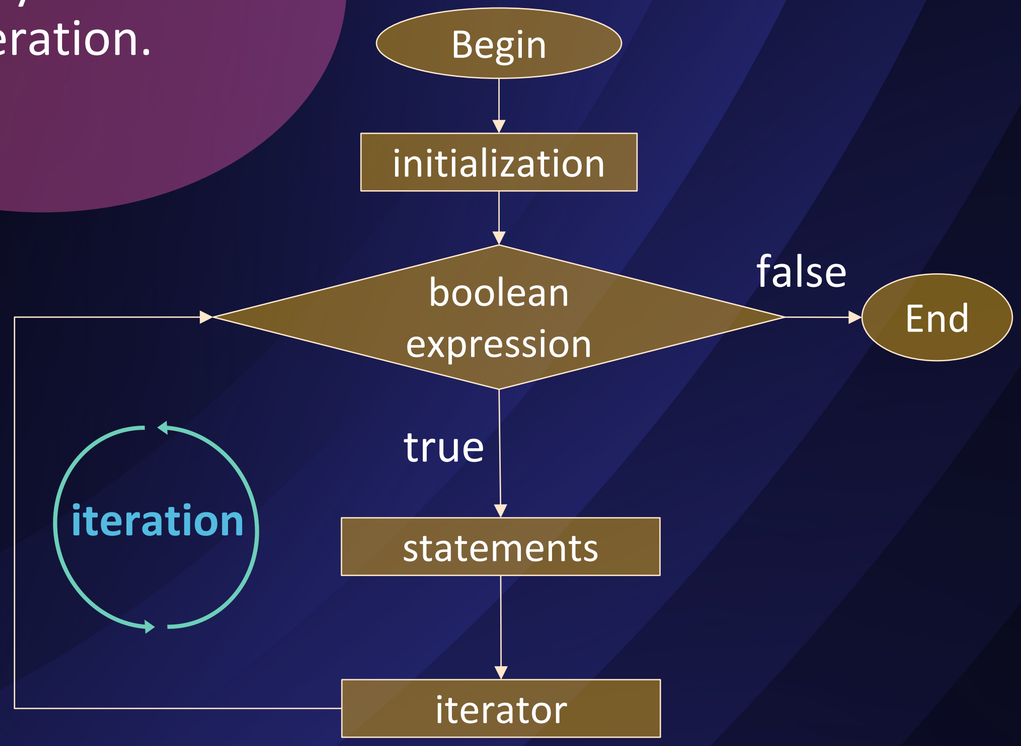
If condition is true, result is value1, if wrong result is value2

Loops in JS

Loops

Each execution of the loop body is called an iteration.

"for" loop allows repeated execution of code block based on a **certain number of iterations**.



Arrays and Data Structures in JS

Arrays and Data Structures

There is no Collection in JS. In JS we only have Arrays and Map

- Arrays are NOT Fixed size
- There are two important Brackets in JS
  1. [] ----> creates an Array
  2. {} ----> creates an object

Functions in JS

Functions

Functions are the main “building blocks” of JS. They allow the code to be called many times without repetition.

```
// To create a function we can use a function declaration.


    function showMessage() {

        console.log( 'Hello everyone!' );

}
```

Functions

They call functions as first class citizens of Javascript

* 1. you can pass a function as an argument to other functions
* 2. you can return a function from a function
* 3 . you can store a function into a varible

Functions and Callback Hell

Difference between Sync and Async Codes, How to handle async functions

* Sync Code

  --- Connect DB

  --- gets data

  --- is waiting until the data is received

  --- then executes rest of the code

* in JS – Async Code:

  --- while trying to connect DB

  *--- the rest of the functions does not wait

Functions and Callback Hell

WHAT THE HECK IS CALLBACK HELL?

```
3
4  a(function (resultsFromA) {
5      b(resultsFromA, function (resultsFromB) {
6          c(resultsFromB, function (resultsFromC) {
7              d(resultsFromC, function (resultsFromD) {
8                  e(resultsFromD, function (resultsFromE) {
9                      f(resultsFromE, function (resultsFromF) {
10                         console.log(resultsFromF);
11                     })
12                 })
13             })
14         })
15     })
16 });
17
```

Arrow Functions in JS

The basics of Arrow Functions

There’s another very simple and concise syntax for creating functions, that’s often better than Function Expressions.

```
let func = (arg1, arg2, ..., argN) => expression;
```

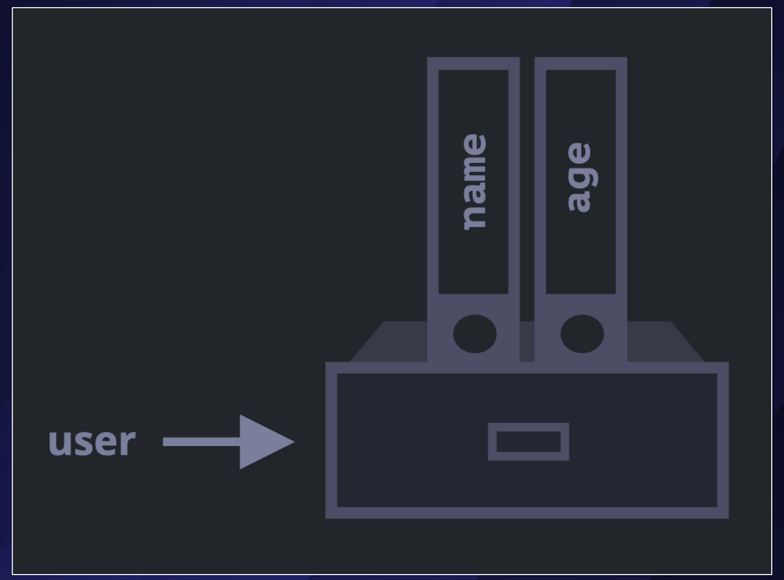
This creates a function func that accepts arguments arg1..argN, then evaluates the expression on the right side with their use and returns its result.

Objects in JS

Objects

An object can be created with figure brackets {…} with an optional list of properties.

A property is a “key: value” pair, where key is a string (also called a “property name”), and value can be anything.



Object Actions

We can add, remove and read files from it at any time.

Property values are accessible using the dot notation

- add new property to an object
- check a property if it exists
- put arrays, objects or functions inside objects
- loop using keys of the object

JavaScript Promise Object

Promise Objects in JS

A JavaScript Promise object contains both the producing code and calls to the consuming code.

A JavaScript Promise object can be:

- Pending
- Fulfilled
- Rejected

Promise Syntax

```
let myPromise = new Promise(function(myResolve, myReject) {
// "Producing Code" (May take some time)

  myResolve(); // when successful
  myReject();  // when error
});

// "Consuming Code" (Must wait for a fulfilled Promise)
myPromise.then(
  function(value) { /* code if successful */ },
  function(error) { /* code if some error */ }
);
```