

Programação de IAs Conversacionais

Módulo 1 - Revisão

Comando para criar variável

Podemos usar

let

var

const

Ou não usar nada e atribuir o valor diretamente

Operadores aritméticos

Soma	+
Subtração	-
Divisão	/
Multiplicação	*
Módulo	%
Incremento	++
Decremento	--

Diferença entre ++i e i++

Sabemos que os operadores ++ e -- incrementam e decrementam, ou seja adiciona +1 ou subtrai -1 do valor da variável.

A diferença da posição em que colocamos, se é antes da variável ou depois é a seguinte:

`++i` primeiro soma 1 ao valor de i e depois retorna i

`i++` primeiro retorna i e depois soma 1 ao valor de i

Diferença entre ++i e i++

Nesse exemplo usamos i++ veja que aparentemente não adiciona valor ao i pois primeiro o código imprimiu o valor de i e depois incrementou.

Executar { } ^ v 🔍 X 🗑️ 🔍 Filtrar saída		
1		Erros Warnings Logs Info
2	var i = 10	>> var i = 10
3		console.log(i++)
4	console.log(i++)	10
		← undefined

Diferença entre ++i e i++

Mas ao colocarmos a incrementação antes da variável imprime o que esperávamos, pois primeiro o código somou 1 ao valor da variável e depois imprimiu.

Executar { } ^ v 🔍 X		🗑️ Filtrar saída
1		Erros Warnings Logs
2	var i = 10	>> var i = 10
3		console.log(++i)
4	console.log(++i)	11
		← undefined

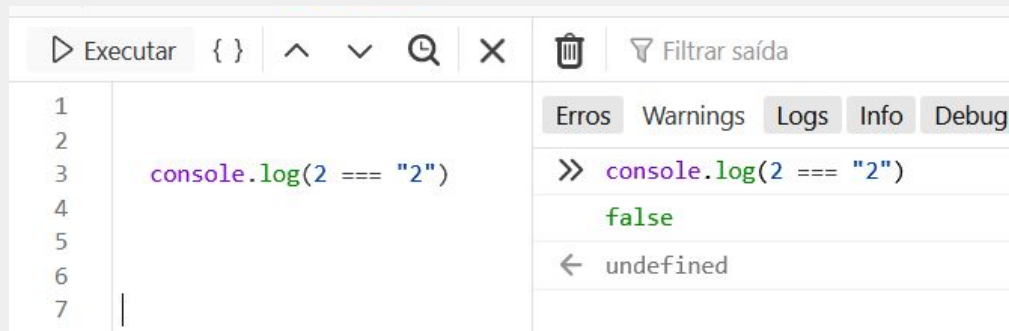
Operadores relacionais

Maior	>
Menor	<
Igual	=
Comparação	==
Igualdade estrita	===
Maior igual	>=
Menor igual	<=

Operadores relacionais

Igualdade estrita	===
-------------------	-----

A igualdade estrita compara tipo e valor!



The screenshot shows a code editor with a JavaScript file. Line 3 contains the code `console.log(2 === "2")`. The console output on the right shows the result of the execution: `>> console.log(2 === "2")` followed by `false` on the next line. The console also shows a `← undefined` message, which is likely the return value of the `console.log` function.

Veja que esse teste deu falso pois comparamos um number e uma string

Tipos de dados

String	“Qualquer coisa entre aspas duplas ou simples”
Number	Pode ser inteiro 235 Pode ser real 45.2
Boolean	true ou false
Object	Usamos as chaves { }, separamos atributo e valor com : , e separamos as propriedades com a vírgula { nome: “Aline”, idade: 35, genero “F” }

O tipo Array

Array é uma lista.

Pode ser uma lista de números, texto, uma lista de objetos... pode até ser tudo de uma vez

Podemos verificar qual é o tipo de dado usando uma função chamada `typeof()`

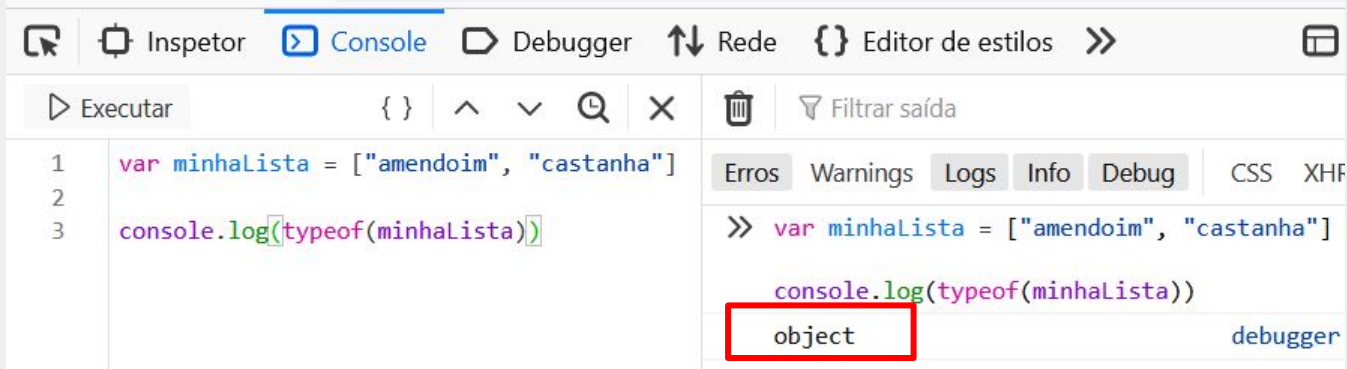
typeof()

Quando quisermos descobrir qual é o tipo de dado que há numa variável, usamos o `typeof(variavel)`

Dessa forma:

```
var minhaLista = ["amendoim", "castanha"]
```

Veja que guardamos em `minhaLista` um **array de strings**
`console.log(typeof(minhaLista))` // qual tipo é informado?



typeof()

Imprimiu na tela a palavra object !!!

Mas não é um array?

De acordo com o site

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Working_with_Objects

“Em JavaScript, quase tudo é um objeto...”

Tudo é objeto em JS?

A definição de objeto é algo que possui propriedades e tipos.

Outra citação do site <https://developer.mozilla.org/>:

Em JavaScript, um objeto é uma entidade independente, com propriedades e tipos.

Compare-o com uma xícara, por exemplo.

Uma xícara é um objeto, com propriedades. Uma xícara tem uma cor, uma forma, peso, um material de composição, etc. Da mesma forma, objetos em JavaScript podem ter propriedades, que definem suas características

Array vs Object

Um object é **uma coisa**

Ou seja, usamos **object com as { }** quando vamos criar **uma coisa só** que possui propriedades e atributos

Uma casa na cor azul, com 5 janelas, telhado vermelho, é **um** objeto:

```
var meuObjeto =  
{  
  nome: "casa1",  
  cor: "azul",  
  janela: 5,  
  corTelhado: "vermelho"  
}
```

Array vs Object

Um array é uma lista de várias coisas

Usamos **array com os []** quando vamos criar **uma lista de coisas** ordenadas, que podemos acessar pelo índice.

Uma **lista** de casas é um array:

```
var minhaLista =
```

```
[{  
  nome: "casa1",  
  cor: "azul",  
  janela: 5,  
  corTelhado: "vermelho"  
},  
{  
  nome: "casa2",  
  cor: "verde",  
  janela: 3,  
  corTelhado: "laranja"  
}]
```

IF ELSE

IF

(se) testa se uma condição é verdadeira

ELSE

(senão) faz o comando para o teste que não passou no IF

IF, ELSE IF

(senão se) Testa mais uma condição se não passou no primeiro IF antes de chegar no ELSE

Operadores lógicos

AND	&&
OR	
NOT	!
Comparação	==
Igualdade estrita	===
Maior igual	>=
Menor igual	<=

Métodos

<code>.indexOf()</code>	Retorna o índice no array daquilo que estamos procurando, pode ser uma palavra ou uma propriedade de um array de objetos
<code>.slice()</code>	Recebe o índice do início e fim, recorta o array nesses pontos
<code>.trim()</code>	Remove espaços antes e depois da frase
<code>.split()</code>	Recebe o dado para recortar a string
<code>.replace()</code>	Substitui uma parte da string por outra que passamos dentro dos parênteses

Métodos

Confira mais métodos em:

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/String

No menu à esquerda.

Funções

Sintaxe

```
function minhaFuncao (parametrosOpcionais){  
    //o que a função faz  
}
```

Usamos para dividir as soluções dos problemas e para organizar o código

Chamar uma função:

```
minhaFuncao(parametrosOpcionais)
```

Funções

Para saber mais sobre funções acesse a documentação disponível em:

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Functions>

Funções Nativas

<code>.typeof()</code>	Retorna o tipo de dado
<code>.pop()</code>	Remove o último elemento do array
<code>.map()</code>	Passa por cada elemento e realiza a ação que definirmos
<code>.filter()</code>	Passa por cada elemento e compara com algo, retorna as comparações corretas
<code>.forEach()</code>	Possui uma função nativa que separa os elementos do array em elemento, index e o array em si. Realiza a função que desejarmos
<code>.push()</code>	Insere um novo elemento ao fim do array

Funções Nativas

Math.min()	Retorna o menor valor dos que foram passados como parâmetro
Math.max()	Retorna o maior valor dos que foram passados como parâmetro
.toUpperCase()	Retorna a string com letras maiúsculas
.toLowerCase()	Retorna a string com letras minúsculas

Funções Nativas

Mais sobre Math.

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Math

Mais sobre métodos com arrays

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array