

JavaScript

Igor N Faustino





Bem vindos!

- Igor Neves Faustino (@igornfaustino)
- Formado em ciência da computação pela UTFPR
- Mestrado em Sistema de informação pelo IPB
- Atuando como desenvolvedor de softwares com mais de 5 anos de experiencia



Objetivos

- Entender o que é uma linguagem de programação
- Desenvolver lógica
- Entender como funciona a linguagem Javascript
 - Porque ela foi criada
 - Onde é utilizada
 - Qual a sintaxe básica
 - Quais são suas características



O que é Programação??

- Falar para o computador o que ele deve fazer para executar uma tarefa
- Usamos códigos para falar o que queremos que o computador faça (Algoritmos)
- Algoritmos são uma sequência de instruções que o computador deve seguir para realizar uma tarefa
 - Similar com uma receita de bolo
- As linguagens de programação são as linguagens utilizadas para escrever essas instruções.
- Diversas linguagens com características diferentes podem ser utilizadas
 - Javascript é uma dessas linguagens



Lógica

- Lógica de programação é o processo de pensar em soluções para problemas de forma estruturada.
- É a base para a escrita de algoritmos
- A lógica de programação pode ser aplicada sem conhecimento de linguagem
 - Permite com que programadores experientes troquem de linguagem de forma mais fácil



Estruturas Lógicas Básicas

- **Sequência:** Execução passo a passo de ações, como uma receita ou uma lista de tarefas.
 - **Exemplo:** Ações diárias: levantar, escovar os dentes, tomar café, ir ao trabalho.
- **Decisão (Condicional):** Realização de escolhas baseadas em condições.
 - **Exemplo:** Se estiver chovendo, leve um guarda-chuva; senão, vá sem ele.
- **Repetição (Loop):** Repetição de ações até que uma condição seja satisfeita.
 - **Exemplo:** Continue andando até chegar ao destino.



Exemplo: Achando o maior número

- Dado um conjunto de quatro números, como você determinaria qual deles é o maior?

Passos:

1. Compare o primeiro com o segundo e armazene o maior
2. Compare o atual maior com o terceiro e armazene o maior
3. Compare o atual maior com o quarto e armazene o maior
4. Exiba o maior!



Logicas condicionais

- **Condicional Simples:** "Se algo for verdadeiro, faça isso."
 - **Exemplo:** Se a nota do aluno for maior ou igual a 7, ele está aprovado.
- **Condicional Composta:** "Se algo for verdadeiro, faça isso; senão, faça aquilo."
 - **Exemplo:** Se a temperatura estiver acima de 30°C, ligue o ar-condicionado; senão, abra as janelas.
- **Condicional Aninhada:** Decisões dentro de decisões.
 - **Exemplo:** Se for feriado, veja se o tempo está bom; se sim, vá à praia, senão, fique em casa.



Exemplo: Verificação de Acesso

- Crie um algoritmo verbal que verifique se uma pessoa pode entrar em uma festa, considerando idade mínima (18 anos) e se o nome dela está na lista de convidados.

Passos

1. Verifique a idade
2. SE maior que 18 verifique a lista de convidados
3. SE dentro da lista permita a entrada
4. SENÃO recuse



Lógicas de repetições

- **Repetição Definida:** Executa um número fixo de vezes.
 - **Exemplo:** Conte de 1 a 10.
- **Repetição Indefinida:** Continua até que uma condição seja falsa.
 - **Exemplo:** Continue estudando até entender a matéria.



Exemplo: Contando vogais

- Dado uma palavra qualquer, crie um algoritmo verbal para contar quantas vogais (a, e, i, o, u) ela contém.

Passos:

1. Inicie a contagem em zero
2. Percorra cada letra da palavra
3. Verifique se a letra atual é uma vogal
4. Se for vogal some 1 na contagem
5. Repita esse procedimento para todas as letras
6. Exiba o valor de vogais 😊



Soma de Números Ímpares

- Como você somaria apenas os números ímpares de uma lista de números inteiros?
 - Pense em todos os passos para realizar essa operação



Exercitando a lógica

- [Problema da travessia no rio](#)
- [Torre de hanoi](#)
- [Teste de Einstein](#)



linguagens de programação

- Linguagem de maquina (Assembly)
 - Perto de como o computador entende
 - O programador precisa gerenciar todos os recursos
 - Muito complexo
- Linguagens modernas
 - Ex: Javascript, python, Java, C++, etc
 - Mais parecido com a nossa linguagem (ingles)
 - Mais facil de ler
 - Esconde muito da complexidade
 - Traduz para a linguagem que o computador entende



Como uma linguagem de programação funciona

- Instruções são traduzidas para a linguagem do computador
 - Compilado ou interpretado
- Normalmente instruções são executadas na ordem definida no código
- Existem diversos paradigmas de linguagens de programação
 - Estruturada
 - Orientação Objeto
 - Funcional
- Cada paradigma possui uma abordagem diferente para resolver problemas
- Linguagens evoluem com o tempo



Historia do javascript

- 1989 World Wide Web
- Navegadores começaram a ser desenvolvidos rapidamente
- 1995 Brendan Eich foi recrutado para criar uma linguagem de script para navegadores
 - Usou como base Java, Scheme e Self Language (que eram populares na época)
- Foi criado então a linguagem Mocha
 - Mas foi lançado com o nome de livescript pois achavam que mocha era um nome muito comum
- Eventualmente o nome mudou para Javascript
- 1997 Netscape junto com a ECMA criaram o padrão ECMAScript
 - Esse padrão é o que define quais funcionalidades a linguagem deve possuir



Historia do javascript

- Possuía vários problemas de performance no inicio
- Era desacreditado por vários programadores
- AJAX
 - **Asynchronous Javascript and XML** (Javascript e XML Assíncronos)
 - Técnica que permitia a criação de páginas mais dinâmicas
 - Permite atualizar o conteúdo sem atualizar a pagina
 - Ex: Gmail
 - Chamou a atenção para linguagem, que começou a ganhar cada vez mais suporte
- Novas versões foram lançadas



Historia do javascript

- EcmaScript 1 (1997)
 - Linguagem limitada
 - Presa no navegador
 - Não tinha intenção de ser uma linguagem de programação completa
- EcmaScript 2 (1998)
 - Pequenos ajustes
- EcmaScript 3 (1999)
 - Handler de exceções
 - Expressões regulares
 - Switch
 - Do-While



Historia do javascript

- EcmaScript 5 (2009)
 - JSON
 - API de array



Historia do javascript

- EcmaScript 6 (2015)
 - Class
 - Arrow Function
 - Proxy
 - Reflect
 - Map
 - Set
 - Destructing
 - Default Value
 - Template Literal
 - Spread Operator
 - Generators
 - Promises



Historia do javascript

- EcmaScript 7 (2016)
 - Array.prototype.include
- EcmaScript 8 (2017)
 - Async/Await
 - Object.* (values, entries)



Historia do javascript

- NodeJS
 - Criado em 2009 por Ryan Dahl
 - Permitia utilizar Javascript fora dos navegadores
 - Criada a partir do motor V8 presente no Google Chrome
 - Ganhou muita popularidade rapidamente, por permitir utilizar a mesma linguagem para desenvolver aplicações Fullstack
 - É bastante performático para o desenvolvimento de servidores devido a sua arquitetura não bloqueante



Características do javascript

- Linguagem de scripts
 - Comumente utilizada para automações e manipulações na execução de um programa
 - Alto nível, na qual se torna possível solucionar tarefas complexas com pouco código
- Tipagem dinâmica
- Assíncrona
- Precisa manter o suporte a versões antigas
 - Por ser a linguagem utilizada nos navegadores
- Single thread
 - Só processa uma coisa por vez
 - Delega tudo o que pode



Primeiros passos com javascript

<https://codesandbox.io/>



Hello World

- No Javascript podemos utilizar a função **console.log()** para exibir valores na tela
- Muito utilizado para investigar problemas



```
console.log("Hello World")
```



Expressões

- Também podemos utilizar expressões dentro de um **console.log**
- Todo trecho de código será primeiro processado, e apenas o seu resultado será exibido



```
console.log(2+2) // 4
```



Operações matemáticas

Operador	Descrição
+	Soma
-	Subtração
*	Multiplicação
/	Divisão
%	Resto (Modulo)
**	Expoente



Variáveis

- Valores devem poder ser utilizados em outros lugares
- Para isso podemos armazená-los nas chamadas variáveis
- Uma variável possui um nome para ser referenciada no código
- Uma mesma variável pode ser utilizada em N lugares



```
let a = 2
```

```
let b = 3
```

```
console.log(a+b) // 5
```



Como declarar variáveis

- Quando vamos utilizar uma variável pela primeira vez precisamos declarar ela
- Declarar uma variável significa pedir para o Javascript alocar um espaço na memória para armazenar nossos valores
- Quando tentamos utilizar uma variável sem declara-la, o Javascript vai estourar um erro. Já que não podemos acessar uma coisa que não existe



Como declarar variáveis

- Para declarar uma variável utilizamos a palavra **let**
 - Podemos também utilizar o **var**, mas não é mais comumente utilizado
- Um valor pode ou não ser passado na declaração de uma variável



```
let a // declarando sem atribuir um valor  
let b = 10 // declarando atribuindo um valor
```



Utilizando variáveis

- Como o nome sugere, o valor de uma variável pode ser alterado a qualquer momento
- Para isso basta atribuir o novo valor
- Não precisamos declarar novamente



```
let a = 10  
a = 9  
a = 8  
a = 7  
console.log(a) // 7
```



Exercício

1. Declare uma variáveis **x** que representa o lado de um quadrado
2. Atribua um valor qualquer para ela
3. Calcule a área do quadrado e armazene em uma nova variável chamada **area**
4. Exiba o valor de **area**



Exercício

1. Declare duas variáveis, **nota1** e **nota2**
2. Atribua um valor de 0 a 10 para cada uma
3. Calcule a média das duas notas e armazene em uma nova variável chamada **soma**
4. Exiba o valor de **soma**



Exercício

- Dada uma temperatura C em graus Celsius, escreva uma expressão para convertê-la para Fahrenheit.
- **Dica:** Use a fórmula $F = \frac{9}{5} * C + 32$



Exercício

- Declare duas variáveis `a` e `b`, atribua dois valores quaisquer e troque o conteúdo das duas variáveis

Ex:

- A: 1, B: 2 -> A: 2, B: 1



Constantes

- Funciona igual uma variável, porém não permite alterar seu valor após criada
- Ajuda a garantir que um valor não será alterado em nenhuma parte do código
 - Em alguns casos alterar um valor pode ser visto como uma má prática, e pode levar a erros inesperados
- Maior parte dos valores normalmente podem ser constantes
- O valor deve ser passado durante a declaração



Constantes

- Para declarar uma constante utilizamos a palavra **const**



```
const a = 10  
console.log(a) // 10
```



Tipos de dados

- Cada linguagem possui um conjunto diferente de tipos de dados
- Os tipos básicos podem ser utilizados para criar tipos complexos
- Os tipos básicos no Javascript são:
 - Numérico (number)
 - Texto (string)
 - Verdadeiro ou Falso (boolean)
 - Nulo (null)
 - Não definido (undefined)
 - Símbolos (symbols) // Não é muito utilizado
- No Javascript, esses tipos são atribuídos dinamicamente



Números (number)

- **Number:** Representa tanto números inteiros quanto de ponto flutuante.
 - números inteiros são os números positivos e negativos, que não apresentam parte decimal
 - Ponto flutuante são valores decimais
- Diferente de outras linguagens, o Javascript não faz diferenciação entre esses dois tipos



Números (number)

- Declarando um number



```
const nota1 = 10  
const pi = 3.14
```



Números (number)

- Apesar de todo número ser automaticamente considerado ponto flutuante (float) no Javascript, é possível obter apenas a parte inteira desse número
- Para isso podemos utilizar a função **parseInt**



```
const pi = 3.14  
  
const inteiro = parseInt(pi)  
console.log(inteiro) // 3
```



Exercício

1. Declare duas variáveis, uma com um número inteiro e outra com um número de ponto flutuante.
2. Some os valores e exiba o resultado.



Exercício

- Dado um valor decimal, remova a parte decimal mantendo apenas a parte fracionada

Ex: 2,50 -> 0,50



Textos (string)

- Representam dados textuais, delimitados por aspas simples ('), aspas duplas ("), ou crases (`).



```
let name = "Igor"
```



Texto (string)

- É possível juntar varias strings utilizando o operador +



```
let name = "Igor"  
let lastName = "Faustino"  
  
let fullName = name + " " + lastName  
console.log(fullName) // Igor Faustino
```



Texto (string)

- Podemos usar o crase (`) para combinar strings de uma maneira mais simples



```
let name = "Igor"
```

```
let saudacao = `Ola, ${name}`
```



Operações com strings

- **Comprimento:** `let comprimento = nome.length;`
- **Maiúsculas/Minúsculas:** `let maiusculo = nome.toUpperCase();`
- **Includes:** `let contemPalavra = nome.includes("João");`



```
let mensagem = "Olá, Mundo!";  
let grito = mensagem.toUpperCase(); // "OLÁ, MUNDO!"  
let comprimento = mensagem.length; // "11,"  
let includes = mensagem.includes("Mundo") // true
```



Exercício

- Declare uma variável chamada **nome** e atribua seu nome a ela.
- Exiba a mensagem: Olá, NOME!
- Modifique toda essa mensagem para letras maiúsculas



Exercício

- Dadas duas variáveis **str1** e **str2**, escreva um código que concatene essas duas strings.



Exercício

- Dada uma string **str** e uma substring **sub**, escreva um código que determine se **str** contém **sub**.
- Exemplo:
 - **str**: Ola mundo
 - **sub**: Ola
 - **Result**: true (verdadeiro)



Exercício

- Dada uma string `str`, escreva um código que determine o comprimento da string.
- Resultado esperado:
 - A string **`str`** tem **comprimento** letras



Exercício

- Declare duas variáveis `a` e `b`, atribua dois números quaisquer e troque o conteúdo das duas variáveis
- **SEM CRIAR UMA TERCEIRA VARIÁVEL**
- **Dica:** Utilize operadores matemáticos

Ex:

- A: 1, B: 2 -> A: 2, B: 1

