

# DESENVOLVIMENTO EM JAVASCRIPT

## Unidade 1

### Princípios Do Javascript

#### Aula 1

Introdução ao JavaScript

#### Introdução ao Javascript



##### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

#### Ponto de Partida

Bem-vindo, estudante! É com prazer que iniciamos, agora, nossa jornada pelo mundo da programação utilizando a linguagem Javascript, que é uma das linguagens mais importantes quando pensamos em aplicações voltadas para web. Você verá que o mercado de trabalho tem necessitado, cada vez mais, de mão de obra qualificada, e é aí que você entra. Com o Javascript você será capaz de desenvolver aplicações dinâmicas e interativas.

E por falar em programação, imagine-se na função de programador de uma empresa que precisa atender a um cliente que está com uma aplicação para seleção de estagiários apresentando um problema. Vale mencionarmos que esse processo de seleção possui duas etapas: a primeira se trata de uma prova online cujas regras exigem que os estudantes estejam cursando o terceiro período de um curso de nível superior e tenham rendimento de, no mínimo, 70%, para, então, serem selecionados para a segunda etapa, que diz respeito a uma entrevista presencial.

# DESENVOLVIMENTO EM JAVASCRIPT

No entanto, a equipe de recursos humanos percebeu que alguns estudantes que não tiveram qualificação foram selecionados pelo sistema para a prova presencial. Dois casos chamaram a atenção: (1) um aluno do 5º semestre acertou 50% de questões e foi aprovado; (2) já outro, que tinha 80%, cursava o 2º período, mas o sistema aprovou. E aí, tem alguma ideia de como resolver isso?

Vamos ver!

## Vamos Começar!

A linguagem Javascript é indispensável nos dias de hoje. Para a profissão de desenvolvimento web, é uma das principais linguagens que compõem um rol de tecnologias utilizadas no desenvolvimento de aplicações que funcionam nesse tipo de plataforma. Entre elas, estão: HTML e CSS, que funcionam com o Javascript.

Muitos estudantes e até mesmo profissionais da área de tecnologia costumam confundir o nome Javascript com Java. Não à toa, essa linguagem foi criada pela antiga empresa Netscape, logo que a web começou a ser desenvolvida, com o nome ECMAScript (MDN Web Docs, 2023). A linguagem Javascript é considerada de alto nível, dinâmica, interpretada e não tipada (Flanagan, 2013).

Diante disso, prepare-se, pois iniciaremos nosso percurso de estudos de desenvolvimento web!

## Como começar

Antes de tudo, para conseguir acompanhar as orientações desta e das próximas aulas, acesse a área de console do navegador web de sua escolha. Você precisa clicar com o botão direito dentro da página e escolher a opção “Inspecionar”, como mostra a figura a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

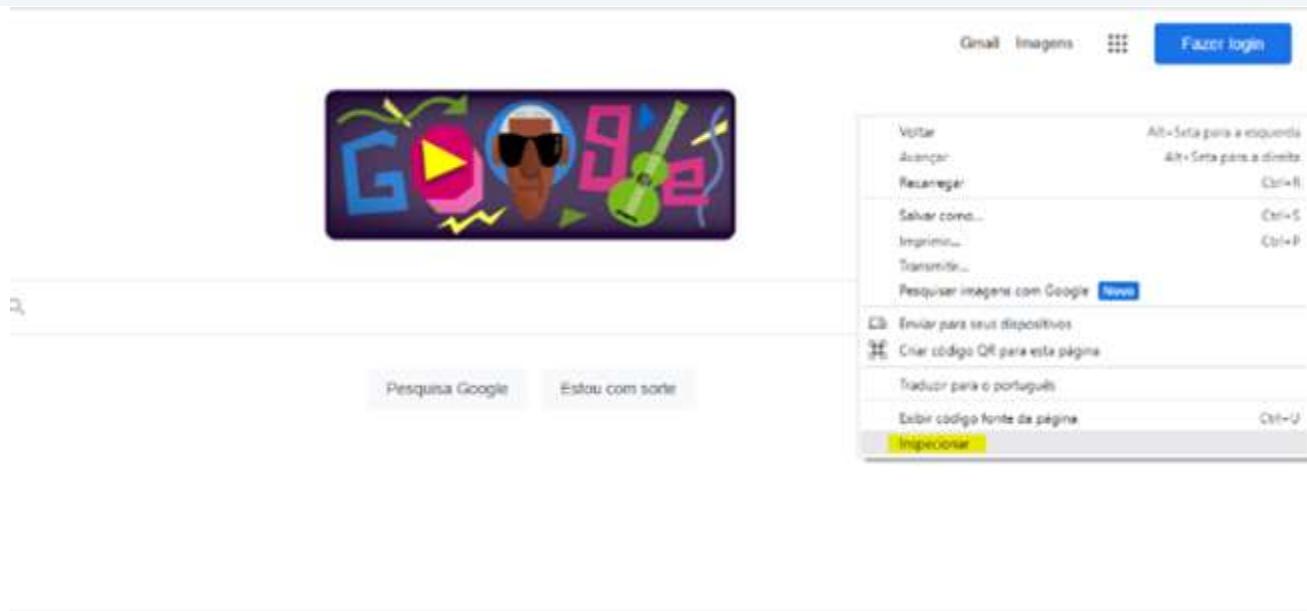


Figura 1 | Inspecionar página. Fonte: captura de tela do navegador elaborada pelo autor.

Em seguida, acesse a opção console; nela, você conseguirá executar os códigos simples que utilizaremos para esta parte introdutória da disciplina. Veja:

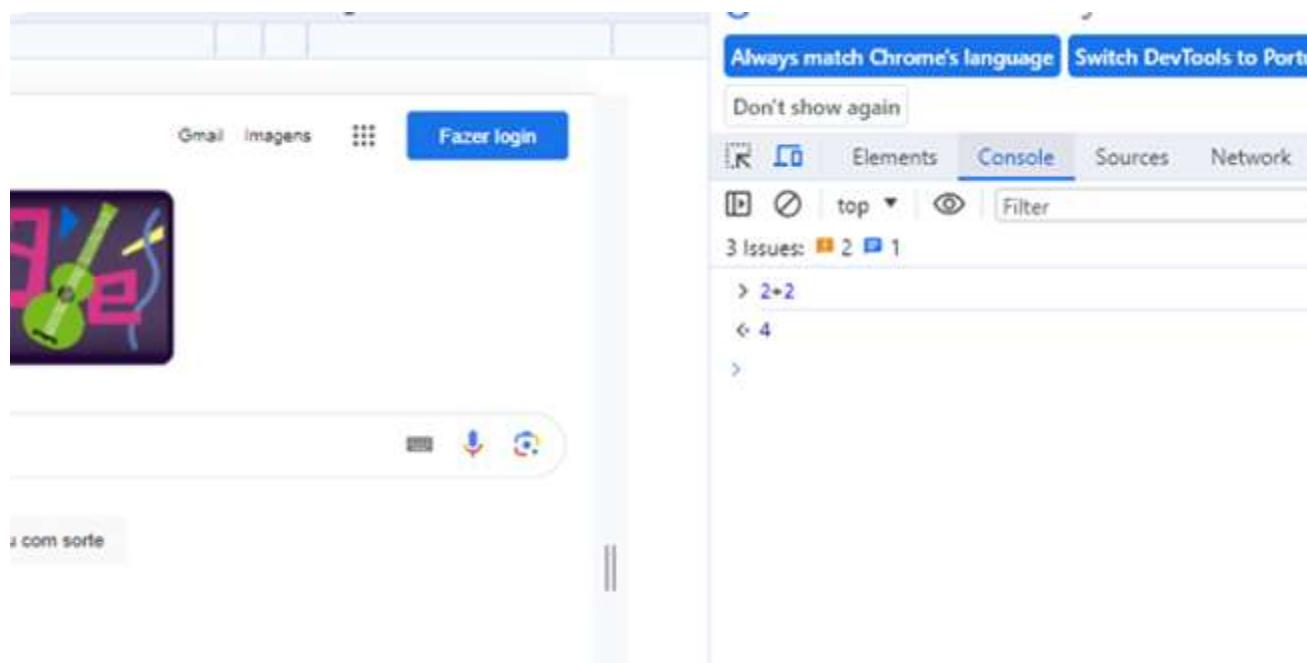


Figura 2 | Acesso ao console. Fonte: captura de tela do navegador elaborada pelo autor.

Em um projeto de página web, é possível criar um arquivo de Javascript com a extensão “.js”; ainda, é preciso ligar os arquivos de Javascript ao arquivo HTML de seu projeto, e um exemplo de código que cumpre essa tarefa é o seguinte:

# DESENVOLVIMENTO EM JAVASCRIPT

```
<script src=>
</script>.
```

A sintaxe da linguagem Javascript é simples, como você verá adiante

## Tipos de dados e variáveis

Para entender como são tratados os valores, as variáveis e seus tipos em Javascript, leia o bloco de código escrito nessa linguagem e que pode ser encontrando em Flanagan (2013), como mostra a figura a seguir:

```
// Tudo que vem após barras normais duplas é um comentário em linguagem natural.
// Leia os comentários atentamente: eles explicam o código JavaScript.

// variável é um nome simbólico para um valor.
// As variáveis são declaradas com a palavra-chave var:
var x;                                // Declara uma variável chamada x.

// Valores podem ser atribuídos às variáveis com o sinal =
x = 0;                                    // Agora a variável x tem o valor 0
x                           // => 0: Uma variável é avaliada com seu valor.

// JavaScript aceita vários tipos de valores
x = 1;                                    // Números.
x = 0.01;                                 // Apenas um tipo Number para inteiros e reais.
x = "hello world";                        // Strings de texto entre aspas.
x = 'JavaScript';                         // Apóstrofos também delimitam strings.
x = true;                                 // Valores booleanos.
x = false;                               // O outro valor booleano.
```

Figura 3 | Código explicando Javascript. Fonte: Flanagan (2013).

## Siga em Frente...

## Operadores aritméticos e relacionais

Para as operações matemáticas, vejamos alguns operadores que podem ser utilizados:

Operador	Descrição
+	Utilizado para efetuar soma.
-	Utilizado para efetuar subtração.

# DESENVOLVIMENTO EM JAVASCRIPT

*	Utilizado para efetuar multiplicação.
/	Utilizado para efetuar divisão.
**	Utilizado para efetuar exponenciação.
%	Utilizado para obter o resto de uma divisão.
=	Operador de atribuição.

Quadro 1 | Operadores aritméticos. Fonte: adaptado de Oliveira (2020).

Quanto aos operadores de relacionais, também chamados de operadores de comparação, temos alguns dos principais:

Operador	Descrição
==	Igual
====	Exatamente igual (conteúdo e tipo de dado)
!=	Diferente
<	Menor que
<=	Menor ou igual
>	Maior que
>=	Maior ou igual

Quadro 2 | Operadores relacionais. Fonte: adaptado de Oliveira (2020).

## Operadores lógicos

Os operadores lógicos são muito utilizados em estruturas de condição, mas não se preocupe, pois você conhecerá melhor esse tipo de estrutura.

Operador	Descrição
&&	E (AND)
	OU (OR)
!	NÃO (NOT)

Quadro 3 | Operadores lógicos. Fonte: adaptado de Oliveira (2020).

Veja o exemplo de código a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```

1 numero = 4;
2
3 //teste lógico
4 verificarA = (numero % 2 == 0); // True
5 verificarB = (numero % 2 == 1); // False
6
7 console.log(verificarA && verificarB);

```

Figura 4 | Uso de operadores lógicos. Fonte: elaborada pelo autor.

## Vamos Exercitar?

Vejamos, agora, o código que precisa ser corrigido. É importante lembrar que você precisa corrigir um erro no sistema de seleção de estagiários de seu cliente. Recorde que alguns estudantes que não tiveram qualificação foram selecionados pelo sistema para a prova presencial. Vamos resolver isso:

```

9 // determina aprovados para a segunda etapa
10 function primeiraEtapa(acertoProva, semCursado) {
11     const NumQuestoes = 20
12     const notaMinAprov = 0.7
13     const qtdeSemestres = 3
14
15     let nota = acertoProva / NumQuestoes // calcula % nota
16     console.log(nota);
17     if ((nota >= notaMinAprov) || (semCursado >= qtdeSemestres)) { // verifica se aprovado
18         return "Aprovado";
19     }
20     else {
21         return "Reprovado";
22     }
23 }
24 alert(primeiraEtapa(14, 2));

```

Figura 5 | Código a ser analisado. Fonte: elaborada pelo autor.

O código acima apresenta uma função que calcula se um estudante foi aprovado ou não para o processo seletivo de estágio, e para a solução desse problema, faz-se necessário o conhecimento de operadores lógicos e relacionais.

Vejamos as regras de aprovação:

# DESENVOLVIMENTO EM JAVASCRIPT

- A nota deve ser de, pelo menos, 70%, logo, a nota deve ser maior ou igual a 0.7.
- O aluno deve ter cursado, pelo menos, 3 semestres.

Convertendo essas regras para expressões, temos:

- nota  $\geq$  0.7.
- semestres\_cursados  $\geq$  3.

Agora, vamos analisar as linhas 17 a 22, em que temos as expressões que definem se o aluno foi aprovado ou não para a entrevista. O primeiro erro está no uso do operador  $\geq$  na linha 18. Ainda, na mesma linha, o operador  $\|$  está sendo usado incorretamente, pois deveria ser utilizado o  $\&\&$ , e esse fragmento de código deveria ficar assim:

```
if ((nota >= notaMinAprov) && (semCursado >= qtdeSemestres)) {  
    return "Aprovado";  
}  
else {  
    return "Reprovado";  
}
```

Figura 6 | Correção. Fonte: elaborada pelo autor.

Simples, não é?

## Saiba mais

No artigo, *The Top Programming Languages 2023*, você verá a ranking das principais linguagens de programação de 2023 elaborado pela organização IEEE.

CASS, S. [The Top Programming Languages 2023](#). **IEEE Spectrum**. 2023.

O MDN Web Docs, é uma fonte oficial mantida pela comunidade de desenvolvedores. Nela, você encontrará tudo sobre Javascript.

MDN WEB DOCS. [Guia de JavaScript](#). 2023.

Há, também, um artigo bem completo sobre vários recursos do Javascript, incluindo tudo que estudamos na aula de hoje, chamado *Aprendendo JavaScript*, de Filipe Del Nero Grillo e Renata Pontin de Mattos Fortes.

GRILLO, F. D. N.; FORTES, R. P. de M. [Aprendendo JavaScript](#). 2008.

# DESENVOLVIMENTO EM JAVASCRIPT

Por fim, acesse o material que fala sobre todos os operadores, expressões e keywords da linguagem Javascript.

MDN WEB DOCS. [Expressões e operadores](#). 2023.

## Referências

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Javascript**. 2022. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 10 jan. 2024.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Javascript descomplicado**: programação para web, IoT e dispositivos móveis. São Paulo: Érica, 2020.

## Aula 2

Estruturas condicionais

### Este conteúdo é um vídeo!



Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

Agora que você já compreendeu os conceitos básicos da linguagem Javascript, é hora de identificar conceitos de programação existentes em aplicações mais avançadas, compreendendo

# DESENVOLVIMENTO EM JAVASCRIPT

o funcionamento das estruturas condicionais. Esse tipo de estrutura é muito utilizado em diversas aplicações, pois se trata de uma das formas mais simples e eficazes de resolver questões de ordem lógica. O interessante é que nós utilizamos decisões baseadas em condições o tempo todo.

Ao longo desta aula, nós veremos a importância do domínio das estruturas condicionais para se atuar no mercado de trabalho de TI com programação, sendo crucial entender esse conceito para empregá-lo, quando for o caso. Aliás, nós utilizaremos estruturas condicionais para implementar melhorias em uma aplicação que você já conhece. Você se lembra daquela empresa na qual você hipoteticamente trabalha e que tem um sistema para seleção de estagiários? Pois é, você precisará alterar a aplicação a pedido da empresa, que, agora, deseja que o estudante entre para o banco de currículos cumprindo apenas a nota mínima de 70%. Se ele também estiver cursando ao menos o terceiro semestre, continuará sendo aprovado, caso contrário, reprovado.

Você imagina como pode resolver esse problema utilizando estruturas condicionais? Reflita um pouco sobre essa proposta e logo nos encontraremos para a resolução do caso.

Bons estudos!

## Vamos Começar!

A utilização de estruturas condicionais é indispensável na construção de códigos. Em Javascript e outras linguagens, é possível executar partes específicas de código, chamadas de blocos, que só serão executadas se algumas condições forem verdadeiras. Para o caso de a condição não ser verdadeira, o fluxo e execução serão alterados.

Nesta aula, você conhecerá a estrutura de repetição if, if..else e essa mesma estrutura organizada de forma encadeada, e isso significa que podemos escrever uma estrutura do tipo if...else dentro de outra. Confuso? Você verá que não. Venha conosco, pois lhe mostraremos como esse tipo de estrutura funciona.

## Estrutura condicional if

As estruturas condicionais ou estruturas para controle de fluxo do programa (Oliveira; Zanetti, 2020) servem para mexer no fluxo de execução de um código. O if, quando traduzido para o português, significa “se”, dessa forma, podemos determinar uma condição, em que, se verdadeira, o código do bloco será executado. Um bloco é considerado uma parte do código, que é delimitada por chaves { }. Veja um exemplo:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 CONDIÇÃO{  
2     Comandos de código;  
3 }
```

Figura 1 | Bloco condicional. Fonte: elaborada pelo autor.

A ideia por trás do funcionamento desses blocos é bem simples; o resultado da condição precisa ser verdadeiro para que os comandos dentro do bloco sejam executados, caso contrário, não serão.

Para tornar nosso trabalho interessante, vamos utilizar o ambiente de desenvolvimento VSCode. Veja:

```
1  <!DOCTYPE html>  
2  <html lang="pt-BR">  
3  <head>  
4      <meta charset="UTF-8">  
5      <title>Código javascript</title>  
6  </head>  
7  <body>  
8      <script>  
9          let nome = "Leonardo";  
10         if (nome) {  
11             console.log("O nome informado é "+ nome)  
12             alert("O nome informado é "+ nome)  
13         }  
14     </script>  
15  </body>  
16  </html>
```

Figura 2 | Exemplo de código com if. Fonte: elaborada pelo autor.

No bloco que está entre as linhas 10 e 13, verifica se há um nome dentro da variável; se houver, mostrará uma mensagem utilizando o alert, que é uma caixa de diálogo, e a mesma mensagem

# DESENVOLVIMENTO EM JAVASCRIPT

aparecerá no console do navegador, ficando assim:

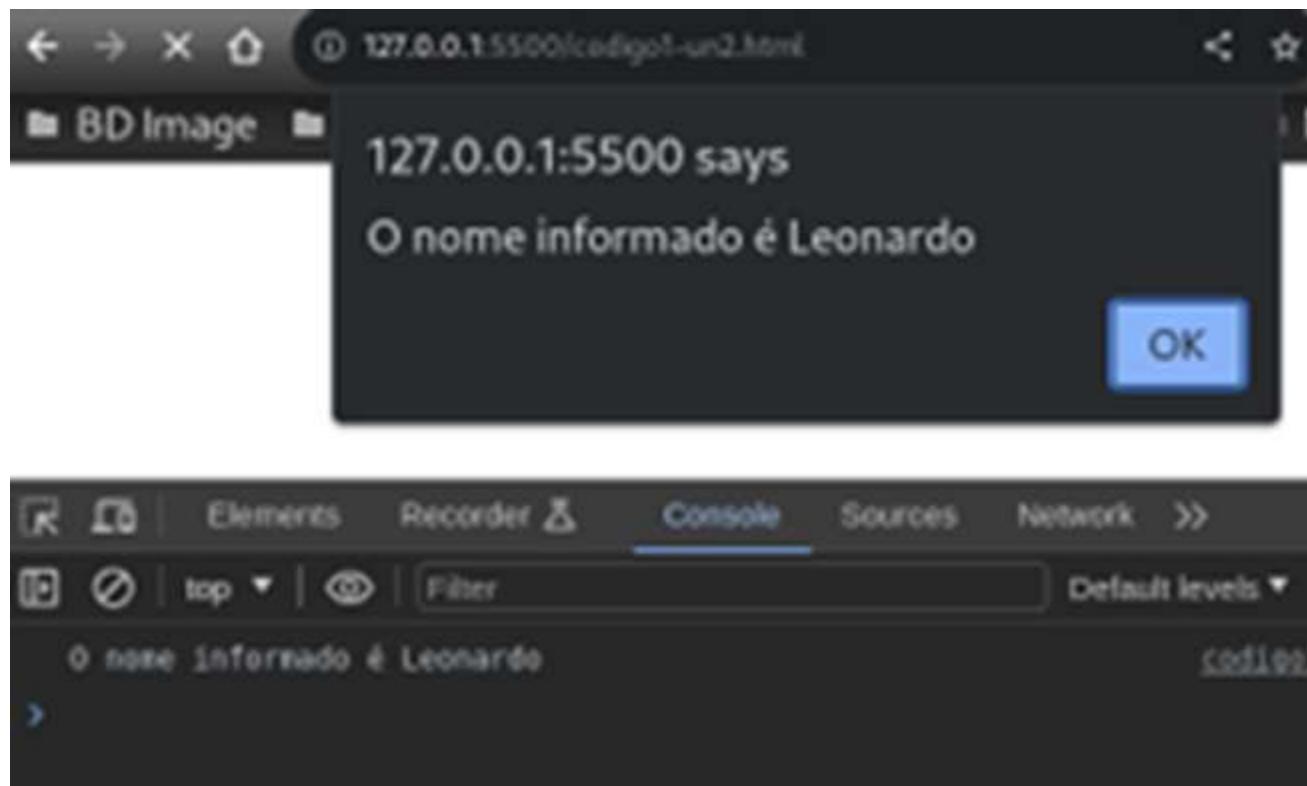


Figura 3 | Resultado do código. Fonte: elaborada pelo autor.

## Siga em Frente...

### Estrutura condicional if...else

A estrutura condicional permite adicionar uma execução à estrutura if; diante disso, já pensou oferecer uma alternativa para o caso de a condição que está sendo testada no If não ser falsa? É aí que entra o else. Ele será executado caso a condição em if for falsa e a condição else exista (MDN Web eDocs, 2023). Assim sendo, como, então, ficaria a utilização dessa estrutura se você levasse em consideração o exemplo de código anterior?

Tire o comando `console.log` e comente o código. Veja como ficará:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1  <!DOCTYPE html>
2  <html lang="pt-BR">
3  <head>
4      <meta charset="UTF-8">
5      <title>Código javascript</title>
6  </head>
7  <body>
8      <script>
9          let nome = "Leonardo"; //Nome guardado na variável
10         if (nome) { // se houver um nome na variável
11             alert("O nome informado é " + nome) // Mostra mensagem
12         }else{ // Se o if for falso
13             alert("Nenhum nome informado") //mostra essa mensagem
14         }
15     </script>
16 </body>
17 </html>
```

Figura 4 | Estrutura condicional if...else. Fonte: elaborada pelo autor.

## Estrutura condicional if...else aninhada

A estrutura condicional if...else pode, ainda, ser construída contendo, dentro de si, outra estrutura if...else. A depender do problema que se pretende resolver, esse formato de escrita fará muito sentido, e para que fique fácil a compreensão, imagine uma aplicação que faça a relação entre nota e conceito. Por exemplo, se um estudante tira nota entre 8 e 10, isso significa conceito A; já as notas entre 5 e 7,9, conceito B; e se ele não recebe nota, obtém, por sua vez, uma mensagem. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1  <!DOCTYPE html>
2  <html lang="pt-BR">
3  <head>
4      <meta charset="UTF-8">
5      <title>Nota x Conceito</title>
6  </head>
7  <body>
8      <script>
9          let nota;
10         let conceito;
11
12         if (nota) {
13             if ((nota <= 10) && (nota >= 8)) {
14                 alert('Conceito A')
15             }
16             if ((nota < 8) && (nota >= 5)) {
17                 alert('Conceito B')
18             }
19         } else {
20             alert('Você precisa se dedicar um pouco mais')
21         }
22     </script>
23  </body>
24  </html>
```

Figura 5 | if...else aninhado. Fonte: elaborada pelo autor.

## Vamos Exercitar?

Você viu como funciona a estrutura condicional if...else nesta aula e entendeu sua importância no contexto da programação, agora, vamos fazer uso dessa forma de escrita de código para resolver o problema apresentado no início desta aula.

Como deve se lembrar, a empresa solicitou uma alteração para armazenar em um banco de currículos os nomes dos estudantes que tiverem rendimento igual ou superior a 70%. Para resolver esse problema, vamos utilizar o else...if e incluir mais uma condição a ser verificada no código já atualizado anteriormente. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
if ((nota >= notaMinAprov) && (semCursado >= qtdeSemestres)) {
    return "Aprovado";
}
else if(nota >= notaMinAprov){
    return 'Você foi incluído no banco de currículos';
}else {
    return "Reprovado";
}
```

Figura 6 | Terceira condição com if...else. Fonte: elaborada pelo autor.

## Saiba mais

Agora que você já compreendeu a relevância do uso de estrutura condicional, vale a pena aprofundar seus conhecimentos, afinal, isso criará boas oportunidades para você no mercado de trabalho.

O livro ***Javascript Descomplicado: Programação Para Web, IoT e Dispositivos Móveis*** aborda, de um jeito simples e descomplicado, a utilização de código Javascript, indispensável na busca de uma boa colocação no mercado de trabalho.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. [\*\*JAVASCRIPT DESCOMPLICADO - PROGRAMAÇÃO PARA WEB, IOT E DISPOSITIVOS MÓVEIS\*\*](#). Editora Saraiva, 2020.

Leia, também, este excelente artigo que aborda a estrutura condicional de um jeito bem simples. O tema é apresentado de forma clara e objetiva, logo, sem dúvida, o ajudará a entender melhor a utilização de if...else.

DEVMEDIA. [\*\*JavaScript: Estrutura condicional if.\*\*](#) 2019.

Há, também, um artigo bem completo sobre vários recursos do Javascript, incluindo tudo que estudamos nesta aula; chama-se *Aprendendo JavaScript*, de Filipe Del Nero Grillo e Renata Pontin de Mattos Fortes, *Tópico 4.3 Estruturas de controle* (p. 13 e 14).

GRILLO, F. del N.; e FORTES, R. P. de M. [\*\*Aprendendo javascript\*\*](#). São Carlos: ICMC-USP. 2008.

## Referências

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

# DESENVOLVIMENTO EM JAVASCRIPT

MDN WEB DOCS. **Javascript**. 2024. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 10 jan. 2024.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Javascript descomplicado**: programação para web, IoT e dispositivos móveis. São Paulo: Érica, 2020.

## Aula 3

Estruturas de repetição

### Estruturas de repetição



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Partida

Nesta aula, você terá contato com outro tipo de estrutura muito comum na programação e que está presente em todas as linguagens de programação. Estamos falando da estrutura de repetição, que possibilita, por exemplo, que um dado bloco de código seja repetido quantas vezes forem necessárias.

A estrutura de repetição é muito útil quando se deseja trabalhar com listagem; no nosso caso, utilizaremos estrutura de repetição para permitir que a nossa cliente, a empresa com a qual estamos trabalhando aquele código para seleção de estágio, defina um número máximo de pessoas a ocupar as vagas de estágio definidas.

As estruturas de repetição, também conhecidas como estruturas de laços e iterações, oferecem um jeito fácil e rápido de executar uma ação repetidas vezes (MDN Web Docs, 2023). Na

# DESENVOLVIMENTO EM JAVASCRIPT

linguagem Javascript, existem, ao menos, oito formas de implementar esse tipo de estrutura, e nesta aula você conhecerá as mais comuns e utilizadas.

O domínio desse tipo de estrutura permitirá que você desenvolva ainda mais seu raciocínio lógico-matemático, habilidade indispensável para atuar no mercado de tecnologia.

Você se lembra da empresa na qual você hipoteticamente trabalha e tem um sistema para seleção de estagiários? Pois é, será necessário implementarmos uma nova regra à aplicação para seleção de estagiários, que, agora, será feita com, no máximo, dez estudantes. Para isso, empregaremos os conhecimentos desta aula e utilizaremos uma estrutura de repetição, para que, ao chegar no número de 10 pessoas, o sistema informe que a quantidade máxima limite foi alcançada e encerre a aplicação.

Vamos começar?

## Vamos Começar!

### Estrutura de repetição for

Esse tipo de estrutura permite a repetição de um determinado bloco de comandos enquanto uma condição especificada previamente seja verdadeira (Oliveira; Zanetti, 2021).

A sintaxe dessa estrutura é bem simples. Vamos entendê-la:

```
1 for(início; condição de parada; passo){  
2     // comandos que serão repetidos  
3 }
```

Figura 1 | Sintaxe laço For. Fonte: elaborada pelo autor.

Como você pode notar, dentro dos parênteses desse laço, existem 3 partes; a primeira cria a variável que será utilizada para a contagem de iterações, sendo necessário iniciá-la com algum valor; depois, na segunda parte, faz-se necessário definir a condição de parada; por fim, deve-se incrementar ou decrementar a variável criada na primeira parte.

Vejamos um breve exemplo de um simples contador:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 console.log("Contando... ");
2 for(var i = 0; i < 5; i++){
3     console.log(i);
4 }
```

Figura 2 | Exemplo de contador. Fonte: adaptada de Oliveira e Zanetti (2021).

Nesse código, na linha 2, foi criada a variável *i*, que recebe um valor zero no começo. A condição de parada definida é para a variável *i* com um valor menor que 5; nesse caso, quando *i* estiver com o valor 4, interromperá a execução do bloco. Ao final, foi feito incremento da variável *i* com o símbolo de adição duas vezes; dentro do bloco, será exibido um número sequencial no console do navegador. Perceba que tudo que será executado foi delimitado na estrutura de repetição por chaves {}, ou seja, tudo que será executado foi colocado dentro das chaves.

## Siga em Frente...

### Estrutura de repetição while

O comando while executa um bloco de operações até que a condição da instrução seja atendida (Zobot *et al.*, 2020). Veja como é a sintaxe dessa estrutura:

```
1 while(condição de parada){
2     comandos
3
4     incrementar variável de entrada
5 }
```

Figura 3 | Sintaxe da estrutura. Fonte: elaborada pelo autor.

Agora, montaremos um exemplo semelhante ao que criamos com laço For, só que com While. Vamos montar um contador para percorrer os mesmos números anteriormente. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 let i = 0
2 while(i < 5){
3     console.log(i);
4     i++;
5 }
```

Figura 4 | Contador com While. Fonte: elaborada pelo autor.

Assim como na estrutura do laço For, aqui, a condição de parada é determinada dentro dos parênteses, e os comandos a serem executados estarão dentro de chaves {} com o incremento da variável.

## Estrutura de repetição do...while

Esse tipo de estrutura é executado e só é encerrado quando a condição especificada for falsa. Aqui, a verificação da condição só acontecerá após a execução do bloco de programa a ser repetido (Oliveira, 2020), e a sintaxe para esse tipo de estrutura é definida da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 do
2   instrução
3 while(condição);
```

Figura 5 | Sintaxe do...while. Fonte: adaptada de Flanagan.

Agora, vamos fazer o mesmo contador feito anteriormente, só que com a estrutura de repetição apresentada aqui. Veja como ficaria:

```
1 let i=0;
2 do{
3   console.log(i);
4   i++;
5 }while(i < 5);
```

Figura 6 | Contador com do...while. Fonte: elaborada pelo autor.

Veja que, nessa estrutura, o conteúdo do bloco será, obrigatoriamente, executado ao menos uma vez, até que chegue a condição de parada, e só então ele poderá continuar ou parar definitivamente.

# DESENVOLVIMENTO EM JAVASCRIPT

## Vamos Exercitar?

Agora que você já conhece as principais estruturas de repetição, é chegada a hora de colocar em prática. Lembra do nosso caso, o qual é necessário implementar uma nova regra à aplicação para seleção de estagiários, que, agora, será feita com, no máximo, dez estudantes. Para isso, empregaremos os conhecimentos desta aula e utilizaremos a estrutura de repetição While, para que, ao chegar no número de 10 pessoas, o sistema informe que a quantidade máxima limite foi alcançada e encerre a aplicação.

O código será parcialmente similar ao que desenvolvemos na aula anterior, cabendo-nos, apenas, utilizar a estrutura de repetição mencionada. Veja como ficará nossa aplicação:

```
<script>
    /* se não for atribuída uma nota para a variável de mesmo nome, |
     * é porque o usuário cancelou a operação de inserção de dados. |
     * Neste caso, a estrutura de repetição deve ser encerrada. |
     */
    let nota, conceito;
    let count = 1;

    while (count <= 10) {
        nota = parseFloat(prompt("Informe a nota do estudante: "));
        if (nota) {
            if ((nota <= 10) && (nota >= 8)) {
                console.log('Conceito A');
            }
            if ((nota < 8) && (nota >= 5)) {
                console.log('Conceito B');
            }
            else {
                console.log('Você precisa se dedicar um pouco mais');
            }
            count++;
        }
    }
</script>
```

Figura 7 | Seleção de estágio para 10 pessoas. Fonte: elaborada pelo autor.

## Saiba mais

Faz-se importante conhecer bem as estruturas de repetição, por isso, é recomendado que você faça uma leitura dos seguintes artigos:

MDN WEB DOCS. [Laços e iterações](#). 2023.

# DESENVOLVIMENTO EM JAVASCRIPT

Complementando esse artigo, este apresenta o mesmo tipo de estrutura, com uma explicação clara e objetiva:

GARCEZ, L. P. <https://dev.to/acaverna/lacos-de-repeticao-em-javascript-50hj> Laços de repetição em Javascript. DEV. 2021.

Há, também, um artigo bem completo sobre vários recursos do Javascript, incluindo tudo que estudamos nesta aula; chama-se *Aprendendo JavaScript*, de Filipe Del Nero Grillo e Renata Pontin de Mattos Fortes, *Tópico 4.3 Estruturas de controle* (p. 15 e 16).

GRILLO, F. del N.; e FORTES, R. P. de M. [Aprendendo javascript](#). São Carlos: ICMC-USP. 2008.

## Referências

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Javascript**. 2024. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 10 jan. 2024.

OLIVEIRA, C. L. V.; ZANETTI, H. A. O. **Javascript descomplicado**: programação para web, IoT e dispositivos móveis. São Paulo: Érica, 2020.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Node.js**: programe de forma rápida e prática. São Paulo: Editora Saraiva, 2021. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9786558110217/>. Acesso em: 10 jan. 2024.

ZABOT, D.; MATOS, E. de S. **Aplicativos com bootstrap e angular**: como desenvolver apps responsivos. São Paulo: Érica, 2020. Disponível em: <https://integrada.minhabiblioteca.com.br/#/books/9788536533049/>. Acesso em: 10 jan. 2024.

## Aula 4

Estrutura de dados

### Estrutura de dados

Este conteúdo é um vídeo!

# DESENVOLVIMENTO EM JAVASCRIPT



Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

Neste momento, iniciamos o estudo de estrutura de dados, além disso, você terá contato com conceitos de funções e outro tipo de dado que é fundamental na linguagem Javascript, pois permite armazenar coleções de chaves e valores. Estamos falando dos objetos.

Para nortear nossos estudos nesta aula, temos um novo desafio. A empresa para a qual você está escrevendo código, agora, precisa de uma aplicação que realize o agendamento para as entrevistas com as pessoas selecionadas para as vagas de estágio. A empresa quer que esse agendamento seja feito de forma a ordenar alfabeticamente as pessoas e, ainda, ofereça dois horários disponíveis, para o caso de ser necessário realizar algum reagendamento de forma manual. Diante disso, fique atento à aula, pois você verá tudo que será necessário para resolver esse desafio.

E aí, o que está achando? Você deve ter notado que os conteúdos começam a ficar um pouco mais complexos, não é isso? Tenho certeza que você dará conta do que estamos propondo para esta aula. Vamos começar?

## Vamos Começar!

## Estrutura de dados

As estruturas de dados na programação são essenciais em muitos sentidos; elas favorecem o trabalho de qualquer pessoa que trabalhe com programação. Em Javascript, as estruturas de dados mais conhecida é o array, que é considerado objeto de alto nível, semelhante a listas (MDN, 2023).

### Array

No array, elementos do tipo texto devem ser escritos entre aspas e, se houver mais de um, separados por vírgula. Veja um exemplo:

# DESENVOLVIMENTO EM JAVASCRIPT

```
> let listaCompras=['leite', 'maçã', 'iogurte'];
< undefined
> listaCompras
< (3) ['leite', 'maçã', 'iogurte']
```

Figura 1 | Lista de compras com array. Fonte: elaborada pelo autor.

Um array é organizado por posições que são acessadas por número que iniciam em zero. Por exemplo, na imagem acima, o elemento “leite” está na posição zero do array, já “maçã” está na posição um e assim por diante. Essas posições são chamadas de índices (Oliveira; Zanetti, 2020).

Existem algumas propriedades e métodos que podem ser utilizados para manipular um array. Vejamos:

- **length**: propriedade utilizada para identificar o número de elementos de um array (MDN, 2023).
- **splice()**: esse método retorna uma cópia de parte de um array criando um subarray, que é definido a partir da informação de uma posição inicial e final (MDN, 2023).
- **push()**: Método que serve para adicionar elementos ao final de um array, com retorno do novo comprimento desse array (MDN, 2023).
- **pop()**: esse método remove o último elemento de um array (MDN, 2023).
- **shift()**: para apagar o primeiro elemento, basta utilizar esse método (MDN, 2023).

**Siga em Frente...**

## Funções

As funções são consideradas blocos de códigos que são executados quando um script é interpretado. Para declarar uma função, é necessário fazer uso da palavra-chave `function`, e ela pode ter ou não argumentos (Grillo; Fortes, 2008). Você entenderá a sintaxe de uma função com o exemplo a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```

1 function NOME-DA-FUNÇÃO(parâmetros de entrada){
2     comandos
3 }
4 // chamada da função
5 NOME-DA-FUNÇÃO(ARGUMENTOS DE ENTRADA)

```

Figura 2 | Sintaxe de uma function. Fonte: elaborada pelo autor.

Veja o exemplo de uma função para somar dois números que serão informados pelo usuário da aplicação:

```

1 function soma(n1, n2){
2     return n1 + n2;
3 }
4 let valor1 = parseInt(prompt('Informe o primeiro número'))
5 let valor2 = parseInt(prompt('Informe o segundo número'))
6 let resultado = soma(valor1, valor2);
7 alert('O resultado da soma é ' + resultado)

```

Figura 3: Função para soma. Fonte: elaborada pelo autor.

Se utilizássemos uma maneira mais elegante para escrita de função, a Arrow Function, a mesma função escrita anteriormente, ficaria desta maneira:

```
soma = (n1, n2) => n1 + n2
```

Percebeu como esse tipo de escrita deixa muito mais enxuto, limpo e elegante o código de uma função?

## Objetos

Objeto pode conter diversos valores de tipos diferentes armazenados nele, bem como possui funções que operam sobre esses valores (Grillo; Fortes, 2008). Esses objetos podem ser criados de duas maneiras, veja:

- Nesse exemplo, criamos um objeto genérico

```
var objeto = new Object();
```

# DESENVOLVIMENTO EM JAVASCRIPT

- Neste outro, criamos um objeto literal

```
var objeto = {atributo1: 'valor1', atributo2: 'valor2'}
```

Note que a estrutura dessa segunda forma de declaração assemelha-se muito à estrutura de chave e valor.

## Vamos Exercitar?

E então, o que você achou dos conteúdos que estudamos na aula de hoje? Interessante compreender como funciona um Array, funções e objetos, não é mesmo?

Agora que você compreendeu bem esses novos conceitos, é hora de colocar a mão na massa e realizar a criação de uma aplicação de agendamento para as pessoas que foram selecionadas para o estágio de sua empresa cliente. Lembre-se de que a empresa lhe pediu um agendamento por ordem crescente de nomes e com dois horários vagos para possíveis reagendamentos manuais. A solução para esse problema será como o código apresentado a seguir:

```

10 <script>
11     var hora = 8
12     var minutos = 20
13     var total_entrevistas = 0
14     const saida = 11
15
16     var entrevistados = [
17         "João Mariano",
18         "Adélia de Souza",
19         "Fábio Almeida",
20         "Carla Silva",
21         "Paulo Arruda",
22         "Leonardo Rocha",
23         "Tiago de Lima",
24         "Patrícia de Lima",
25         "Fernanda Brito",
26         "Maria da Conceição",
27     ]
28
29     entrevistados.sort(); // ordenando os entrevistados alfabeticamente
30

```

Figura 4 | Resolução com array – parte 1. Fonte: elaborada pelo autor.

Nessa primeira parte, cria-se as variáveis necessárias para se definir os horários e o array com os nomes selecionados, bem como é feita a ordenação desses nomes.

# DESENVOLVIMENTO EM JAVASCRIPT

```

30     for (i = hora; i <= saida; i = i + 1) {
31         if ((i == 12) || (i == 13)) {
32             continue
33         }
34         for (j = 0; j < 60; j = j + minutos) {
35             total_entrevistas++;
36             if (j == 0) {
37                 console.log(i + ":" + j + "0", "", entrevistados[total_entrevistas - 1])
38             }
39             else {
40                 console.log(i + ":" + j, "", entrevistados[total_entrevistas - 1])
41             }
42         }
43     }
44   
```

</script>

Figura 5 | Resolução com array – parte 2. Fonte: elaborada pelo autor.

## Saiba mais

Para um melhor entendimento de todos os métodos e propriedades que existem para manipulação de arrays em Javascript, faz-se importante a leitura do artigo:

MDN WEB DOCS. [Array](#). 2024.

Há, também, um artigo bem completo sobre vários recursos do Javascript, incluindo tudo que estudamos nesta aula; chama-se *Aprendendo JavaScript*, de Filipe Del Nero Grillo e Renata Pontin de Mattos Fortes, *Tópico 4.3 Estruturas de controle (p. 16 - 19)*.

GRILLO, F. del N.; e FORTES, R. P. de M. [Aprendendo javascript](#). São Carlos: ICMC-USP. 2008.

## Referências

GRILLO, F. del N.; FORTES, R. P. de M. [Aprendendo JavaScript](#). São Carlos: ICMC-USP, 2008.

MDN WEB DOCS. [Javascript](#). 2024. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global\\_Objects/Array](https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Reference/Global_Objects/Array). Acesso em: 11 jan. 2024.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Javascript descomplicado**: programação para web, IoT e dispositivos móveis. São Paulo: Érica, 2020.

## Aula 5

Encerramento da Unidade

# DESENVOLVIMENTO EM JAVASCRIPT

## Videoaula de Encerramento



### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Chegada

Ao longo de nossas aulas, vimos os fundamentos iniciais da linguagem Javascript. Iniciamos nossos estudos abordando os tipos de dados e variáveis existentes na linguagem, compreendemos a importância dos tipos de operadores que podemos utilizar e vimos como utilizá-los juntamente com estruturas condicionais. Esse tipo de estrutura está presente em várias linguagens e é muito utilizado na construção de código aplicado às mais variadas situações, pois com esse tipo de estrutura resolvemos vários tipos de problema.

As estruturas de repetição, por sua vez, permitem a execução repetidas vezes de um determinado bloco de código, até que uma determinada condição seja atendida, mas vale lembrar que os vários tipos de repetição mudam a forma como o bloco é executado, logo, fique atento às nuances desses tipos de estrutura. Por fim, vimos vários tipos de estrutura de dados e, entre eles, exploramos conceitos de arrays e como esse tipo de estrutura de dados pode, facilmente, substituir um banco de dados, caso o volume de dados seja relativamente pequeno.

Ainda, vimos também como construir uma função e vimos qual é a finalidade desse tipo de código. Quando temos uma tarefa específica que precisa ser executada várias vezes, faz muito sentido montar a solução para essa tarefa dentro de uma função, pois elas nos auxiliam na execução de blocos de código, e o mais legal é que, agora, você sabe como é possível escrever uma função de forma eficiente e elegante. Sabe de qual tipo de estrutura de função estamos falando, não é? Claro, as arrow functions! Por fim, outra estrutura de dados, os objetos, também têm muitas utilidades e pode ser crucial na implementação de uma dada aplicação para resolução de um problema.

Com os conhecimentos desta unidade você será capaz de elaborar e analisar estruturas fundamentais do desenvolvimento utilizando Javascript.

# DESENVOLVIMENTO EM JAVASCRIPT

## É Hora de Praticar!



### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

A utilização de arrays se assemelha muito ao que uma aplicação que utiliza banco de dados faz: armazenar um determinado dado. Como visto nas aulas, os fundamentos do Javascript nos possibilita construir uma aplicação mais robusta, tendo em vista que compreendemos os principais recursos dessa linguagem até o momento.

Agora, para colocarmos em prática tudo o que aprendemos, o seu desafio consiste em desenvolver uma pequena aplicação que servirá para criar uma lista de cadastro de usuários, algo presente em todos os sistemas de e-commerce existentes atualmente. Alguns exemplos são: Mercado Livre, Shopee, Shein, Amazon e outros.

Esse cadastro servirá para as pessoas que estão sendo alocadas nas vagas de estágio, situação com a qual trabalhamos ao longo de todas as aulas. Para lidar com esse desafio, você poderá utilizar uma estrutura de dados em Javascript que aprendemos na última aula. Que tal utilizar a estrutura de array para esse desafio?

Sua aplicação deverá receber o nome informado do candidato e armazená-lo no array, além de configurar a possibilidade de edição e exclusão dessa lista. Se você, assim como eu, sente-se instigado por desafios, vai se sentir muito bem com este.

E então, aceita o desafio? Vamos lá!

- Os tipos de dados e variáveis em Javascript são semelhantes a outro tipo de linguagem que você conhece?
- Você concorda que estamos, a todo instante, tomando decisões com base em lógicas de raciocínio semelhantes àquelas das estruturas de decisão?
- Ficou claro como os elementos de um array são acessados via código?
- As operações possíveis são variadas, você não acha?

A aplicação consiste na criação de uma tela de cadastro simples, somente para nomes. A ideia é armazenar esses nomes em um array. Como a tarefa consiste em permitir a edição e exclusão dos nomes, será necessário criarmos um formulário em HTML. Para isso, vamos criar um formulário simples, como no código a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1  <!DOCTYPE html>
2  <html>
3
4  <head>
5      <script src="https://ajax.aspnetcdn.com/ajax/jquery/jquery-3.3.1.min.js"></script>
6      <script src="controller.js"></script>
7  </head>
8
9  <body>
10     <div>
11         <label id="nome">Nome:</label>
12         <input id="nomeUser" type="text">
13         <button type="button" onclick="salvarUser()">Salvar</button>
14     </div>
15
16     <div>
17         <table id="tabela">
18             <tr>
19                 <th>Nome Usuario</th>
20                 <th>Ações</th>
21             </tr>
22             </table>
23     </div>
24 </body>
25
26 </html>
```

Figura 1 | Código html. Fonte: elaborada pelo autor.

O resultado será algo como o mostrado na imagem a seguir:]

Nome:

Nome Usuario Ações

Figura 2 | Resultado da página. Fonte:  
elaborada pelo autor.

O Javascript para essa página é algo que envolverá variáveis, estruturas condicionais, estruturas de repetição, array e função. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```

13 /*FUNÇÃO DE CRIAÇÃO DE ARRAY PARA ARMAZENAMENTO DE DADO*/
14 var dadosLista = [];
15 function salvarUser() {
16
17     let nomeUser = document.getElementById("nomeUser").value;
18
19     if(nomeUser){
20         dadosLista.push(nomeUser);
21         criaLista();
22         console.log(dadosLista);
23         document.getElementById("nomeUser").value = "";
24     } else {
25         alert("Usuário favor preencher o campo nome");
26     }
27 }
```

Figura 3 | Função 1 – manipula usuário. Fonte: elaborada pelo autor.

Essa função acessa o formulário e coleta do campo o nome informado pela pessoa que está usando a aplicação; depois, adiciona esse nome ao array criado na linha 14. Na linha 21, foi chamada uma função que veremos a seguir:

```

28 /*FUNÇÃO PARA CRIAR A LISTA*/
29 function criaLista() {
30     let tabela = document.getElementById("tabela").innerHTML = "<tr><th>Nome Usuario</th><th>Ações</th></tr>";
31     for(let i = 0; i <= (dadosLista.length - 1); i++) {
32         tabela += "<tr><td>" + dadosLista[i] + "</td><td><button class='btn btn-success' onclick='editar(this.parentNode.parentNode.rowIndex)'>Editar</button> <button class='btn btn-danger' onclick='excluir(this.parentNode.parentNode.rowIndex)'>Excluir</button></td></tr>";
33     }
34 }
35 }
```

Figura 4 | Função 2 – cria lista de usuários. Fonte: elaborada pelo autor.

Essa função cria uma lista abaixo da linha criada no html, exibida na Figura 2, e ao criá-la, exibe o nome lançado e, à frente dele, dois botões, um para editar e outro para excluir. Esses botões funcionarão graças as funções a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
36  /*FUNÇÃO PARA EXCLUIR A LISTA*/
37  function excluir(i) {
38      dadosLista.splice((i -1), 1);
39      document.getElementById("tabela").deleteRow(i);
40      console.log(dadosLista);
41  }
42
43  /*FUNÇÃO PARA EDITAR ITENS DA LISTA*/
44  function editar(i) {
45      document.getElementById("nomeUser").value = dadosLista[(i -1)];
46      dadosLista.splice(dadosLista[(i -1)], 1);
47      console.log(dadosLista);
48 }
```

Figura 5 | Funções 3 e 4 - editar e excluir. Fonte: elaborada pelo autor.

Agora, ao digitar um nome e clicar no botão Salvar, aparecerá, abaixo, o nome digitado com dois botões, um para editar e outro para excluir, veja:

Nome:

Salvar

Nome Usuario	Ações
Leonardo	<input type="button" value="Editar"/> <input type="button" value="Excluir"/>

Figura 6 | Aplicação de guarda de nomes. Fonte: elaborada pelo autor.

Agora vamos ver um breve resumo de tudo que vimos até o momento, por meio desse infográfico que sintetiza os conhecimentos que conquistamos até aqui.

# DESENVOLVIMENTO EM JAVASCRIPT

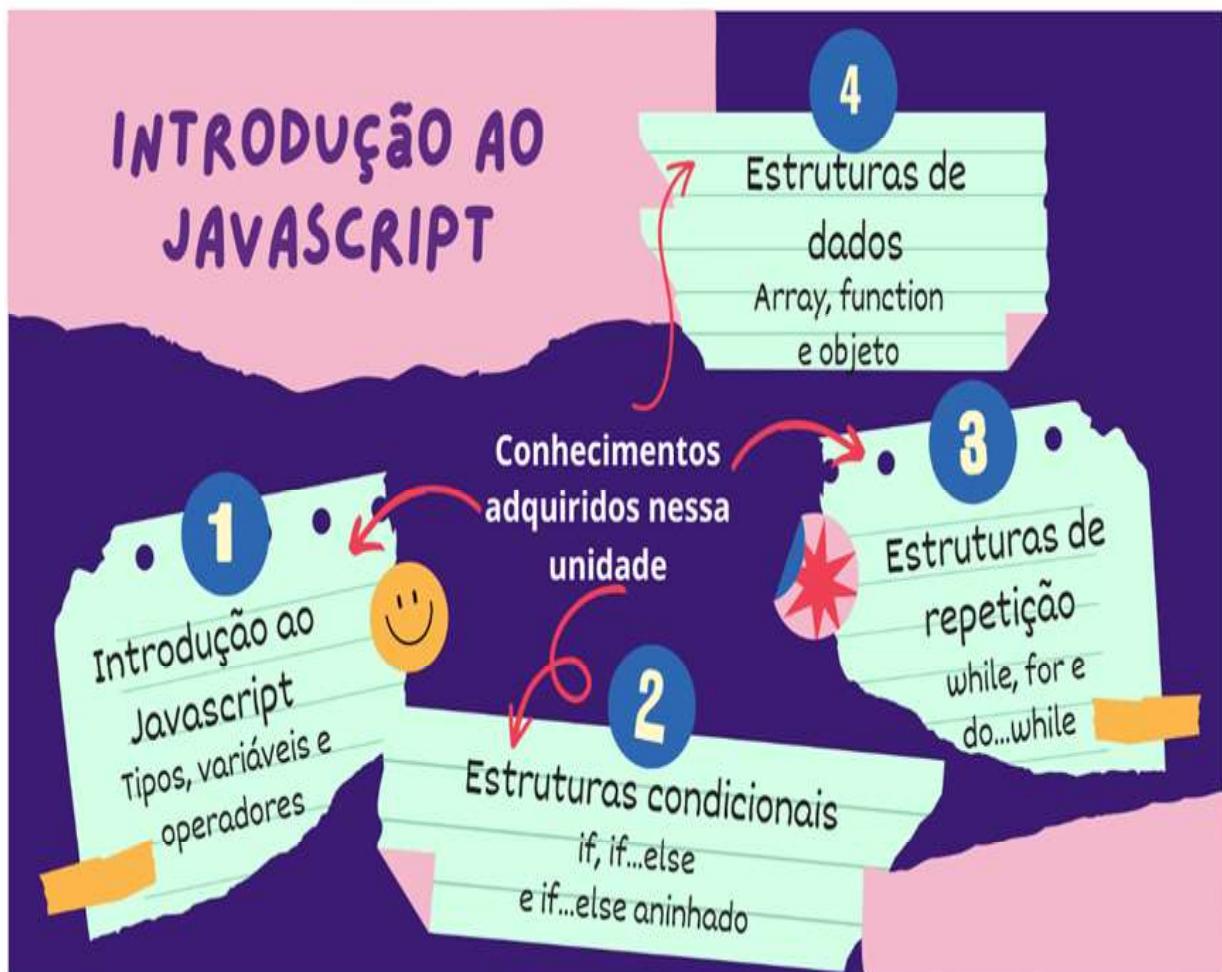


Figura | Introdução ao Javascript. Fonte: elaborada pelo autor.

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Javascript**. 2024. Disponível em: <https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>. Acesso em: 11 jan. 2024.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Javascript descomplicado**: programação para web, IoT e dispositivos móveis. São Paulo: Érica, 2020.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Node.js**: programe de forma rápida e prática. São Paulo: Editora Saraiva, 2021. Disponível em:

<https://integrada.minhabiblioteca.com.br/#/books/9786558110217/>. Acesso em: 11 jan. 2024.

ZABOT, D.; MATOS, E. de S. **Aplicativos com bootstrap e angular**: como desenvolver apps responsivos. São Paulo: Érica, 2020. Disponível em:

<https://integrada.minhabiblioteca.com.br/#/books/9788536533049/>. Acesso em: 11 jan. 2024.

# DESENVOLVIMENTO EM JAVASCRIPT

## Unidade 2

### APIs - Bibliotecas Para Desenvolvimento Em Javascript

#### Aula 1

APIs de Navegador - Manipulando documentos

#### APIs de navegador – manipulando documentos



##### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

#### Ponto de Partida

Nesta aula, trabalharemos o conceito de árvore DOM e entenderemos o que significa esse acrônimo, sendo de suma importância entender o seu funcionamento para conseguir manipular documentos escritos em HTML e CSS utilizando Javascript. Não se preocupe, não será nada complexo, você verá!

Para a compreensão desse novo conceito que aprenderemos, vamos ver um exemplo prático, onde você terá de trabalhar em uma aplicação que solicitará o nome e a idade de uma pessoa; além disso, com o auxílio de uma estrutura de decisão, informar ao usuário se ele é uma pessoa menor de idade ou não. Considere que a maioridade passa a valer a partir dos 18 anos, quando a pessoa fica habilitada à prática de todos os atos da vida civil, logo, você deverá manipular a propriedade style para alterar a cor de fundo da página e a cor do texto, de acordo com a resposta para menor de 18 anos e para maior de idade. Escolha as cores e apresente uma mensagem no corpo da página para o usuário.

Desejamos que esta aula te auxilie no aprofundamento da programação web!

# DESENVOLVIMENTO EM JAVASCRIPT

Vamos Começar!

## Introdução ao DOM

DOM (*Document Object Model*) é uma interface (API) que possibilita manipular o conteúdo de documentos HTML e XML (Flanagan, 2013). Graças ao Modelo de Objeto de Documentos, é possível utilizar uma estrutura padronizada para que os programas possam alterar essa estrutura, seu estilo e conteúdo (MDN, 2023). A árvore DOM é a estrutura do HTML em código, como mostra a figura a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <title></title>
5 </head>
6 <body>
7     <h1></h1>
8     <ul>
9         <li></li>
10    </ul>
11 </body>
12 </html>
```

Figura 1 | Árvore DOM. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

## Métodos e elementos DOM

O acesso a elementos DOM com a linguagem Javascript é feito graças à existência de métodos que permitem a realização desse acesso. Veja alguns exemplos a seguir:

Método de acesso	Funcionalidade e referência do elemento pelo seu ID
getElementById()	Retorna a referência de um elemento pelo seu ID.
getElementsByClassName()	Retorna um vetor de objetos com todos os elementos filhos que possuem o nome da classe informada.
getElementsByName()	Retorna uma coleção de elementos por meio do name deles.
querySelectorAll()	Retorna uma lista de elementos presentes no documento que coincidem com o grupo de seletores especificado.

Quadro 1 | Principais métodos de acesso DOM. Fonte: adaptado de MDN (2024).

## Siga em Frente...

## Manipulação do CSS em um documento usando o DOM

As folhas de estilo são indispensáveis na tarefa de construção de páginas web, pois é graças a elas que é possível a aplicação de formatação às páginas. O CSS (*Cascading Style Sheets*) é responsável por permitir a manipulação de tudo que diz respeito à parte visual (aparência) de uma página web, enquanto isso, o HTML é responsável por todo o conteúdo. O CSS, além de permitir o uso de recursos de formatação, possibilita, também, a reutilização em vários documentos HTML (Alves, 2021).

A sintaxe de uma regra de CSS deve ser escrita da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT



Figura 2 | Sintaxe de regra CS. Fonte: elaborada pelo autor.

Cada objeto Element tem propriedades style e className que permitem aos scripts especificar estilos CSS para um elemento do documento ou alterar os nomes de classe CSS que se aplicam ao referido elemento (Flanigan, 2013). Veja o exemplo a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 window.onload = ()=>{
2
3     // selecionando o objeto
4     var selecionado = document.getElementsByTagName('header');
5
6     // pegando apenas o primeiro resultado
7     selecionado = selecionado[0];
8
9     // alterando a cor do fundo para roxo
10    selecionado.style.backgroundColor = 'purple';
```

Figura 3 | Modifica cor de fundo de elemento. Fonte: elaborada pelo autor.

## Vamos Exercitar?

Agora que você conhece os métodos Javascript para acessar elementos DOM e comprehende como é feito esse acesso, será necessário resolver o problema proposto no início desta aula. O código será construído num arquivo Javascript separadamente; a partir de agora, faremos isso com todos os nossos projetos futuros. Veja como fica o código do arquivo HTML:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset='utf-8'>
5     <title>Verifica idade</title>
6 </head>
7 <body>
8     <div id="mensagem"></div>
9     <script src='exercicio1.js'></script>
10 </body>
11 </html>
```

Figura 3 | Código html – codigoidade.html. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 /*CÓDIGO PARA VERIFICAÇÃO DE MAIORIDADE*/
2 let idade = parseInt(prompt('Informe sua idade: '));
3 let body = document.body;
4 let msg = document.getElementById('mensagem');
5
6 if(idade <= 18){
7     body.style.background = 'Darkred';
8     msg.style.fontSize = 'xx-large';
9     msg.style.color = 'Cornslk';
10    msg.innerHTML = 'Você é menor de idade';
11} else{
12    body.style.background = 'Aquamarine';
13    msg.style.fontSize = 'xx-large';
14    msg.style.color = 'CadeBlue';
15    msg.innerHTML = 'Você é maior de idade';
16}
```

Figura 4 | Código Javascript – exercício1.js. Fonte: elaborada pelo autor.

Note que o código Javascript está sendo chamado antes do fechamento do elemento <body>, e isso acontece para que todos os elementos da página sejam renderizados antes de o código Javascript ser interpretado.

Este é o resultado para as duas situações possíveis:

# DESENVOLVIMENTO EM JAVASCRIPT

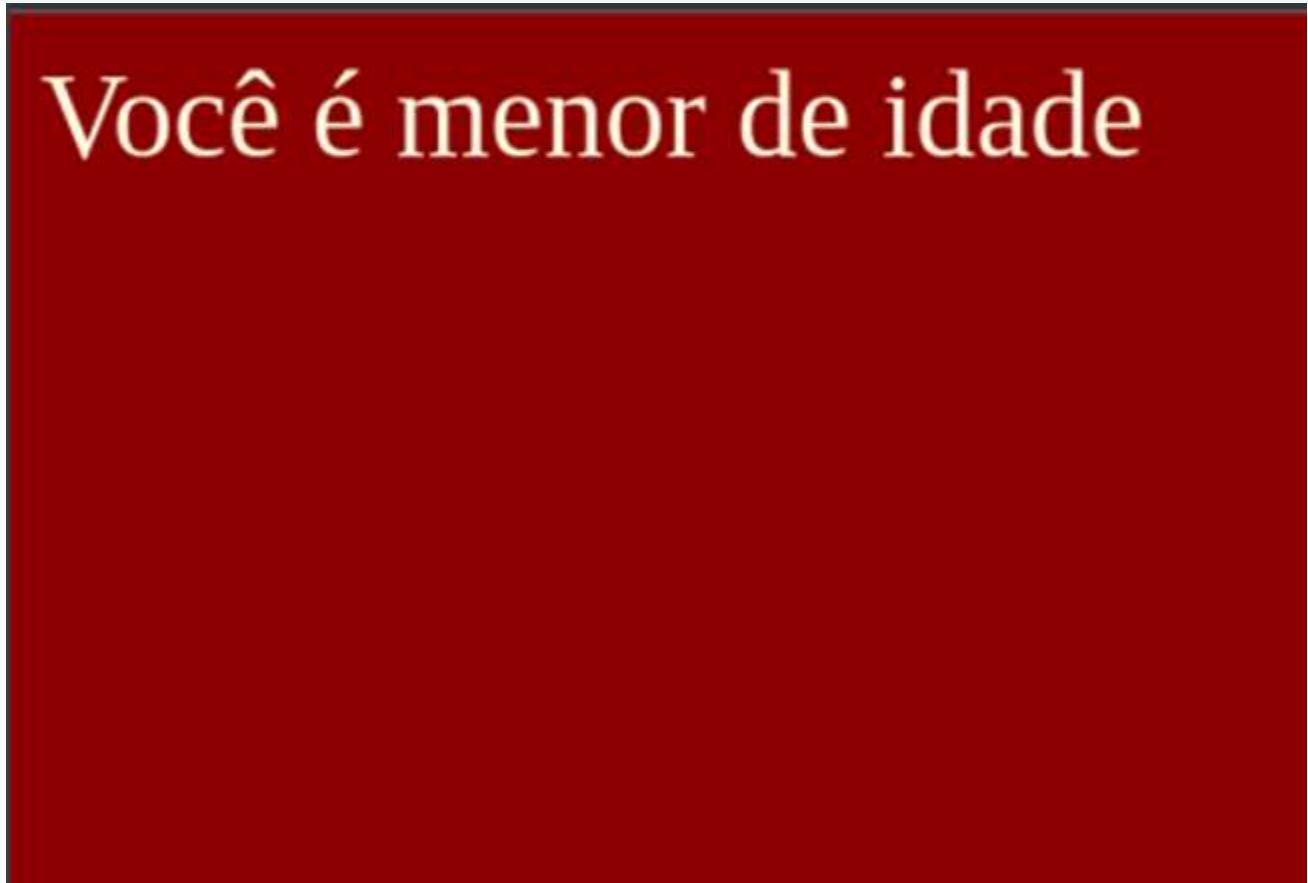


Figura 5 | Resultado – Menor de idade. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

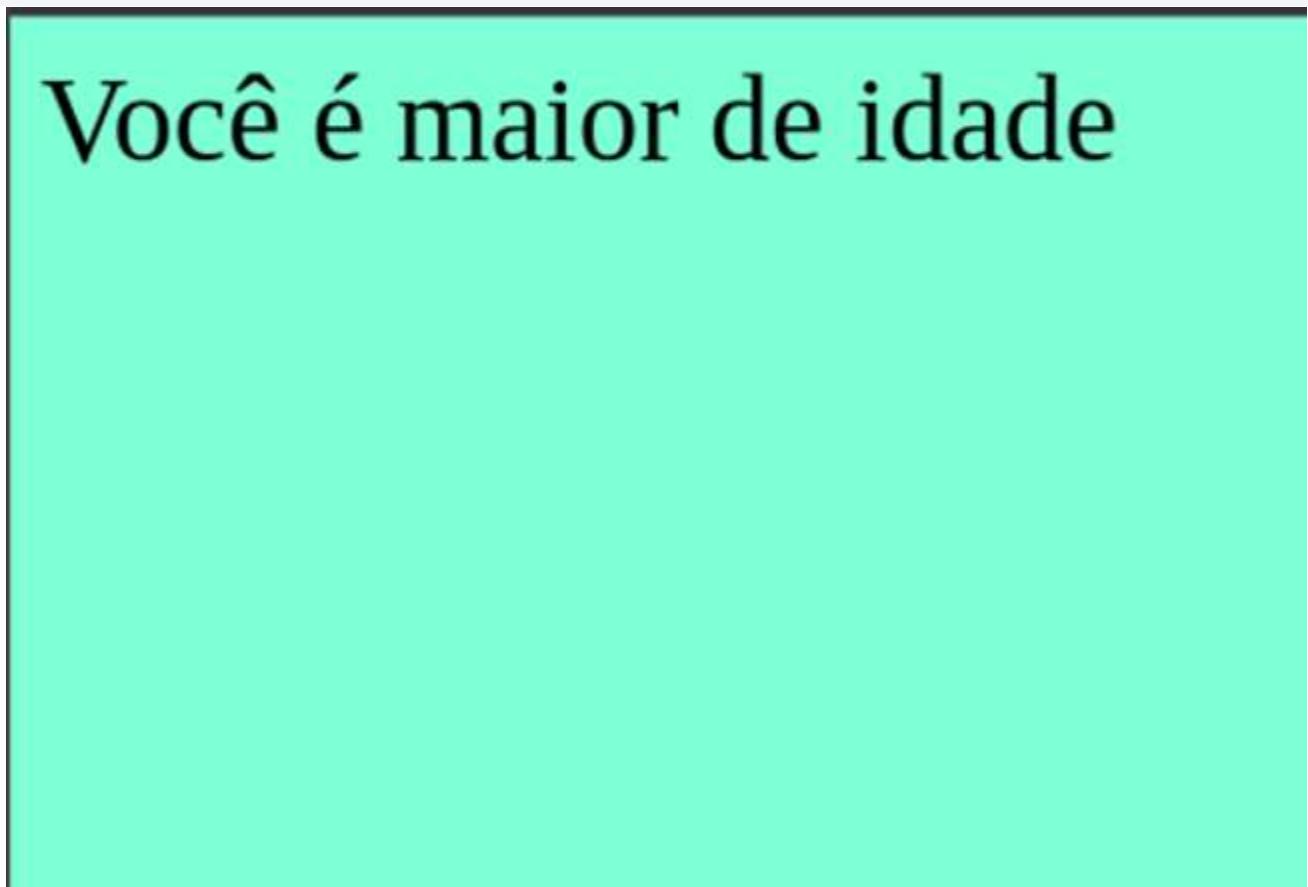


Figura 6 | Código Javascript. Fonte: elaborada pelo autor.

## Saiba mais

Sugerimos a leitura do Capítulo 15 do livro *Javascript*, do autor David Flanagan, a fim de que possa compreender os conceitos relacionados à árvore DOM, bem como a leitura de *Modelo de Objeto de Documento (DOM)*.

MDN WEB DOCS. [Modelo de Objeto de Documento \(DOM\)](#). 2023.

Conheça, por fim, a árvore] DOM e sua interface para desenvolvimento web, disponível em: e os seletores em CSS.

VIEIRA, D. [O que é DOM e por que essa interface é essencial para a web?](#) 2023.

## Referências

ALVES, W. P. **HTML e CSS**: aprenda como construir páginas web. São Paulo: Expressa, 2021.

# DESENVOLVIMENTO EM JAVASCRIPT

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Javascript**. 2024. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Web/API/Document\\_Object\\_Model/Introduction#dom\\_interfaces](https://developer.mozilla.org/pt-BR/docs/Web/API/Document_Object_Model/Introduction#dom_interfaces). Acesso em: 11 jan. 2024.

## Aula 2

APIs de Navegador - Manipulando áudio e gráfico

### APIs de navegador – manipulando áudio e gráfico



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

Os recursos audiovisuais estão sempre presentes no nosso dia a dia, não é mesmo? É muito comum estarmos com algum tipo de dispositivo eletrônico, como o celular, para ouvir música, por exemplo, e o uso desse tipo de recurso em páginas web é muito necessário, pois trata da dimensão interativa de um website que é extremamente importante, afinal, é o que torna a experiência do usuário interessante.

Nesta aula, falaremos um pouco sobre animações e áudios, desenho de formas e aplicação de estilos utilizando a linguagem Javascript, bem como veremos o elemento canvas, disponível no HTML, que torna todo esse trabalho possível de ser desenvolvido em aplicações de internet. Além disso, você será desafiado a resolver um exemplo prático a montar um recurso de carregamento de página, montar o efeito de barra de progresso, presente em muitos websites e que é utilizado enquanto itens estão sendo carregados na página.

# DESENVOLVIMENTO EM JAVASCRIPT

Diante disso, como está a sua animação para conhecer esses recursos e poder utilizá-los em seu projeto? Aprofunde-se e verá que é muito interessante e até mesmo divertido! Vamos lá?

## Vamos Começar!

### Introdução ao canvas

O canvas surge na versão 5 do HTML para facilitar a vida dos programadores de página web. Ele permite que sejam criados objetos gráficos (formas geométricas) e textos; com ele, também é possível criar animações gráficas (Alves, 2021). Esse elemento permite definir um objeto container que pode ser comparado a uma tela branca de pintura, em que, numa analogia, o pintor pode criar sua arte. Agora, para nós, criadores de programas, com esse objeto, podemos criar nossos gráficos utilizando Javascript (Alves, 2021).

Com canvas, é possível criar muitas coisas, pois ele conta com um conjunto extenso de recursos, como recortes, sombra, textos, desenho de linhas, cores, transparência, degradês e muito mais, e parte desses recursos são escritos com a linguagem Javascript (Flanagan, 2013).

Quanto à manipulação do elemento canvas com Javascript, existem alguns métodos importantes que podem auxiliar nessa tarefa. Veja:

Método	Descrição
getContext()	Responsável pelo retorno de um contexto que podemos utilizar na chamada de métodos para criação dos objetos gráficos e textos.
fillStyle()	Permite definir a cor de um objeto.
fillRect()	Utilizado para criação de retângulos.
fillText()	Utilizado para adicionar texto a um elemento canvas.

Quadro 1 | Métodos para manipulação do canvas. Fonte: adaptado de Alves (2021).

Esses são apenas alguns dos vários métodos e funções disponíveis para lidar com elemento canvas.

### Aplicando animação e áudio

Um dos avanços importantíssimos que chegou com a versão 5 do HTML diz respeito aos elementos semânticos. Além disso, entre os vários elementos que surgem nessa versão, está o canvas, responsável por possibilitar a construção de animações em páginas web. Com ele, é possível criar jogos em HTML5, pois ele permite o desenho de gráficos com Javascript (MDN, 2023).

Com esse elemento, é muito simples desenhar um quadrado numa página web, basta executar o seguinte código:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>Quadrado com Canvas</title>
6   <link rel="stylesheet" href="exercicio2.css">
7 </head>
8 <body>
9   <canvas id="square" width="200px" height="200px"></canvas>
10  <script src="exercicio2.js"></script>
11 </body>
12 </html>
```

Figura 1 | exercicio2.html. Fonte: elaborada pelo autor.

```
1 canvas{
2   border: 2px solid black;
3 }
```

Figura 2 | exercicio2.css. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT



Figura 3 | Resultado. Fonte: elaborada pelo autor.

O áudio também é um recurso que pode ser utilizado facilmente, graças ao elemento áudio também criado na versão 5 do HTML.

Imagine que você tenha um arquivo de mp3 chamado música em seu projeto, no mesmo diretório do arquivo index.html, e para reproduzir o som desse arquivo em sua página web, basta inseri-lo no projeto da seguinte forma:

```
1 <audio id="musica" controls autoplay>
2   <source src="música.mp3" type="audio/mp3">
3 </audio>
```

Figura 4 | Elemento áudio. Fonte: adaptada de MDN (2023).

**Siga em Frente...**

**Desenhando formas, aplicando estilo e cores**

# DESENVOLVIMENTO EM JAVASCRIPT

Agora, por meio do ID do elemento canvas, é possível utilizar Javascript para desenhar algo dentro do quadrado criado, nesse caso, um novo quadro, menor e preenchido. Basta adicionar ao código criado anteriormente, e o Javascript que fará isso. Veja:

```
1 // Seleciona o elemento canvas  
2 var canvas = document.getElementById('square');  
3 // Variável de contexto 2d  
4 var ctx = canvas.getContext('2d');  
5 // desenha um quadrado ao centro da forma  
6 ctx.fillRect(45, 45, 100, 100);
```

Figura 5 | exercício2.js. Fonte: adaptada de MDN (2023).



Figura 6 | Resultado. Fonte: adaptada de MDN (2023).

# DESENVOLVIMENTO EM JAVASCRIPT

Note que está sendo utilizado o método `fillRect()`, que é quem cria o quadrado preenchido ao centro. Ele recebe como parâmetro as coordenadas X (coluna), Y (linha), largura e altura (Alves, 2021). Vamos, agora, a um último exemplo, em que faremos o desenho de interseção de dois retângulos e, em um deles, aplicaremos transparência. Veja:

```
1<!DOCTYPE html>
2<html>
3<head>
4    <meta charset='utf-8'>
5    <title>Page Title</title>
6    <link rel='stylesheet' type='text/css' href='exercicio3.css'>
7</head>
8<body onload="draw();">
9    <canvas id="canvas"></canvas>
10   <script src='exercicio3.js'></script>
11</body>
12</html>
```

Figura 7 | `exercicio3.html`. Fonte: adaptada de MDN (2023).

```
1 canvas {
2     width: 150;
3     height: 150;
4 }
```

Figura 8 | `exercicio3.css`. Fonte: adaptada de MDN (2023).

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 function draw() {  
2     var canvas = document.getElementById("canvas");  
3     if (canvas.getContext) {  
4         var ctx = canvas.getContext("2d");  
5  
6         ctx.fillStyle = "rgb(200,0,0)";  
7         ctx.fillRect(10, 10, 85, 80);  
8  
9         ctx.fillStyle = "rgba(0, 0, 200, 0.5)";  
10        ctx.fillRect(30, 30, 85, 80);  
11    }  
12}
```

Figura 9 | exercício3.js. Fonte: adaptada de MDN (2023).

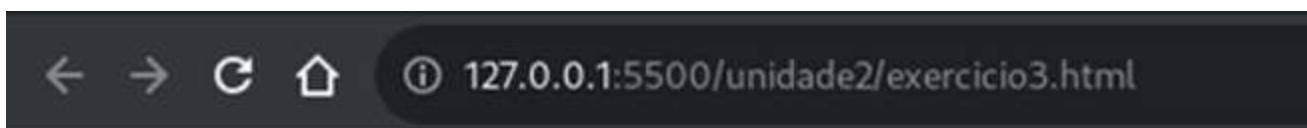


Figura 10 | Resultado. Fonte: adaptada de MDN (2023).

# DESENVOLVIMENTO EM JAVASCRIPT

## Vamos Exercitar?

Como apresentado no início desta aula, você está trabalhando em um projeto em que, entre outros recursos que estão sendo empregados, será necessário montar o efeito de barra de progresso, presente em muitos websites e que é utilizado enquanto itens estão sendo carregados na página.

Alguns dados importantes: o efeito de taxa de atualização é de 10 milissegundo; a variável fator, responsável pela velocidade, receberá 60; e a resolução na qual deverá ser testada é de 1280px de largura. O time de design informou o código de cor a ser utilizado, que é "#4169E1", e teremos o seguinte resultado:

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4   <title>Barra de progresso</title>
5   <meta charset="utf-8">
6   <link rel="stylesheet" href="estilo.css">
7 </head>
8 <body>
9   <canvas id="progress" height="10" width="1280"></canvas>
10
11  <script src="exercicio4.js"></script>
12 </body>
13 </html>
```

Figura 11 | exercicio4.html. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 /* resolução de até 1280px */
2 body{
3     margin: 0;
4     padding: 0;
5 }
6
7 #progress{
8     position: fixed;
9     background-color: #c4c1c1;
10}
```

Figura 12 | estilo.css. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 // iniciando o canvas
2 var canvas = document.getElementById('progress')
3 var ctx = canvas.getContext('2d');
4
5 // configurações
6 var x = 0;
7 var y = 0;
8 var altura = 10;
9 var largura = 0;
10 var fator = 60;
11 var resolucao = 1280;
12
13 // cor da barra requisitada pela equioe
14 ctx.fillStyle = "#4169E1";
15
16 // função que anima a barra de progresso
17 function animacao(){
18     ctx.fillRect(x, y, largura = largura+fator, altura);
19
20     // CÓDIGO AVANÇADO: interrompe a função setInterval()
21     // para evitar carregamento excessivo
22     if(largura > resolucao){
23         clearInterval(atualiza);
24     }
25 }
26
27 // atualiza a barra a cada 10 milissegundos
28 var atualiza = setInterval(animacao, 10)
```

Figura 13 | exercicio4.js. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

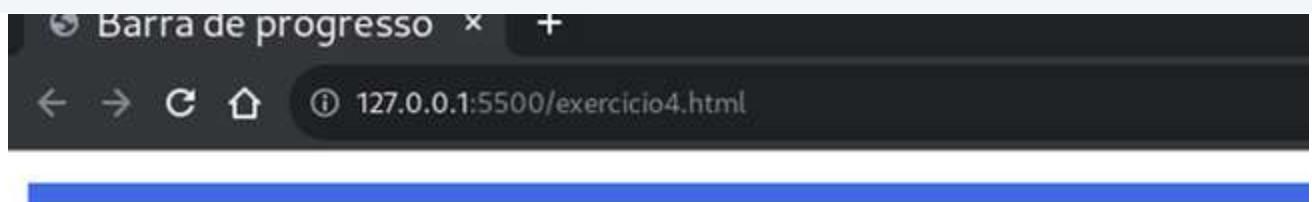


Figura 14 | Resultado. Fonte: elaborada pelo autor.

## Saiba mais

Sugerimos a leitura do Capítulo 21 do livro *Javascript*, de Davi Flanagan, a fim de que compreenda os conceitos relacionados à mídia e gráficos em scripts.

FLANAGAN, D. [JavaScript: o guia definitivo](#). Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

Aprofunde seus conhecimentos no elemento áudio.

MDN WEB DOCS. [Audio](#). 2023.

## Referências

ALVES, W. P. **HTML e CSS**: aprenda como construir páginas web. São Paulo: Expressa, 2021.

FLANAGAN, D. **JavaScript: o guia definitivo**. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Canvas**. 2024. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Web/API/Canvas\\_API](https://developer.mozilla.org/pt-BR/docs/Web/API/Canvas_API). Acesso em: 11 jan. 2024.

# DESENVOLVIMENTO EM JAVASCRIPT

## Aula 3

APIS de Navegador - Comunicando com o servidor

### APIs de navegador – comunicando com o servidor



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

A arquitetura cliente-servidor é uma das mais utilizadas para o desenvolvimento de aplicações web. Esse tipo de desenvolvimento tem sido muito facilitado pelas inúmeras ferramentas que surgiram e surgem todos os dias, como as APIs de comunicação, que fazem parte do rol de soluções criadas para facilitar a vida de quem trabalha com programação.

A comunicação com banco de dados por meio de tecnologias como o AJAX, que, de certa forma, não é tão recente, tem melhorado consideravelmente a experiência do usuário. As conexões assíncronas só são possíveis graças às linguagens HTML5 e Javascript a partir da API XMLHttpRequest, que realiza requisições HTTP e HTTPS diretamente para o servidor, como veremos adiante.

Agora vamos a um desafio, imagine que você está num programa de estágio de uma empresa de tecnologia e a liderança do seu grupo lhe passou um desafio: você deve montar um projeto simples utilizando XMLHttpRequest com Javascript, pois, provavelmente, você será alocado num time que precisará utilizar essa tecnologia. Disposto a encarar o desafio? Vamos começar?

Bons estudos!

## Vamos Começar!

# DESENVOLVIMENTO EM JAVASCRIPT

## Introdução ao Ajax

Ajax é o acrônimo (sem letras maiúsculas) de Asynchronous Javascript and XML cujo termo foi cunhado por Jesse James Garrett e surgiu pela primeira vez em seu ensaio de fevereiro de 2005 “Ajax: a new approach to web applications” (Flanagan, 2013). Percebe-se que Ajax é uma tecnologia relativamente antiga, mas que ainda tem muita relevância pelas soluções que entrega. Atualmente, Ajax é considerado termo útil para uma arquitetura de aplicativo web baseada em scripts de requisições HTTP (Flanagan, 2013).

Um bom exemplo de uso de Ajax está no buscador Google, que implementou essa tecnologia. Ao digitar algo na barra de pesquisa, o site usa Ajax para obter resultados comuns do banco de dados a cada tecla pressionada (Freitas, 2021).

## Métodos e propriedades XMLHttpRequest

Os navegadores definem suas APIs HTTP em uma classe XMLHttpRequest, e cada instância dessa classe representa um único par de requisição/resposta (Flanagan, 2013). Essa classe surge, então, para facilitar comunicação assíncrona com servidores; com ela, os dados podem ser comunicados por meio de requisições HTTP (ou HTTPS) e arquivos no formato XML, permitindo recuperar dados de um URL sem necessitar atualizar a página inteira. Veja como se dá a comunicação:

Requisição HTTP	Resposta HTTP
Método ou “verbo” da requisição HTTP.	Código de status numérico e textual indicando sucesso ou falha.
O URL que está sendo solicitado.	Conjunto de cabeçalho de resposta.
Conjunto opcional de cabeçalhos de pedido.	Corpo da resposta.
Corpo e requisição opcional.	

Quadro 1 | Comunicação assíncrona. Fonte: adaptado de Flanagan (2013).

Simplificando: ao se criar uma instância do objeto XMLHttpRequest, abre-se uma url e envia-se a requisição. O resultado será disponibilizado quando a transação for completada

## Siga em Frente...

## Ajax e banco de dados

O processo de comunicação entre cliente e servidor utilizando o Ajax é relativamente simples. Veja um exemplo de uma função para executar essa transação:

# DESENVOLVIMENTO EM JAVASCRIPT

```

1 function reqListener() {
2   console.log(this.responseText);
3 }
4 // instancia objeto
5 var oReq = new XMLHttpRequest();
6
7 oReq.onload = reqListener;
8 // método para abrir a url
9 oReq.open("get", "yourFile.txt", true);
10 // método para envio da requisição
11 oReq.send();

```

Figura 1 | Requisição XMLHttpRequest. Fonte: adaptada de MDN (2024).

Vejamos os principais métodos e propriedades para o uso dessa classe:

Métodos/Propriedades	Descrição
open()	Inicializa um pedido.
getResponseHeader()	Retorna string contendo o texto do cabeçalho especificado.
send()	Envia a solicitação.
readyState	Retorna o estado de um XMLHttpRequest
response	Retorna o conteúdo do corpo da resposta.

Quadro 2 | Principais métodos e propriedades. Fonte: adaptado de MDN (2024).

Ainda, temos alguns códigos de status de resposta HTTP que são importantes; eles indicam o tipo de problema que se está tendo com uma requisição dessa natureza. Veja:

Código	Classe de código
100 – 199	Respostas informativas.
200 – 299	Respostas bem-sucedidas.

# DESENVOLVIMENTO EM JAVASCRIPT

300 – 399	Mensagens de relacionamento.
400 – 499	Respostas de erro do cliente.
500 – 599	Respostas de erro do servidor.

Quadro 3 | Códigos de status de resposta HTTP. Fonte: adaptado de MDN (2024).

## Vamos Exercitar?

Considerando o desafio que você recebeu no início da aula, agora, você precisa colocar em prática os conhecimentos adquiridos até aqui. Como a intenção do desafio é testar sua capacidade de resolução de problemas, você pode montar um projeto simples, utilizando o XMLHttpRequest.

No entanto, antes de mais nada, você precisa de uma URL que possa ser consultada com uma requisição e dar uma resposta. Como a transação acontecerá com arquivo JSON, sugerimos que utilize a seguinte URL: <https://jsonplaceholder.typicode.com/posts/1>. Ao acessá-la, verá que ela apresenta 4 quatro nomes, mas vamos trabalhar com o nome title, e o código HTML ficará assim:

```

1<!DOCTYPE html>
2<html>
3
4<head>
5    <meta charset='utf-8'>
6    <title>Utilização do XMLHttpRequest</title>
7</head>
8
9<body>
10   <h1>Utilizando XMLHttpRequest</h1>
11   <button type="button" onclick="Conexao()">Testar</button>
12<script src="exercicio5.js"></script>
13</body>
14
15</html>
```

Figura 2 | HTML para uso de XMLHttpRequest. Fonte: adaptada de MDN (2024).

Agora, vamos montar o código Javascript que fará a consulta à URL informada, fazendo uma requisição e aguardando resposta. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 function Conexao() {
2     // Cria um novo objeto XMLHttpRequest
3     const xhr = new XMLHttpRequest();
4
5     // Configura a solicitação, especificando o método (GET) e a URL
6     xhr.open("GET", "https://jsonplaceholder.typicode.com/posts/1", true);
7
8     // Define uma função de callback para lidar com a resposta
9     xhr.onload = function () {
10         if (xhr.status === 200) {
11             // A solicitação foi bem-sucedida, e o status da resposta é 200 (OK)
12             const response = JSON.parse(xhr.responseText);
13             console.log("Título do post:", response.title);
14         } else {
15             // A solicitação falhou
16             console.log("Ocorreu um erro durante a solicitação.");
17         }
18     };
19
20     // Envia a solicitação
21     xhr.send();
}
```

Figura 3 | Javascript para requisição. Fonte: elaborada pelo autor.

Note que realizamos o teste para verificar se teve como resposta o código 200 utilizando estrutura de decisão, bem como escrevemos a resposta no console do navegador. Caso aconteça algum erro, temos uma mensagem para essa possibilidade.

## Saiba mais

Sugerimos a leitura do Capítulo 18 do livro *JavaScript: o guia definitivo*, de David Flanagan, que fala sobre Scripts HTTP.

FLANAGAN, D. [JavaScript: o guia definitivo](#). Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

Sugerimos também o capítulo *Definição de Ajax*, do livro *Programação Back End III*, de Freitas et al.

FREITAS, P. H. C. et al. [Programação back end III](#). Porto Alegre: SAGAH, 2021.

Sugerimos, também, a leitura do texto *Usando XMLHttpRequest*.

MDN WEB DOCS. [Usando XMLHttpRequest](#). 2023.

# DESENVOLVIMENTO EM JAVASCRIPT

E, ainda, o capítulo *Introdução à arquitetura de sistemas*, do livro *Arquitetura de sistemas*, de Zenker, Santos e Couto, a fim de que compreenda os conceitos relacionados à arquitetura cliente-servidor.

ZENKER, A. M.; SANTOS, J. C. dos; COUTO, Júlia M C.; et al. [Arquitetura de sistemas](#). Grupo A, 2019.

## Referências

ALVES, W. P. **HTML e CSS**: aprenda como construir páginas web. São Paulo: Expressa, 2021.

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

FREITAS, P. H. C. et al. **Programação back end III**. Porto Alegre: SAGAH, 2021.

MDN WEB DOCS. **Canvas**. 2024. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Web/API/Canvas\\_API](https://developer.mozilla.org/pt-BR/docs/Web/API/Canvas_API). Acesso em: 12 jan. 2024.

## Aula 4

Trabalhando com APIs de Terceiros

### Trabalhando com APIs de terceiros



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

# DESENVOLVIMENTO EM JAVASCRIPT

A linguagem Javascript, especialmente client-side, possui diversas APIs disponíveis, que, por sua vez, não fazem parte da linguagem em si, mas são escritas sobre o core da linguagem JavaScript, fornecendo superpoderes para serem utilizados em seu código (MDN, 2023), bem como recursos para facilitar a vida de quem esteja montando uma aplicação web. Aliás, existem diversos tipos de APIs, no entanto, hoje, conheceremos as mais importantes e indispesáveis para qualquer pessoa que queira desenvolver boas aplicações.

A fim de que você entenda melhor as funcionalidades e finalidades das Web APIs, você terá um desafio de construir uma aplicação de geolocalização que utilize uma delas de um jeito bem fácil, para que você entenda bem o seu funcionamento.

Você é capaz de fazer coisas incríveis, basta acreditar em sim! Diante disso, vamos descobrir do que você é capaz?

## Vamos Começar!

## Principais APIs de terceiros

As APIs (*Application Programming Interfaces*) são construções disponíveis nas linguagens de programação que permitem aos desenvolvedores uma criação facilitada de funcionalidades complexas, porque essas APIs fornecem funcionalidades por meio da abstração do código mais complexo, permitindo à pessoa que está escrevendo a aplicação o uso de sintaxes mais simples em seu lugar (MDN, 2023).

As APIs de terceiro são chamadas assim pois se trata daquelas APIs que não fazem parte do navegador. Assim, você precisa recuperar seu código e informações de outra parte da web (MDN, 2023). A API do Twitter possibilita a exibição dos últimos tweets de uma conta em um determinado site, e esse é apenas um exemplo do que uma API pode oferecer.

Existem algumas APIs de terceiros que são mais conhecidas e buscadas por qualquer pessoa que esteja desenvolvendo aplicações que necessitem delas, e as mais populares são:

API	Descrição
Twitter API	Permite coisas, como mostrar os últimos tweets em um site.
Google Maps API	Permite várias coisas com uso de mapas em páginas web.
Facebook suite of APIs	Permite o uso do ecossistema do facebook, como aproveitar o login da rede social em sua aplicação.
Youtube API	Permite, entre outras coisas, incorporar vídeos do youtube em páginas web.

# DESENVOLVIMENTO EM JAVASCRIPT

Quadro 1 | APIs populares. Fonte: adaptado de MDN (2023).

Vejamos agora, na prática, como funciona o consumo de APIs utilizando para esse exemplo a API da VIACEP. Essa API fornece dados de endereço a partir do número de CEP. Primeiro, vamos construir uma página de endereço simples, com um formulário em HTML5 utilizando Bootstrap para o estilo da página. Veja:

```

1 <form method="get" action=".">
2   <div class="container">
3     <h1>Cadastro de Endereço</h1>
4     <!-- CAMPO CEP -->
5     <div class="form-group col-md-2">
6       <label for="inputCEP">CEP</label>
7       <input type="text" class="form-control" id="cep">
8     </div><br><br>
9     <!-- BLOCO PARA DADOS DE RUA, NÚMERO E BAIRRO -->
10    <div class="row">
11      <div class="form-group col-md-4">
12        <label for="rua">Rua</label>
13        <input type="text" class="form-control" id="rua" placeholder="Rua dos Bobos">
14      </div>
15      <div class="form-group col-md-2">
16        <label for="inputAddress">N.º</label>
17        <input type="text" class="form-control" id="numero" placeholder="nº 0">
18      </div>
19      <div class="form-group col-md-4">
20        <label for="inputAddress2">Bairro</label>
21        <input type="text" class="form-control" id="bairro" placeholder="Apartamento, hotel, casa, etc.">
22      </div>
23    </div><br><br>
```

Figura 1 | Formulário endereço. Fonte: elaborada pelo autor.

```

24   <!-- BLOCO PARA DADOS DE CIDADE E ESTADO -->
25   <div class="row">
26     <div class="form-group col-md-3">
27       <label for="complemento">Complemento</label>
28       <input type="text" class="form-control" id="complemento">
29     </div>
30     <div class="form-group col-md-4">
31       <label for="cidade">Cidade</label>
32       <input type="text" class="form-control" id="cidade">
33     </div>
34     <div class="form-group col-md-3">
35       <label for="estado">Estado</label>
36       <input type="text" class="form-control" id="estado">
37     </div>
38   </div>
39   <br><br>
40   <!-- <div id="centro"> -->
41   <button class="btn btn-primary" onclick="">Entrar</button>
42   <!-- </div> -->
43 </div>
44 </form>
```

Figura 2 | Formulário endereço - cont. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

Note que o código se refere ao formulário construído em HTML com Bootstrap, que deve ser colocado entre os elementos <body> e </body>. O nome do arquivo pode ser cadendereco.html.

Agora, vamos ver o código em Javascript que fará a consulta à API da ViaCEP.

```
1 'use strict'; //Modo "Restrito"
2 //Consumindo API de CEP, do ViaCep
3 // https://viacep.com.br/
4
5 //Limpa o Form (do CEP pra baixo)...
6 const limparFormulario = (endereco) =>{
7     document.getElementById('rua').value = '';
8     document.getElementById('bairro').value = '';
9     document.getElementById('cidade').value = '';
10    document.getElementById('estado').value = '';
11 }
12
13 //Preenche os campos relacionados ao CEP...
14 const preencherFormulario = (endereco) =>{
15     document.getElementById('rua').value = endereco.logradouro;
16     document.getElementById('bairro').value = endereco.bairro;
17     document.getElementById('cidade').value = endereco.localidade;
18     document.getElementById('estado').value = endereco.uf;
19 }
20
21 //Verifica se o CEP é válido...
22 const eNumero = (numero) => /^[0-9]+$/ .test(numero); //Expressão Regular
23 // É possível testar e entender a RegEx em https://www.regexpal.com/
24 const cepValido = (cep) => cep.length == 8 && eNumero(cep);
25
26 //Consumindo API... 2- passo
27 const pesquisarCep = async() => {
28     limparFormulario();
29     const url = `https://viacep.com.br/ws/${cep.value}/json/`;
30 }
```

Figura 3 | Consumo da API ViaCEP. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```

30
31  if(cepValido(cep.value)){
32      const dados = await fetch(url); //await = esperar
33      const adres = await dados.json(); // fetch = promessa
34
35      // hasOwnProperty retorna um booleano indicando se o objeto
36      // possui a propriedade especificada como uma propriedade definida
37      // no próprio objeto em questão
38      if(adres.hasOwnProperty('erro')){
39          // document.getElementById('rua').value = 'CEP não encontrado!';
40          alert('CEP não encontrado!');
41      }else {
42          preencherFormulario(adres);
43      }
44  }else{
45      // document.getElementById('rua').value = 'CEP incorreto!';
46      alert('CEP incorreto!');
47  }
48 }
49
50 //Adicionando um evento DOM, no input CEP... 1- passo
51 document.getElementById('cep').addEventListener('focusout', pesquisarCep);

```

Figura 4 | Consumo da API ViaCEP - cont. Fonte: elaborada pelo autor.

**Saiba mais:** esse projeto está documentado e disponível em uma versão mais completa em:  
<https://bit.ly/consulta-cep-rep>

**Siga em Frente...**

## Manipulação de dados com APIs de terceiros

O processo de manipulação de dados consiste, basicamente, no tratamento dos dados coletados para um fim específico. Nesse tratamento, pode haver etapas de modelagem, limpeza, validação e preparação do dado, que pode ou não ser publicado. Um bom exemplo é a utilização de geolocalização, que, em muitos sentidos, pode oferecer vários benefícios. Vamos ver um exemplo com Open Layers, recurso que facilita a utilização de mapas dinâmicos em páginas web. Com ele, é possível ver blocos de mapa, dados vetoriais e marcadores carregados de qualquer fonte, e o melhor: é totalmente gratuito sob licença BSD e 2 cláusulas.

Imagine que você precisa coletar a posição dos usuários que acessam uma determinada página web. A intenção não é obter a localização dos possíveis interessados no produto que está sendo vendido nessa página, uma vez que, para isso, é preciso autorização, mas é possível saber ao menos a localidade do usuário. O projeto ficaria assim:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1<!DOCTYPE html>
2<html>
3  <head>
4    <title>API MAPA</title>
5    <meta charset="utf-8">
6    <link rel="stylesheet" href="geolocalizacao.css">
7    <link rel="stylesheet" href="https://cdn.jsdelivr.net/gh/openlayers/openlayers.github.io@master/en/v6.9.0/
css/ol.css" type="text/css">
8    <script src="https://cdn.jsdelivr.net/gh/openlayers/openlayers.github.io@master/en/v6.9.0/build/ol.js"></
script>
9  </head>
10 <body>
11   <div id="mapa"></div>
12   <script src="geolocalizacao.js"></script>
13 </body>
14 </html>
```

Figura 5 | geolocalizacao.html. Fonte: elaborada pelo autor.

```
1 #mapa {
2   width: 600px;
3   height: 600px;
4 }
```

Figura 6 | geolocalizacao.css. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 // coleta a localização atual
2 navigator.geolocation.getCurrentPosition(position => {
3
4     // coleta latitude e longitude
5     const { latitude, longitude } = position.coords;
6
7     // imprimindo mapa
8     var map = new ol.Map({
9         target: 'mapa',
10        layers: [
11            new ol.layer.Tile({
12                source: new ol.source.OSM()
13            })
14        ],
15        view: new ol.View({
16            center: ol.proj.fromLonLat([longitude, latitude]),
17            zoom: 10 // aumente este número para aproximar
18        })
19    });
20});
```

Figura 7 | geolocalizacao.js. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

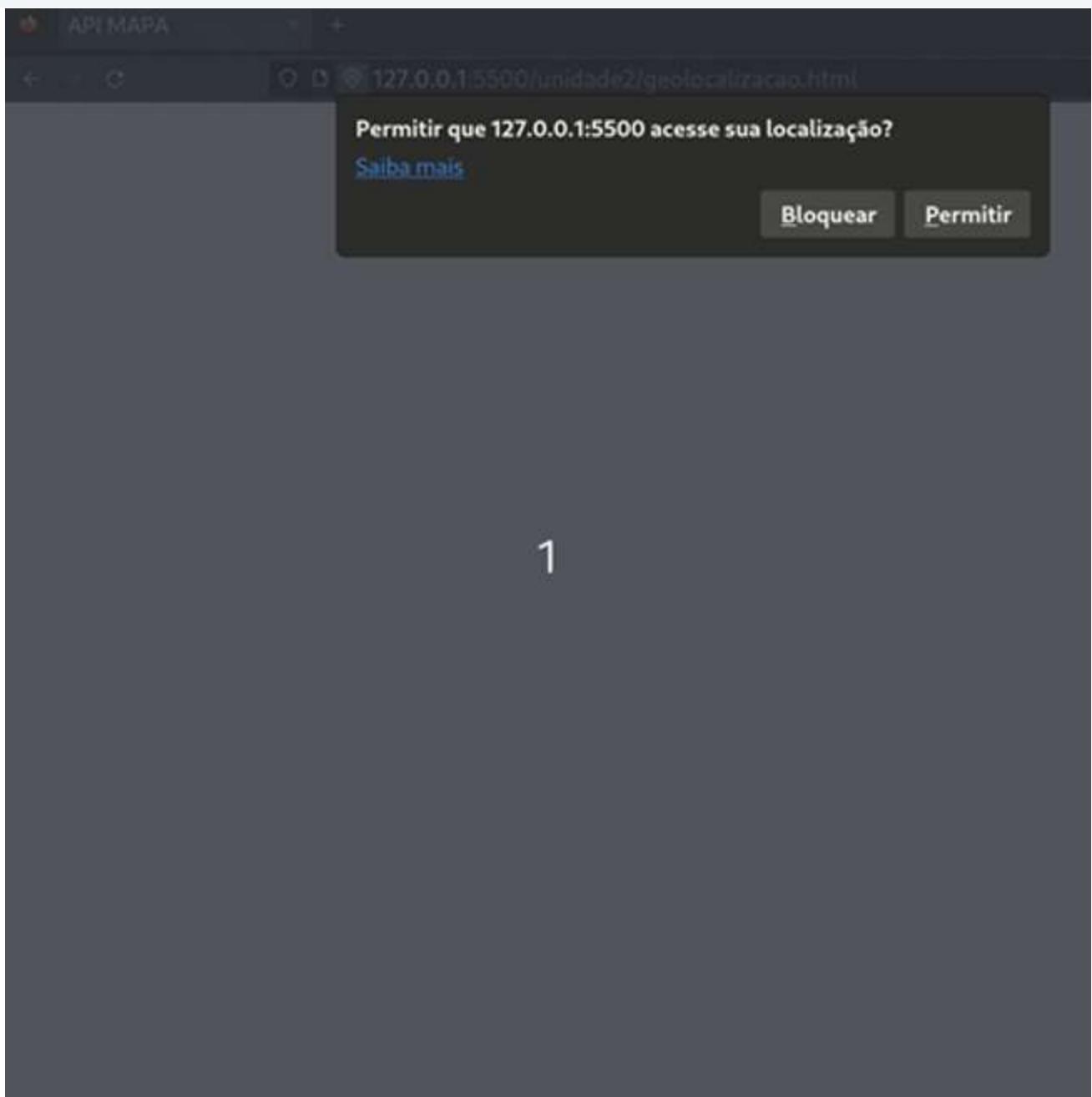


Figura 8 | Resultado animado. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

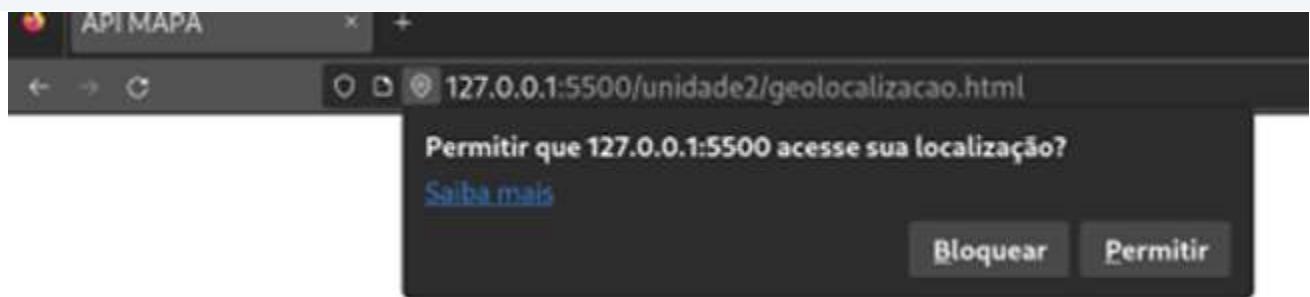


Figura 9 | Resultado – foto 1. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

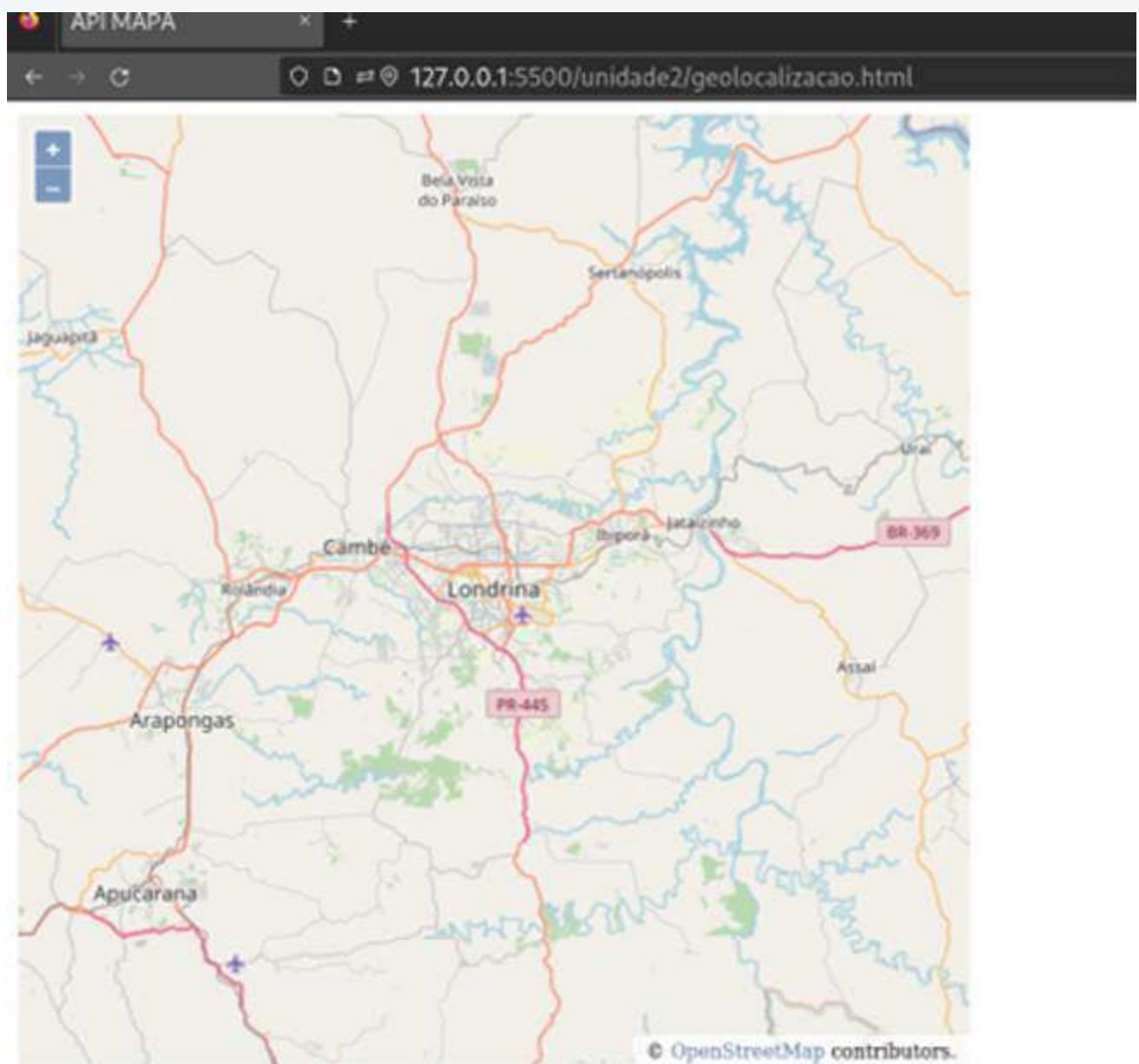


Figura 10 | Resultado – foto 2. Fonte: elaborada pelo autor.

## Vamos Exercitar?

Você precisa desenvolver uma aplicação que, utilizando API, identifique os destinos turísticos mais procurados em duas cidades específicas. As cidades são: Santiago, no Chile, e Gramado, no Rio Grande do Sul, Brasil. Vamos utilizar o openlayers para esse propósito; para tanto, iniciaremos construindo a página HTML, que ficará da seguinte forma:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1<!doctype html>
2<html>
3<head>
4    <title>Agência de turismo</title>
5    <meta charset="utf-8">
6    <link rel="stylesheet" href="destinos.css">
7
8    <!-- biblioteca OpenLayers -->
9    <link rel="stylesheet"
10       href="https://cdn.jsdelivr.net/gh/openlayers/openlayers.github.io@master/en/v6.9.0/css/ol.css" type="text/css">
11   <script src="https://cdn.jsdelivr.net/gh/openlayers/openlayers.github.io@master/en/v6.9.0/build/ol.js"></script>
12   <!--Término-->
13</head>
14<body>
15    <h1 id="turismo">Destinos turísticos</h1>
16
17    <div>
18        <div id="local-1"></div>
19        <div id="local-2"></div>
20    </div>
21    <script src="destinos.js"></script>
22</body>
23</html>
```

Figura 11 | destinos.html. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 * {  
2     margin: 0;  
3 }  
4 #turismo{  
5     text-align: center;  
6 }  
7  
8 #local-1, #local-2{  
9     width: 50%;  
10    height: 300px;  
11    float: left;  
12 }
```

Figura 12 | destinos.css. Fonte: elaborada pelo autor.

Agora, vamos montar o arquivo Javascript para exibir as duas localidades. Daremos ao usuário a escolha do zoom que deve ser dado ao mapa, lembrando que, quanto mais zoom, mais próxima é a visualização. Limite essa visualização para um zoom entre 5 e 15.

# DESENVOLVIMENTO EM JAVASCRIPT

Montamos uma estrutura de dados com nome da cidade, país, latitude e longitude, como você pode ver na linha 2.

```
1 // laço de repetição para gerar mapas
2 let turismo = [["Santiago, Chile", -33.44758, -70.67172], ["Gramado, Brasil", -29.37463, -50.87402]];
3 for(i=0; i < turismo.length; i++){
4     var map = new ol.Map({
5         target: 'local-'+(i+1),
6         layers: [
7             new ol.layer.Tile({
8                 source: new ol.source.OSM()
9             })
10        ],
11        view: new ol.View({
12            center: ol.proj.fromLonLat([
13                turismo[i][2],
14                turismo[i][1]
15            ]),
16            zoom: parseInt(prompt("Zoom desejado [entre 5 e 15]: ")),
17        })
18    });
19 }
```

Figura 13 | destinos.js. Fonte: elaborada pelo autor.

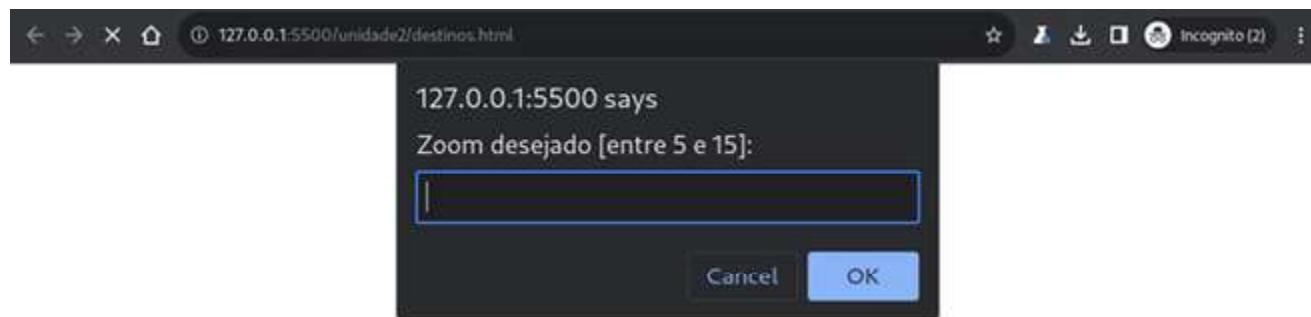


Figura 14 | Resultado – tela 1. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT



Figura 15 | Resultado – tela 2. Fonte: elaborada pelo autor.

Na tela 1 do resultado, será exibida uma mensagem solicitando ao usuário que defina o nível de zoom que deseja para os mapas. No resultado, será exibido o mapa de Santiago com nível de zoom 10 e de Gramado com nível de zoom 15.

## Saiba mais

Sugerimos a leitura do Capítulo 22 *APIs de HTML5* do livro *JavaScript: o guia definitivo*, que trata de APIs.

FLANAGAN, D. [JavaScript: o guia definitivo](#). Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

Também sugerimos a leitura do artigo *Introdução às web APIs*.

MDN WEB DOCS. [Introdução às Web APIs](#). 2023.

Conheça melhor a documentação da API da [ViaCEP](#).

## Referências

# DESENVOLVIMENTO EM JAVASCRIPT

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Introdução às web APIs**. 2023. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Client-side\\_web\\_APIs/Introduction](https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Client-side_web_APIs/Introduction). Acesso em: 12 jan. 2024.

OPEN LAYERS. **API Doc**. 2023. Disponível em: <https://openlayers.org/en/latest/apidoc/>. Acesso em: 12 jan. 2024.

## Aula 5

Encerramento da Unidade

### Videoaula de Encerramento



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Chegada

A utilização das APIs, sejam elas nativas do navegador, sejam aquelas criadas por terceiros, possibilita-nos uma infinidade de possibilidades no que diz respeito à criação de aplicações web. As reflexões feitas ao longo do conteúdo estudado dão conta de vários aspectos indispensáveis na construção desse tipo de aplicação. Por exemplo, conhecemos a API DOM (*Document Object Model*), que serve como interface para a programação de arquivos HTML ou XML (Flanagan, 2013), sendo ela uma maneira de representar os elementos de uma página escritos em HTML.

Em seguida, notamos as possibilidades que surgem com a criação do elemento canvas no HTML, que permite, entre outras coisas, a criação de animação utilizando-se o Javascript. Sobre o canvas, também foram feitas explanações a respeito dos principais métodos para sua

# DESENVOLVIMENTO EM JAVASCRIPT

manipulação (Alves, 2021); além disso, você aprendeu como manipular áudio por meio do elemento áudio, no HTML.

Ainda sobre a animação, foi possível ter uma noção de como é desenhar formas definindo estilos e cores no código escrito — um processo bem interessante, já que é muito utilizado na criação de jogos com a linguagem HTML5. Adiante, vimos um pouco sobre comunicação com servidores utilizando AJAX e a propriedade XMLHttpRequest, muito útil nesse tipo de necessidade.

Finalmente, conhecemos alguns recursos de APIs de terceiros, que são indispensáveis na construção de soluções web, e como você deve ter notado, existem várias, para os mais variados propósitos. Um esforço necessário para compreendê-las é ler a documentação disponível com cada uma, que oferece todas as informações necessárias para o uso da API em questão. Os conteúdos estudados nesta Unidade são necessários para desenvolver a competência que é implementar e utilizar APIs no desenvolvimento com Javascript.

## É Hora de Praticar!



### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Para que você comprehenda como utilizar as APIs com linguagem Javascript, é primordial, inicialmente, que entenda a documentação da API que se pretende utilizar. Só de posse dessas informações, você será capaz de aplicar os conhecimentos necessários para manipulação de documentos, áudios e gráficos, além, é claro, de compreender a dinâmica de uso de APIs de terceiros — pontos fundamentais para que suas aplicações fiquem cada vez mais dinâmicas e interativas.

E já que você estudou bastante sobre APIs, que tal construir uma aplicação utilizando a API do ChatGPT para consultas? Crie um protótipo de um Chatbot para consultas rápidas e curtas. A documentação da API da OpenAI, empresa dona do ChatGPT, é bem completa e tem bastante informação. Será necessário, inicialmente, criar uma conta para acessar o ChatGPT; caso você já tenha, será interessante criar outra, pois há um limite de uso de consultas ao ChatGPT que, provavelmente, você já alcançou. Vamos lá?

- Você já havia visto as APIs para desenvolvimento web em outras linguagens de programação?
- No processo de escrita de código utilizando API de terceiros, você foi capaz de identificar a documentação e, com ela, empregar o código necessário para uso da API?

# DESENVOLVIMENTO EM JAVASCRIPT

- Você compreendeu como as comunicações entre cliente e servidor são realizadas de forma síncrona e assíncrona?

Antes, crie um projeto simples, com um formulário com um único campo, para que a pergunta seja digitada. Utilize CSS para centralizar o formulário na página, e caso queira estilizar um pouco mais, fique à vontade. Para que fique claro, o projeto apresentado aqui será o mais simples possível. Veja:

```

1 <!DOCTYPE html>
2 <html lang="pt-br">
3
4 <head>
5   <meta charset="UTF-8">
6   <title>Consumo API - ChatGPT</title>
7   <link rel="stylesheet" href="chatgpt.css">
8 </head>
9
10 <body>
11   <form action="" id="formQuestion">
12     <h1>Pergunte ao chatGPT</h1>
13     <label for="">Formule a pergunta</label><br>
14     <input type="text" id="question"><br><br>
15     <button type="button" onclick="consultaChat()">Perguntar</button><br><br>
16     <div id="pergunta"></div><br><br>
17     <div id="resposta"></div>
18   </form>
19
20   <script src="chatgpt.js"></script>
21 </body>
22
23 </html>
```

Figura 1 | Projeto chatgpt.html. Fonte: elaborada pelo autor.

```

1 #formQuestion{
2   text-align: center;
3   width: 50%;
4   margin-left: auto;
5   margin-right: auto;
6 }
7 input{
8   width: 350px;
9 }
```

Figura 2 | chatgpt.css. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

Agora, vamos construir o código Javascript, que consultará a API do ChatGPT. A documentação é muito completa e conta com todas as informações necessárias para fazer a conexão com a API funcionar perfeitamente. Leia com atenção.

Será necessária a criação de uma chave a ser utilizada para comunicação com a API, e essa chave pode ser criada no endereço: <https://platform.openai.com/account/api-keys>.

O código Javascript ficará assim:

```

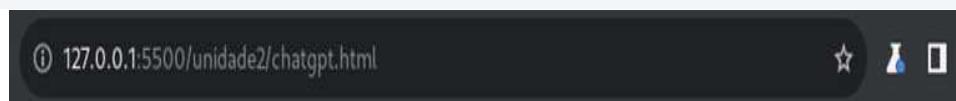
1 const CHATGPT_KEY = "sk-v3QZgqyU6y.....8ihhWeudyRK";
2
3 const consultaChat = async () => {
4     let question = document.getElementById('question').value;
5     document.getElementById('pergunta').innerHTML = question;
6     await fetch("https://api.openai.com/v1/completions", {
7         method: "POST",
8         headers: {
9             Accept: "application/json",
10            "Content-Type": "application/json",
11            Authorization: "Bearer " + CHATGPT_KEY,
12        },
13        body: JSON.stringify({
14            model: "text-davinci-002",
15            prompt: question,
16            max_tokens: 1024,
17            temperature: 0.5 //qualidade da resposta
18        }),
19    })
20    .then((response) => response.json())
21    .then((data) => {
22        //console.log(data);
23        document.getElementById('resposta').innerHTML = data.choices[0].text;
24    })
25    .catch(() => {
26        //console.log(data);
27        document.getElementById('resposta').innerHTML = "reformule a pergunta";
28    });
29}

```

Figura 3 | Código Javascript – chatgpt.js. Fonte: elaborada pelo autor.

Note que na linha um foi ocultada a chave criada para essa atividade. Você, então, deve substituir o conteúdo que está entre aspas (" ") e colocar o código da sua chave. O resultado será uma página que responderá a perguntas feitas pelo usuário, veja:

# DESENVOLVIMENTO EM JAVASCRIPT



## Pergunte ao chatGPT

Formule a pergunta

O que é javascript?

Perguntar

O que é javascript?

JavaScript é uma linguagem de programação interpretada.

Figura 4 | Resultado. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

## APIs Javascript

### 1 NAVEGADOR



Conceitos de árvore DOM e seu significado.  
Manipulação HTML e CSS com Javascript  
pelo DOM por meio de métodos como  
`getElementById()` e `querySelectorAll()`.

### 2 ÁUDIO E GRÁFICO



Animações e áudio em projetos web utilizando  
o elemento canvas e manipulação desse  
elemento com Javascript.

### 3 COMUNICAÇÃO



Recursos como Ajax e a propriedade  
`XMLHttpRequest` para comunicação  
com servidores.

### 4 TERCEIROS



Principais APIs de terceiros:  
`ViaCEP` e `OpenLayers`.

Figura | APIs Javascript. Fonte: elaborada pelo autor.

ALVES, W. P. **HTML e CSS**: aprenda como construir páginas web. São Paulo: Expressa, 2021.

# DESENVOLVIMENTO EM JAVASCRIPT

FLANAGAN, D. **Javascript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

OPENAI. **API reference**. 2023. Disponível em: <https://platform.openai.com/docs/api-reference/introduction>. Acesso em: 12 jan. 2024.

## Unidade 3

### Programação Orientada a Eventos

#### Aula 1

Introdução a eventos

#### Introdução a eventos



##### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

#### Ponto de Partida

Iniciamos, agora, a unidade na qual falaremos bastante sobre eventos na programação web. De maneira bem simples, um evento nada mais é do que a manifestação de algo que aconteceu. No contexto do desenvolvimento de aplicações web, é natural utilizarmos os eventos, pois são eles, de certa forma, que tornam a aplicação interativa, como uma ação que é disparada com o movimento do mouse.

O desafio para esta aula é utilizar o Javascript para realizar uma ação de verificar se os campos de uma tela de login (e-mail e senha) estão preenchidos; se não estiverem, deverá aparecer uma

# DESENVOLVIMENTO EM JAVASCRIPT

mensagem informando ao usuário que os campos não estão preenchidos. Trate, também, o oposto disso.

Ao longo das aulas, você, certamente, identificará a existência desse tipo de recurso a partir da sua experiência de usuário. Esperamos que goste do que vai encontrar nesta unidade, pois enriquecerá a sua prática no desenvolvimento de aplicações web. Que tal dominar a programação orientada a eventos?

Bons estudos!

## Vamos Começar!

### Definindo eventos

Eventos são ações ou ocorrências que acontecem no sistema que estamos desenvolvendo, e este, por sua vez, nos alerta sobre essas ações, para que possamos responder de alguma forma, se desejado (MDN, 2023).

Para lidar com esse tipo de situação, é possível realizar o tratamento de um evento, que pode ser feito com uma função Javascript que registramos no navegador, que chama quando ocorre algum tipo de evento específico (Flanagan, 2013).

Pode-se, ainda, com o uso de programação orientada a eventos, realizar diversas tarefas que tornam a relação entre usuário e aplicação muito mais significativa. Eventos permitem aumentar ainda mais o grau de dinamismo de uma aplicação escrita para web, mas é importante estar atento para que um determinado evento seja utilizado no momento adequado.

### Relacionando eventos com uma solução tradicional

Um dos exemplos mais significativos do emprego de programação orientada a eventos está presente no dia a dia da maioria das pessoas que costumam realizar compras pela internet. Já sabe do que eu estou falando? Se você pensou no cadastro que é necessário ter em um site de ecommerce, você acertou.

Todo site de ecommerce exige que o usuário realize um cadastro com as principais informações pessoais e dados de endereço, afinal, caso o usuário efetive uma compra, faz-se necessário saber para onde o produto deverá ser encaminho, não é mesmo? Esses cadastrados precisam ser validados de alguma maneira, para melhorar a experiência do usuário, bem como garantir que as informações dadas sejam válidas. Nós, então, iniciamos pela informação mais básica: o nome. Você pode desenvolver várias formas de validação; para o exemplo demonstrado a seguir, vamos utilizar uma escolhida para ampliar o seu rol de conhecimento: a propriedade `textContent`, que torna a validação agradável ao usuário.

# DESENVOLVIMENTO EM JAVASCRIPT

Veja o código HTML simples, com um campo para inserção de nome:

```
1<!DOCTYPE html>
2<html>
3
4<head>
5    <meta charset='utf-8'>
6    <meta http-equiv='X-UA-Compatible' content='IE=edge'>
7    <title>Exercício de validação</title>
8    <link rel='stylesheet' type='text/css' media='screen' href='validanome.css'>
9</head>
10
11<body>
12    <div id="cadNome">
13        <h1>Validação</h1>
14        <input type="text" id="nome">
15        <div id="msgerro"></div>
16        <button type="button" id="botao">Verificar</button>
17    </div>
18    <script src='validanome.js'></script>
19</body>
20
21</html>
```

Figura 1 | Código HTML simples - validanome.html. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 *{
2     margin: 0;
3     padding: 0;
4     box-sizing: border-box;
5     font-family: Helvetica;
6     color: #323232;
7     /* border: none; */
8 }
9 #cadNome{
10     background-color: #fff;
11     width: 400px;
12     margin-left: auto;
13     margin-right: auto;
14     margin-top: 10vh;
15     text-align: center;
16 }
17 input{
18     border-bottom: 2px solid #323232;
19     padding: 10px;
20     font-size: 1rem;
21     margin-bottom: 20px;
22     width: 300px;
23 }
24 button{
25     text-align: center;
26     text-transform: uppercase;
27     font-weight: bold;
28     border: none;
29     height: 40px;
30     width: 90px;
31     border-radius: 10px;
32     margin-top: 30px;
33     color: #fff;
34     background-color: blue;
35     cursor: pointer;
36 }
```

Figura 2 | validanome.css. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

Vamos para o arquivo Javascript. Nele, pegamos os campos de mensagem e o campo de nome pelos seus Ids; depois, verificamos se o tamanho do conteúdo do campo nome era igual a zero (0).

Se o tamanho do conteúdo for zero, ele mostrará a mensagem “Informe o nome” em vermelho; se não for, ele exibirá um alerta dizendo “olá” mais o nome digitado. Veja:

```
1 var msg = document.getElementById('msgerro');
2 var nome = document.getElementById('nome');
3
4 const validaNome = () =>{
5     if(nome.value.length == 0){
6         msg.style.color = 'red';
7         msg.textContent = 'Informe o nome';
8     }else{
9         msg.textContent = "";
10        alert('Olá ' + nome.value);
11    }
12 }
13 document.getElementById('botao').addEventListener('click', validaNome);
```

Figura 4 | Arquivo Javascript – validanome.js. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

## Validação

Informe o nome

VERIFICAR

Figura 5 | Resultado sem nome. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

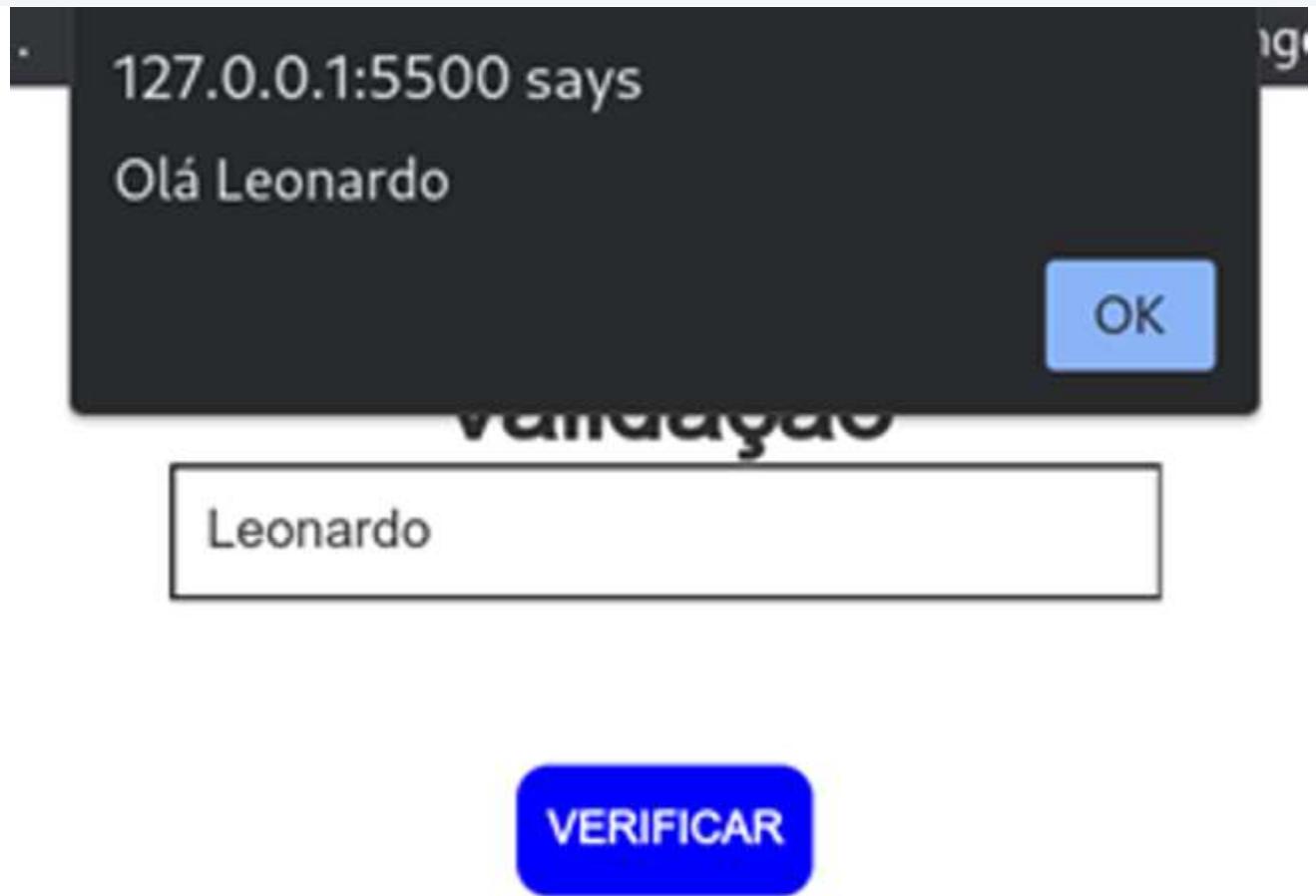


Figura 6 | Resultado com nome  
Fonte: elaborada pelo autor.

Como você pode ver, o Javascript cumpre bem uma das principais qualidades que o tornam indispensável na programação web: sua utilização para verificação e validação local dos dados de um formulário, evitando a realização de requisições desnecessárias ao servidor web e melhorando drasticamente o desempenho da página (Oliveira, 2020).

**Siga em Frente...**

## Tipos de eventos

Com o avanço da internet e de todas as tecnologias relacionadas a ela, os eventos, que antes eram limitados, passaram a ser variados. Uma das tecnologias mais presentes em nossas vidas, o aparelho celular, por sua característica de uso baseado em toque, exigiu a definição de novos tipos de eventos de toque e gesto.

# DESENVOLVIMENTO EM JAVASCRIPT

Segundo Flanagan (2013), os eventos, portanto, podem ser agrupados por categoria: eventos de entrada dependentes de dispositivo, eventos de entrada independentes de dispositivo, eventos de interface com o usuário, eventos de mudança de estado, eventos específicos da API e rotinas de tratamento de cronômetros e erro. No entanto, ainda segundo o autor, existem os tipos legados, ou seja, aqueles que já existem há mais tempo e que são universalmente suportados. Os principais são:

- Eventos de formulário.
- Eventos de janela.
- Eventos de mouse.
- Eventos de teclado.

Entre os eventos legados mais utilizados, estão os eventos de formulário, pois este é um dos recursos mais utilizados no desenvolvimento de aplicações web. Neles, existem diversos elementos que podem ser afetados por eventos, como botões, campos de texto e botões de opção. O evento que geralmente é utilizado é o evento submit, que serve para realizar persistência em bases de dados, e vale a pena saber mais sobre os eventos legados. Confira o [\*\*Saiba mais\*\*](#) e veja as sugestões de leitura sobre eventos.

## Vamos Exercitar?

Um dos problemas mais comuns encontrados e resolvidos com a utilização de eventos diz respeito à validação de campos de formulários. Esses formulários, por sua vez, podem ser diversos, logo, é muito importante que você compreenda como utilizar a programação orientada a eventos para lidar com isso.

Para pôr em prática tudo que aprendemos, vamos construir uma pequena aplicação de tela de login para, com o auxílio de eventos, realizar a validação dos campos dessa tela. A proposta é simples: verificar se os campos foram preenchidos e informar o usuário para que faça o correto preenchimento ou, dar os parabéns pelo preenchimento feito. Veja como ficará nosso código HTML e CSS:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 <!DOCTYPE html>
2 <html>
3 <head>
4     <meta charset='utf-8'>
5     <meta http-equiv='X-UA-Compatible' content='IE=edge'>
6     <title>Login</title>
7     <link rel="stylesheet" href="style.css">
8     <meta name='viewport' content='width=device-width, initial-scale=1'>
9 </head>
10 <body>
11     <div id="login-container">
12         <h1>Login</h1>
13         <form action="">
14             <label for="email">E-mail</label>
15             <input type="email" name="email" id="email">
16             <label for="senha">Senha</label>
17             <input type="password" name="senha" id="senha">
18             <a href="#" id="forgot-pass">Esqueceu a senha?</a>
19             <button type="button" id="button">Login</button>
20         </form>
21     </div>
22     <script src="login.js"></script>
23 </body>
24 </html>
```

Figura 7 | Códigos - login.html. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 *{
2     margin: 0;
3     padding: 0;
4     box-sizing: border-box;
5     font-family: Helvetica;
6     color: #323232;
7     border: none;
8 }
9
10 a{
11     font-size: .8rem;
12 }
13
14 a:hover{
15     color: blue;
16 }
17
18 /*container login*/
19 #login-container{
20     background-color: #fff;
21     width: 400px;
22     margin-left: auto;
23     margin-right: auto;
24     padding: 20px 30px;
25     margin-top: 10vh;
26     text-align: center;
27 }
28
29 /*formulario*/
30 form{
31     margin-top: 30px;
32     margin-bottom: 40px;
33 }
```

Figura 8 | Códigos CSS - login.css – parte 1. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
34 label, input{  
35     display: block;  
36     width: 100%;  
37     text-align: left;  
38 }  
39 label{  
40     font-weight: bold;  
41     font-size: .8rem;  
42 }  
43 input{  
44     border-bottom: 2px solid #323232;  
45     padding: 10px;  
46     font-size: 1rem;  
47     margin-bottom: 20px;  
48 }  
49 #forgot-pass{  
50     text-align: right;  
51     display: block;  
52 }  
53 button{  
54     text-align: center;  
55     text-transform: uppercase;  
56     font-weight: bold;  
57     border: none;  
58     height: 40px;  
59     width: 90px;  
60     border-radius: 10px;  
61     margin-top: 30px;  
62     color: #fff;  
63     background-color: blue;  
64     cursor: pointer;  
65 }
```

Figura 9 | Códigos CSS - login.css – parte 2. Fonte: elaborada pelo autor.

Agora, vejamos o código Javascript que fará o trabalho proposto no problema em estudo:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 var button = document.getElementById('button');
2 var email = document.getElementById('email');
3 var senha = document.getElementById('senha');
4
5 button.addEventListener("click", function () {
6     if (email.value == '' || senha.value == '') {
7         alert("Campo e-mail ou senha não preenchidos");
8     } else {
9         alert("Campos preenchidos com sucesso");
10    }
11});
```

Figura 10 | login.js. Fonte: elaborada pelo autor.

As três primeiras linhas servem para coletar os elementos por seus respectivos Ids; depois, adiciona-se um evento ao botão e, dentro do evento, uma função que verificará se os campos de e-mail e senha foram informados. Caso um dos campos não seja informado, aparecerá uma caixa de diálogo com uma mensagem informando ao usuário que os campos e-mail e senha devem ser preenchidos. Caso eles tenham sido preenchidos, aparecerá uma caixa de diálogo com uma mensagem informando que os campos foram preenchidos.

Perceba que essa validação pode ser feita de diversas formas. Vamos, agora, dar uma olhadinha no resultado, para ver como ficou:

# DESENVOLVIMENTO EM JAVASCRIPT

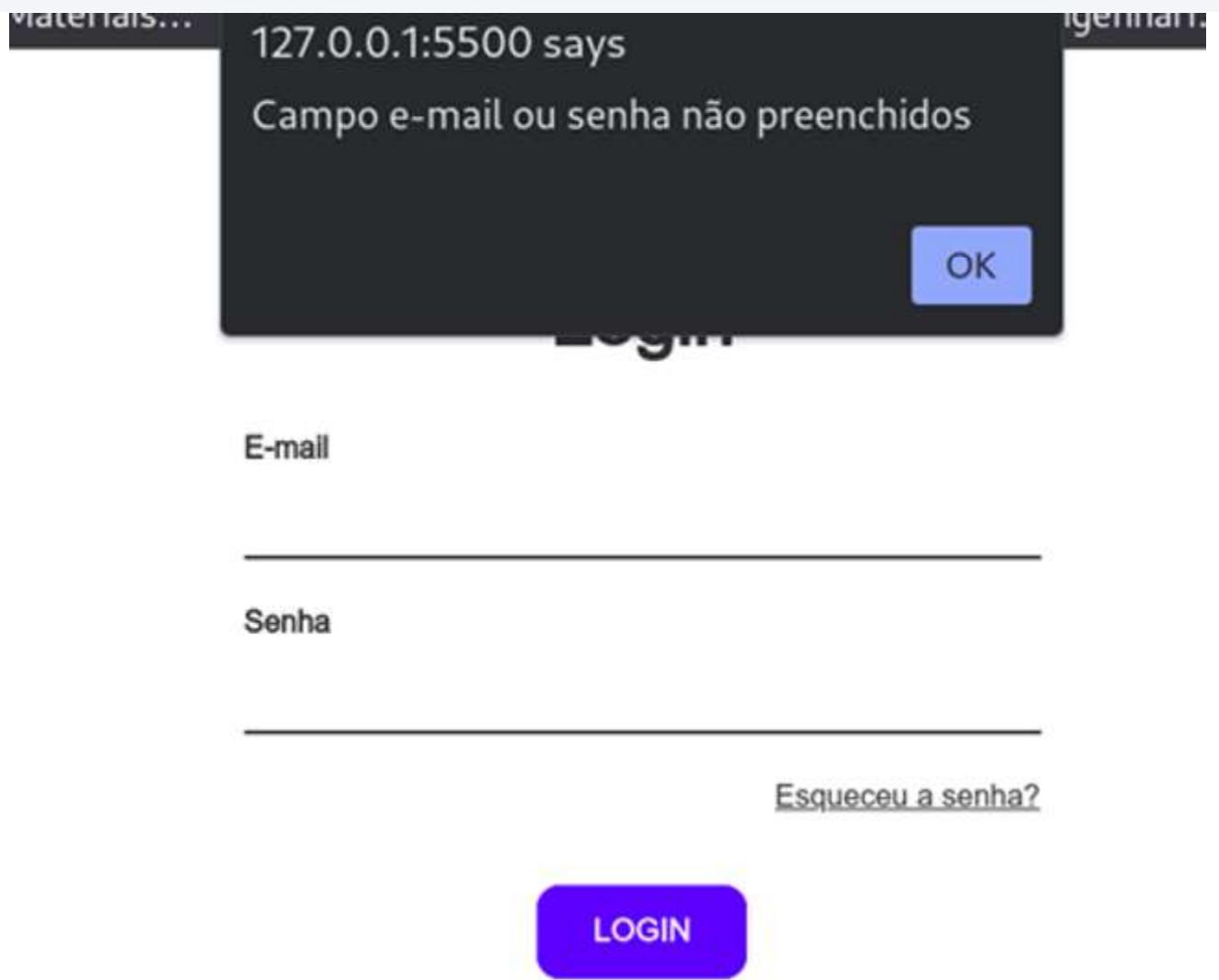


Figura 11 | Resultado da validação: e-mail ou senha não preenchidos. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

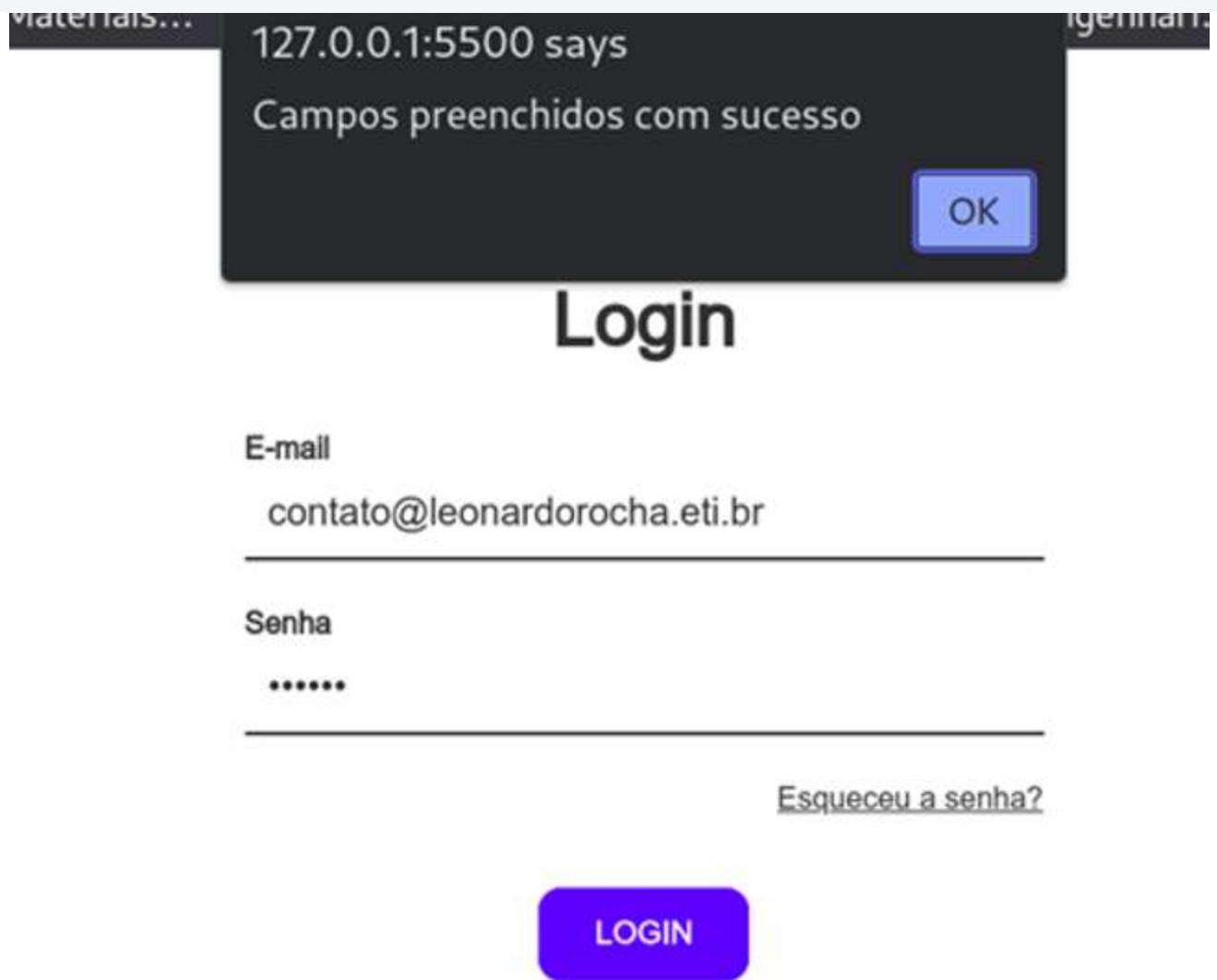


Figura 12 | Resultado da validação: e-mail e senha preenchidos. Fonte: elaborada pelo autor.

Não existe uma forma única de cumprir a tarefa de validação dos campos propostos. Muito está ligado à criatividade e lógica de quem está lidando com a situação e precisa pensar numa solução para resolvê-la. Então, pratique e exercite sua criatividade.

## Saiba mais

Sugerimos a leitura do Capítulo 17 – *Tratando eventos*, do livro *JavaScript: o guia definitivo*, que trata de eventos,

FLANAGAN, D. [JavaScript: o guia definitivo](#). Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

Leia também o artigo *Introdução a eventos*.

# DESENVOLVIMENTO EM JAVASCRIPT

MDN WEB DOCS. [Introdução a eventos](#). 2023.

## Referências

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. [Introdução a eventos](#). 2023. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Building\\_blocks/Events](https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Building_blocks/Events). Acesso em: 12 jan. 2024.

OLIVEIRA, C. L. V.; ZANETTI, H. A. P. **Javascript descomplicado**: programação para web, IOT e dispositivos móveis. São Paulo: Érica, 2020.

## Aula 2

Eventos

### Eventos



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

Olá, estudante, nós daremos continuidade ao estudo de eventos. Como já vimos, os eventos oportunizam o emprego de recursos de interatividade, uma vez que permitem o tratamento de qualquer mudança ou comportamento sofrido no documento web. Por exemplo, com eles, é possível, dar foco a um determinado campo.

# DESENVOLVIMENTO EM JAVASCRIPT

Agora, vamos conhecer eventos que nos permitem observar mudanças e, a partir disso, nos possibilita aplicar ações sobre essas mudanças, com o objetivo de melhorar a experiência do usuário no uso da aplicação web.

A fim de entender melhor o tópico central desta aula, você criará uma aplicação para pessoas que estão sendo alfabetizadas, para que, utilizando a sua aplicação, possam conhecer melhor o alfabeto. Desenvolva um protótipo com as seis primeiras letras, de A à F. O funcionamento é simples: a pessoa clicará em uma letra e, um campo com essa letra receberá foco, para que a pessoa possa redigir uma palavra que inicie com a letra escolhida. Ao escolher a letra, ela sumirá da lista de letras disponíveis, evitando confusão no processo de aprendizagem.

Você nunca sabe que resultados virão da sua ação. Mas se você não fizer nada, não existirão resultados (Mahatma Gandhi).

Bons estudos!

## Vamos Começar!

### Eventos de foco e desfoco

A utilização de recurso de foco e desfoco tem um propósito muito óbvio. Quando pensado a partir do contexto do desenvolvimento de aplicações web, pode ser empregado para as mais variadas possibilidades. Você pode pensar na utilização do foco de um dado campo de formulário para chamar a atenção do usuário, evitando, assim, que ele faça o uso incorreto da aplicação, ou ainda, pensar no público da terceira idade, uma vez que o uso do foco e desfoco pode facilitar a aplicação, mostrando um percurso a ser seguido pela pessoa que está utilizando o aplicativo web. Veja um exemplo de uso de foco e desfoco numa aplicação formulário de contato presente em vários sites na internet:

# DESENVOLVIMENTO EM JAVASCRIPT

**Contate-nos**

**Nome**

Seu nome

**e-mail**

Seu e-mail

**Mensagem**

Sua mensagem

**Enviar**

Figura 1 | Exemplo de uso de foco e desfoco em formulário. Fonte: elaborada pelo autor.

Note que o campo nome aparece com foco, quando sofre a ação do clique do mouse, e isso favorece a compreensão do fluxo a ser seguido no formulário.

Quando se fala da ação de desfocar, ou seja, perder o foco, é possível utilizar recursos que executem uma ação. Por exemplo, é possível utilizar o evento `onblur`, que é uma rotina de tratamento de evento que executa uma ação quando um elemento perde o foco de entrada (Flanagan, 2013).

## Siga em Frente...

## Eventos de interface do usuário

Define-se Interface como o meio pelo qual o usuário interage com o computador (Barreto; Barboza, 2018), por esse motivo, considera-se a linguagem Javascript com sendo a responsável por manter a interação da interface com o usuário de forma perceptiva e agradável, afinal de contas, devemos pensar sempre na experiência do usuário. Alguns dos eventos mais utilizados e conhecidos estão:

Evento	Local de uso
--------	--------------

# DESENVOLVIMENTO EM JAVASCRIPT

focus	Em formulários e eventos Window (janela).
change	É disparado quando o usuário modifica o valor de um elemento, como <input> e <select>.
blur	Remove o foco de teclado do elemento atual
submit	Utilizando quando se submete um formulário

Quadro 1 | Local de uso de eventos. Fonte: adaptado de MDN (2023).

Uma área em que a necessidade de eventos cresceu muito é a de interfaces de dispositivos móveis. Por esse motivo, alguns eventos foram e são criados para atender às demandas desse meio de interação, e alguns eventos touchscreen são mapeados nos tipos de eventos tradicionais, como o click e o scroll (Flanagan, 2013). Por exemplo, a Apple criou eventos de gesto de mudança de escala e rotação feitos com dois dedos.

Evento	Utilidade
gesturestart	É disparado quando um gesto começa.
gestureend	É disparado quando o gesto termina.
gesturechange	Usado para monitorar o progresso do gesto.
touchstart	Disparado quando o dedo toca na tela.
touchmove	Disparado quando o dedo se move.
touchend	Disparado quando o dedo é tirado da tela.

Quadro 2 | Eventos. Fonte: adaptado de Flanagan (2013).

## Mutation event e mutation observers

*Mutation event* é uma abordagem considerada inicial para se detectar e responder às alterações do DOM. Graças a ela, foi possível configurar os chamados ouvintes de eventos específicos de modificação, como inserção de elementos, alteração de atributos ou remoção de nós, no entanto, apresentava alguns problemas de inconsistência entre navegadores. Um exemplo é o método `addEventListener()`.

Diante desses problemas do *Mutation event*, foi criado o *Mutation observer*, que é uma API JavaScript global que fornece aos desenvolvedores uma maneira de reagir às modificações do DOM ou tratá-las (Toreini, 2019). Considerado um construtor que serve para instanciar novos observadores de mutação do DOM, ele conta com vários métodos e propriedades úteis na construção de código.

## Vamos Exercitar?

Ao longo da aula, nós compreendemos um pouco mais os eventos, seus tipos, aplicação e conhecemos detalhes importantes que farão toda a diferença em nossa prática enquanto pessoas responsáveis pela criação de soluções em aplicativos web.

# DESENVOLVIMENTO EM JAVASCRIPT

Agora, vamos colocar em prática um pouco esse conhecimento e construir a aplicação que será utilizada para pessoas que estão em processo de alfabetização, conforme foi apresentada no início da aula. Se você percebeu, falamos muito sobre interface e sua relevância na experiência do usuário, portanto, não podemos deixar de pensar na usabilidade de nossa aplicação. Para facilitar esta etapa, vamos utilizar o Bootstrap 5 (ou superior) como framework para a parte visual do site, ou seja, estilo com CSS. A interface de nosso site deverá ficar assim:

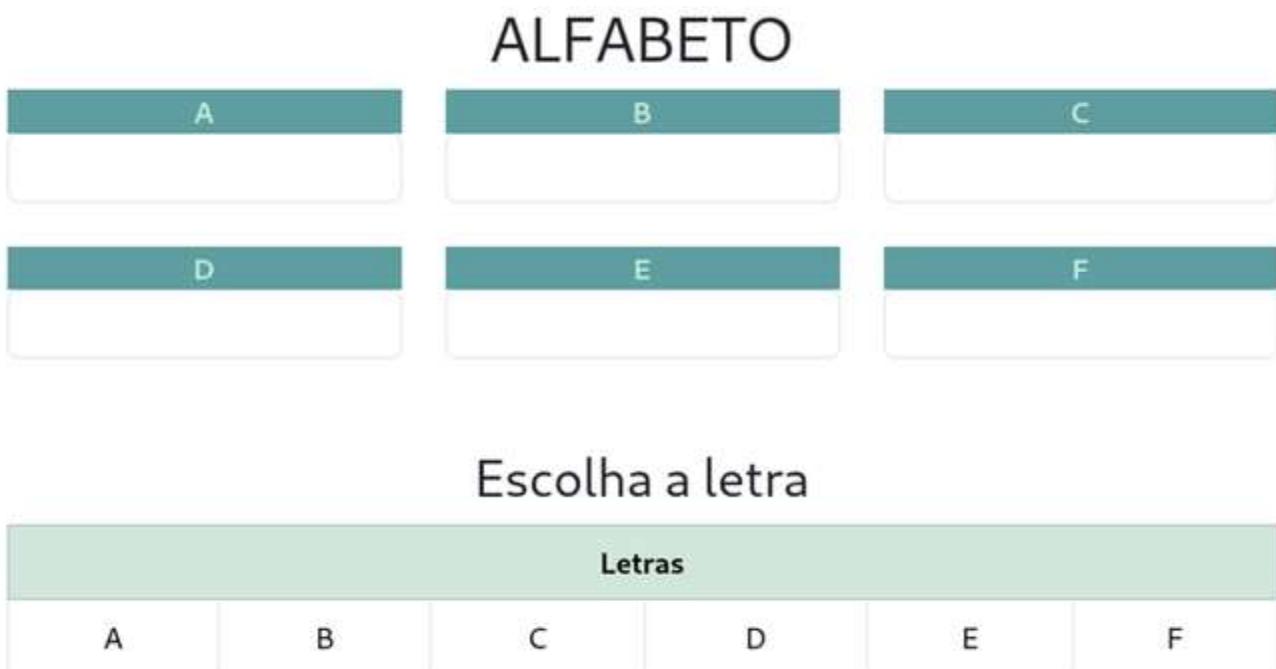


Figura 2 | Interface do site. Fonte: elaborada pelo autor.

Para esse projeto, foram escolhidos tons de verde, e como se trata de um protótipo, em baixo, ficaram 6 letras, de A à F. Ao clicar sobre uma delas, a letra some e o campo referente à letra é colocado em foco, com a cor de fundo alterada. Por exemplo, quando a letra D é clicada, isso acontece:

# DESENVOLVIMENTO EM JAVASCRIPT

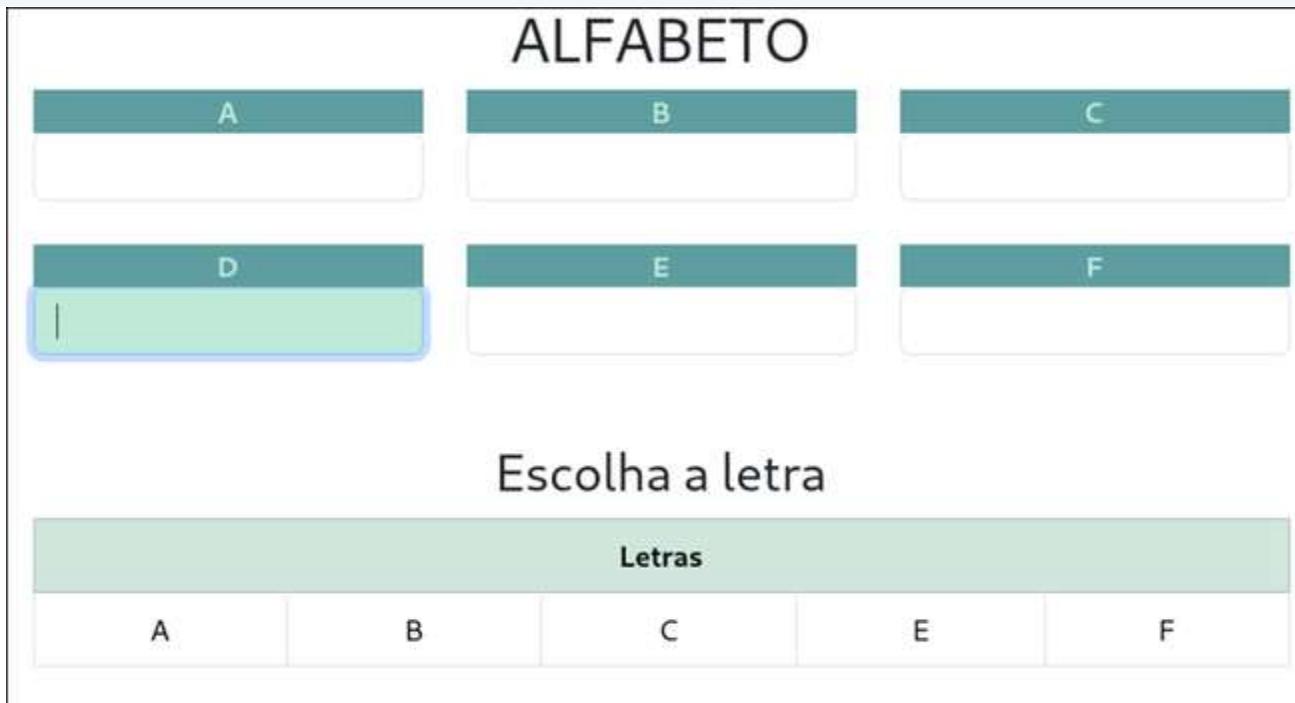


Figura 3 | Teste com a letra D. Fonte: elaborada pelo autor.

O campo fica com foco e muda de cor para que a pessoa que utilizar a aplicação saiba qual letra escolheu e onde digitar a palavra que se inicia com aquela letra, como mencionado no escopo da proposta. Agora, vamos ver o código para a criação dessa aplicação:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5     <meta charset='utf-8'>
6     <title>Seleção de letras</title>
7     <link rel='stylesheet' type='text/css' media='screen'
 href='selecionaletra.css'>
8     <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/
css/bootstrap.min.css" rel="stylesheet"
      integrity="sha384-
T3c6CoIi6uLrA9TneNEoa7RxnatjcdSCmG1MXxSR1GAsXEV/Dwwykc2MPK8M2HN"
      crossorigin="anonymous">
9
10 </head>
11
12 <body>
13     <div class="container">
14         <!-- <form> -->
15         <h1>ALFABETO</h1>
16         <div class="row">
17             <div class="col">
18                 <label for="B">A</label>
19                 <input type="text" class="form-control" id="letraA">
20             </div>
21             <div class="col">
22                 <label for="A">B</label>
23                 <input type="text" class="form-control" id="letraB">
24             </div>
25             <div class="col">
26                 <label for="C">C</label>
27                 <input type="text" class="form-control" id="letraC">
28             </div>
```

Figura 4 | Código de criação da aplicação – escolheletra.html – parte 1. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
29 </div>
30 <br>
31 <div class="row">
32   <div class="col">
33     <label for="D">D</label>
34     <input type="text" class="form-control" id="letraD">
35   </div>
36   <div class="col">
37     <label for="E">E</label>
38     <input type="text" class="form-control" id="letraE">
39   </div>
40   <div class="col">
41     <label for="F">F</label>
42     <input type="text" class="form-control" id="letraF">
43   </div>
44 </div>
45 <!-- <button type="button" onclick="teste()"></button> -->
46 <!-- </form> -->
47 <br><br>
48 <h2 class="text-center">Escolha a letra</h2>
49 <table class="table table-bordered" id="table">
50   <thead>
51     <tr class="table-success text-center">
52       <th colspan="6" scope="row">Letras</th>
53     </tr>
54   </thead>
```

Figura 5 | Código de criação da aplicação – escolheletra.html – parte 2. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
55      <tbody>
56          <tr class="text-center">
57              <td id="A">A</td>
58              <td id="B">B</td>
59              <td id="C">C</td>
60              <td id="D">D</td>
61              <td id="E">E</td>
62              <td id="F">F</td>
63      </tr>
64  </tbody>
65 </table>
66 </div>
67 <script src='selecionaletra.js'></script>
68 </body>
69
70 </html>
```

Figura 6 | Código de criação da aplicação – escolheletra.html – parte 3. Fonte: elaborada pelo autor.

Note que na linha 8 foi inserido o caminho CDN para o uso do Bootstrap. Na parte de cima da página, construímos os títulos e campos em que o usuário deverá digitar alguma palavra que comece com a letra escolhida; abaixo, aparecerá uma tabela para que o usuário possa clicar e escolher a letra que desejar. Foi utilizado, também, um CSS simples no elemento h, para colocar cor de fundo nos elementos label utilizados no projeto. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 h1{  
2     text-align: center;  
3 }  
4 label{  
5     background-color: cadetblue;  
6     color: rgb(191, 234, 215);  
7     font-weight: bold;  
8     text-align: center;  
9     width: 100%;  
10 }
```

Figura 7 | Código CSS. Fonte: elaborada pelo autor.

Agora, vamos observar o Javascript utilizado para esse projeto:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 var letraA = document.getElementById('letraA');
2 var letraB = document.getElementById('letraB');
3 var letraC = document.getElementById('letraC');
4 var letraD = document.getElementById('letraD');
5 var letraE = document.getElementById('letraE');
6 var letraF = document.getElementById('letraF');
7
8 var table = document.getElementById('table');
9
10 table.addEventListener("click", function (event) {
11     if (event.target.id == "A") {
12         letraA.style.backgroundColor = 'rgb(191, 234, 215)';
13         letraA.focus();
14     }
15     if (event.target.id == "B") {
16         letraB.style.backgroundColor = 'rgb(191, 234, 215)';
17         letraB.focus();
18     }
19     if (event.target.id == "C") {
20         letraC.style.backgroundColor = 'rgb(191, 234, 215)';
21         letraC.focus();
22     }
23     if (event.target.id == "D") {
24         letraD.style.backgroundColor = 'rgb(191, 234, 215)';
25         letraD.focus();
26     }
27 }
```

Figura 8 | selecioneletra.js - Parte 1. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
28     if (event.target.id == "E") {
29         letraE.style.backgroundColor = 'rgb(191, 234, 215)';
30         letraE.focus();
31     }
32     if (event.target.id == "F") {
33         letraF.style.backgroundColor = 'rgb(191, 234, 215)';
34         letraF.focus();
35     }
36     setTimeout(function () {
37         event.target.remove();
38     }, 500);
39
40});
```

Figura 9 | selecioneletra.js - Parte 2. Fonte: elaborada pelo autor.

O código coleta os elementos dos campos em que o usuário digitar algo pelo ID, obtém o elemento da tabela e, utilizando o método addEventListener(), verifica qual elemento está sendo acionado pelo click do mouse; com isso, ele coloca o campo do tipo input em foco e modifica sua cor de fundo. Simples, não é?

## Saiba mais

Sugerimos o capítulo sobre *Projeto de interface com o usuário*, do livro *Interface humano-computador*.

BARRETO, J. dos S. et al. [Interface humano-computador](#). Porto Alegre: SAGAH, 2018.

Saiba mais sobre MutationObserver.

MDN WEB DOCS. [MutationObserver](#). 2023.

Saiba mais sobre Javascript Events.

W3SCHOOLS. [JavaScript Events](#). [s. d.]

## Referências

# DESENVOLVIMENTO EM JAVASCRIPT

BARRETO, J. dos S. et al. **Interface humano-computador**. Porto Alegre: SAGAH, 2018.

FLANAGAN, D. **JavaScript**: o guia definitivo. 6. ed. Porto Alegre: Bookman, 2013.

TOREINI, E. et al. DOMtegrity: ensuring web page integrity against malicious browser extensions. **International Journal of Information Security**, Heidelberg, v. 18, n. 6, p. 801-814, 2019.

## Aula 3

Interação com interface física e lógica

### Interação com interface física e lógica



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Partida

Olá, estudante!

Iniciamos, nesta unidade, nossos estudos de eventos, que, como já vimos em outros momentos, nos auxiliará na construção da experiência com o usuário que utilizará nossa aplicação.

Existem muitas possibilidades de aplicação dos conhecimentos de manutenção de eventos utilizando a linguagem Javascript, e para que isso fique evidente, hoje, veremos uma situação um tanto quanto inusitada para ser resolvida com a utilização de eventos.

Um tema muito importante e que frequentemente esquecemos é o da inclusão, por esse motivo, o seu desafio na aula de hoje é construir um protótipo de aplicação que capture a posição do mouse na tela à medida que ele é movido dentro de uma página web. Isso é importante pois

# DESENVOLVIMENTO EM JAVASCRIPT

apoia o desenvolvimento de projetos que consideram o público com baixa ou nenhuma visão, como público potencial para uso das aplicações desenvolvidas por você e sua equipe.

Diante disso, desenvolva seu lado profissional e fique ligado em temas como o da inclusão, pois é crescente a necessidade de profissionais no mercado com essa sensibilidade.

Bons estudos!

## Vamos Começar!

### Eventos de formulário

Antes de qualquer coisa, tenha em mente que os eventos utilizados para formulários são aqueles que, de alguma forma, destinam-se às interações com elementos de um formulário. No entanto, não é tão dividido assim!

Podemos pensar em evento aplicado a formulário iniciando com um exemplo bem simples. A validação de campos de Nome e sobrenome. Veja esse fragmento de código:

```
1 var form = document.querySelector("form");
2 var nome = document.getElementById("nome");
3 var sobrenome = document.getElementById("sobrenome");
4 var submit = document.getElementById("submit");
5 var para = document.querySelector("p");
6
7 form.onsubmit = function (e) {
8     if (nome.value === "" || sobrenome.value === "") {
9         e.preventDefault();
10        para.textContent = "Você precisa preencher os dois campos";
11    }
12};
```

Figura 1 | Fragmento de código. Fonte: adaptada de MDN (2023).

A validação de formulários é algo indispensável, e utilizar o Javascript é crucial para se evitar problemas com dados informados pelos usuários (Miletto; Bertagnolli, 2014). Note que, na linha 7, o evento utilizado no fragmento de código apresentado foi o onsubmit, típico de formulário cujos dados são enviados.

Outros exemplos de evento possível de se utilizar em formulário são o evento formdata e o evento reset. O primeiro é acionado após a construção da lista de entradas que representa os dados do formulário, e isso acontece quando o formulário é enviado, mas também pode ser

# DESENVOLVIMENTO EM JAVASCRIPT

acionado pela invocação de um construtor FormData() (MDN, 2023); já o segundo é acionado quando um formulário é redefinido.

## Siga em Frente...

### Eventos de mouse e teclado

Eventos de mouse e teclado são variados e podem e devem ser empregados em situações de uso adequado. Os eventos de mouse podem ser utilizados nas mais variadas situações, como a captura de movimento do mouse ou a ação de clique dele. Além disso, há que se considerar que existem eventos que são independentes de dispositivos e existem aqueles que são dependentes, como é o caso do onmousedown e do onmouseover.

Quando o assunto é acessibilidade, faz-se importante utilizar os eventos que são independentes, como onfocus e onchange (Flanagan, 2013).

Vejamos os eventos de mouse no quadro a seguir:

Evento	Ocorre quando
onclick	O usuário clica em um elemento.
ondblclick	O usuário clica duas vezes em um elemento.
onmousedown	O usuário pressiona o botão do mouse sobre um elemento.
onmousemove	O ponteiro do mouse está em movimento na tela.
onmouseover	O ponteiro do mouse está sobre o elemento.
onmouseout	O usuário move o ponteiro do mouse para fora do elemento.
onmouseup	O usuário solta o botão do mouse sobre um elemento.

Quadro 1 | Eventos de mouse. Fonte: adaptado de Miletto e Bertagnolli (2014).

Os eventos de teclado são poucos, mas muito úteis em muitas situações. Apesar de poucas opções, os eventos de teclado, geralmente associados a outros eventos e recursos, podem significar uma excelente estratégia tecnológica. Os principais eventos de teclado são:

Evento	Ocorre quando
onKeyDown	O usuário está pressionando uma tecla.
onKeyPress	O usuário pressiona uma tecla.
onKeyUp	O usuário solta a tecla (previamente pressionada).

# DESENVOLVIMENTO EM JAVASCRIPT

Quadro 2 | Principais eventos de teclado. Fonte: adaptado de Miletto e Bertagnolli (2014).

Há uma sutil diferença entre keyPress e keyDown; o último é disparado para teclas que produzem e que não produzem um caractere.

## Eventos mobile

Os eventos do toque surgiram com a finalidade de dar suporte de qualidade para interfaces baseadas em toque. Eventos dessa natureza interpretam a atividade em telas sensíveis ao toque (MDN Web Docs, 2023), e algumas definições são relevantes para se compreender o funcionamento desse tipo de evento. Primeiro, é importante que haja a superfície sensível ao toque, e pode ser uma tela ou trackpad; depois, temos o ponto de contato com a superfície, geralmente, feita com o dedo das mãos, e só então temos os eventos atuando na aplicação.

Veja alguns exemplos a seguir:

Evento	Ocorre quando
ontouchcancel	O toque é interrompido.
ontouchend	O dedo é removido da tela.
ontouchmove	O dedo se move na tela.
ontouchstart	O dedo entra em contato com a tela.

Quadro 3 | Exemplos de eventos de toque. Fonte: adaptado de W3C (2023).

Você deve ter percebido quão ricas são as opções que nós, pessoas que escrevem códigos de programa, temos ao nosso dispor. Por esse motivo, é muito importante conhecer minimamente essas opções, estudá-las e aplicá-las, para saber, num momento de criação de uma solução web, que essas opções podem ser utilizadas. Por isso, leia todo o material recomendado, pois só assim você enriquecerá o seu vocabulário de programação.

## Vamos Exercitar?

Como vimos no início da aula, o desafio é criar um protótipo de aplicação que obtenha, da área de navegação de um browser, a posição em que o mouse se encontra. Note que não é importante implementar algo complexo; esse trabalho será a base para o desenvolvimento de aplicações cujo propósito será disponibilizar ferramentas de acessibilidade e inclusão de pessoas com baixa ou nenhuma visão.

Vamos iniciar o código construindo uma página html que contenha uma div que guardará as posições X e Y da tela, por onde o ponteiro do mouse transitará. O código HTML ficará da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5     <meta charset='utf-8'>
6     <title>Posição do mouse</title>
7     <link rel='stylesheet' type='text/css' media='screen' href='movmouse.css'>
8 </head>
9
10 <body>
11     <h2>Posição do mouse</h2>
12     <br>
13     <div id="position">
14         <p>x: <span id="posicaoX"></span></p>
15         <p>y: <span id="posicaoY"></span></p>
16     </div>
17     <script src='movmouse.js'></script>
18 </body>
19
20 </html>
```

Figura 2 | Código HTML. Fonte: elaborada pelo autor.

Note que se trata de um projeto simples, que mostra um título na página que diz “Posição do mouse” e cria uma div, em que os dados de posição serão apresentados. O CSS dá conta somente da cor da página e dos dados que serão exibidos. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 *{  
2     margin: 0;  
3     padding: 0;  
4 }  
5 #position{  
6     width: 100px;  
7     height: 40px;  
8     color: rgb(0, 0, 0);  
9     padding: 5px;  
10    position: fixed;  
11    top: 0;  
12    left: 0;  
13 }  
14 body{  
15     background-color: bisque;  
16 }  
17 p{  
18     margin: 0;  
19     font-family: sans-serif;  
20 }  
21 h2{  
22     text-align: center;  
23 }  
24 }
```

Figura 3 | Código CSS. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

Agora, temos o código Javascript para a solução do problema apresentado. Para tanto, utilizaremos o evento de movimento do mouse, chamado mousemove; com ele, capturaremos os valores das posições x e y de nossa tela. O código ficará assim:

```
1 window.addEventListener('mousemove', (event) => {  
2  
3     const localiza = document.getElementById('position')  
4  
5     let posicaoX = document.getElementById('posicaoX')  
6     let posicaoY = document.getElementById('posicaoY')  
7  
8     localiza.style.top = event.clientY + 'px'  
9     localiza.style.left = event.clientX + (5) + 'px'  
10  
11    posicaoY.innerText = event.clientY + 'px'  
12    posicaoX.innerText = event.clientX + 'px'  
13  
14}  
15});
```

Figura 4 | Código Javascript. Fonte: elaborada pelo autor.

O resultado do projeto fica bem interessante, com os dados acompanhando o ponteiro do mouse à medida que você o movimento dentro do navegador.

# DESENVOLVIMENTO EM JAVASCRIPT

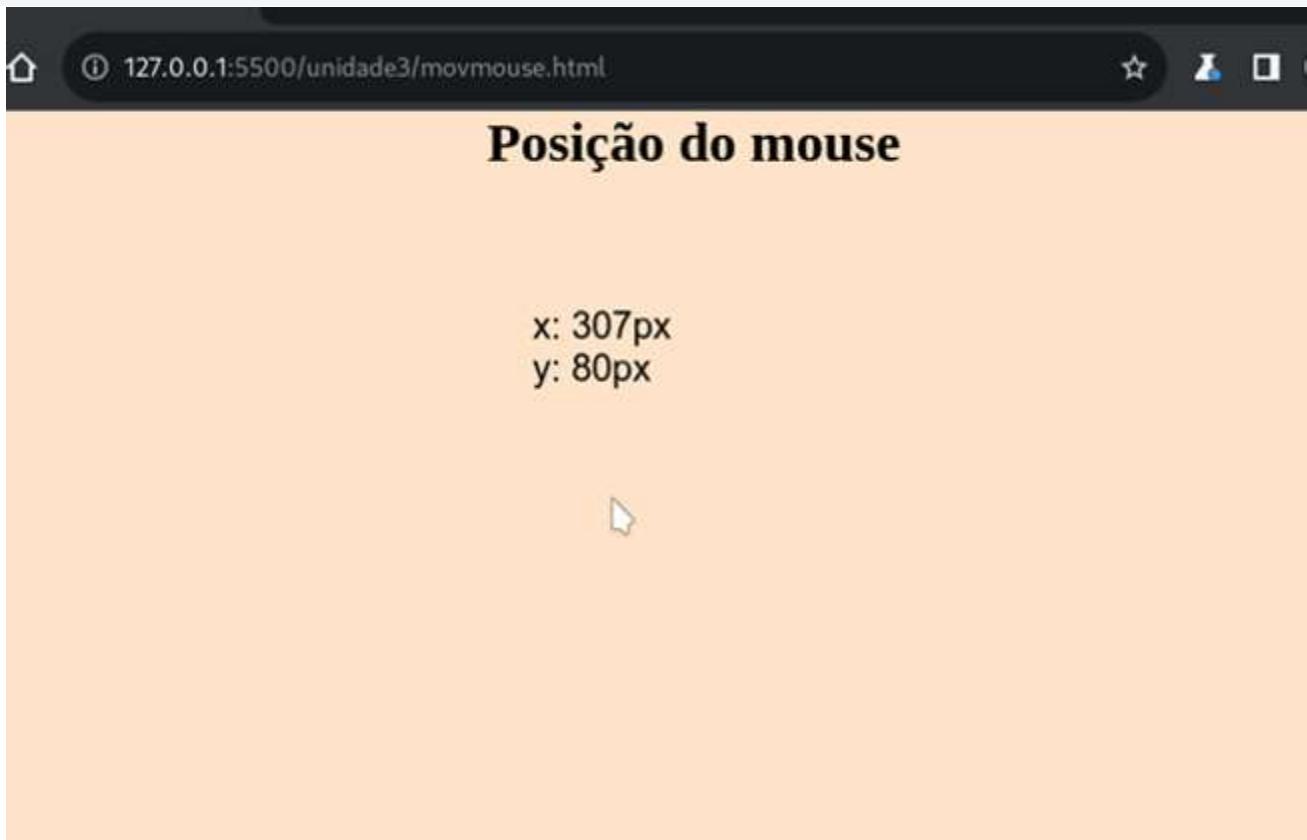


Figura 5 | Resultado do projeto. Fonte: elaborada pelo autor.

## Saiba mais

Sugerimos a leitura do Capítulo 5 – *Comportamento com JavaScript*, do livro *Desenvolvimento de software II: introdução ao desenvolvimento web com HTML, CSS, javascript e PHP*.

MILETTO, E. M.; BERTAGNOLLI, S. C. [Desenvolvimento de software II: introdução ao desenvolvimento web com HTML, CSS, javascript e PHP](#). Porto Alegre: Bookman, 2014.

Assista ao tutorial sobre o uso do evento onclick com HTML e Javascript.

CHRIS, K. [Tutorial sobre button onclick em HTML e evento de clique em JavaScript](#). Traduzido por Daniel Rosa, 2021.

Sugerimos, também, a leitura do artigo *Desenvolvimento para Dispositivos Móveis usando Tecnologias Web com Ênfase em Jogos*, que trata do uso de tecnologias web para desenvolvimento de jogos.

SANTANCHÈ, A. et al. [Desenvolvimento para Dispositivos Móveis usando Tecnologias Web com Ênfase em Jogos](#). 2013.

# DESENVOLVIMENTO EM JAVASCRIPT

E, por fim, encontre mais informações sobre eventos de toque no link.

W3SCHOOL. [HTML DOM TouchEvent](#). [s. d.]

## Referências

FLANAGAN, D. **JavaScript**: o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MDN WEB DOCS. **Introdução a eventos**. 2023. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Building\\_blocks/Events](https://developer.mozilla.org/pt-BR/docs/Learn/JavaScript/Building_blocks/Events). Acesso em: 15 jan. 2024.

MILETTO, E. M.; BERTAGNOLLI, S. C. **Desenvolvimento de software II**: introdução ao desenvolvimento web com HTML, CSS, javascript e PHP. Porto Alegre: Bookman, 2014.

W3C SCHOOL. **HTML DOM touchEvent**. [s. d.]. Disponível em: [https://www.w3schools.com/jsref/obj\\_touchevent.asp](https://www.w3schools.com/jsref/obj_touchevent.asp). Acesso em: 15 jan. 2024.

## Aula 4

Eventos em nível de página

### Eventos em nível de página



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

# DESENVOLVIMENTO EM JAVASCRIPT

Olá, estudante! Nesta aula, falaremos um pouco da importância das técnicas e métodos que possibilitam a organização do processo de desenvolvimento de uma aplicação web. Pensar na fase do projeto do software é fundamental para assegurar a qualidade do produto que está em desenvolvimento, e conhecer brevemente algumas metodologias oportunizará um aprofundamento desse conhecimento. Você verá que é indispensável discutir esse tema, mesmo que repetidas vezes, pois é algo que nunca se dissipará; há sempre algo novo a se pensar, e a prática na utilização de técnicas de gestão de projetos favorece a qualidade, a redução de custo e o cumprimento de prazos.

Como desafio, temos um problema simples que precisa de eventos CSS e do Javascript para ser resolvido. Como é possível inserir uma animação no formulário de login, criado em aulas anteriores, para que, ao pressionar o botão Login da tela, o formulário vá para cima e suma da tela? Você tem alguma ideia de como resolver isso?

Por fim e não menos importante, abordaremos maneiras de lidar com eventos no CSS e HTML5, já que essas linguagens são utilizadas com o Javascript, mas lembre-se de que não há como programar com Javascript focado na web sem o uso das linguagens HTML e CSS. Dedique sua atenção ao estudo desse tema, uma vez que fará muito sentido na sua atuação na construção de aplicações web. Seu empenho e esforço farão diferença, portanto, aproveite ao máximo esta aula.

Bons estudos!

## Vamos Começar!

### Criando um projeto

Ao longo de tudo que já foi visto sobre Javascript, você deve ter percebido a importância de se ter métodos e processos bem definidos para trabalhar no campo da programação. A utilização de estratégias, técnicas e ferramentas no desenvolvimento de aplicações é crucial e determinante para que você consiga trabalhar num determinado intervalo de tempo e entregar o que se espera do seu esforço. Por esse motivo, vamos falar um pouco sobre projetos.

Não é a nossa intenção dedicar todo o tempo disponível ao estudo do tema, mas vamos conceituar minimamente o termo e entender sua utilização no contexto do desenvolvimento de aplicações web. Para Alves (2015), um projeto pode ser definido como um evento com um tempo de duração predefinido e temporário. Em um projeto, principalmente da área de software, costuma-se ter várias pessoas dedicadas ao seu desenvolvimento. Existem propostas diversas de metodologias para projetos, e na área do desenvolvimento de software, as metodologias ágeis têm sido muito utilizadas. O desenvolvimento ágil envolve mudanças rápidas e prioriza, às vezes, agilidade ante à qualidade (Maschietto *et. al.*, 2020). Geralmente, no desenvolvimento ágil, equipes multidisciplinares são formadas e, basicamente, dá-se prioridade à conversa em vez de documentação, diminuindo, assim, a rigidez.

# DESENVOLVIMENTO EM JAVASCRIPT

## Eventos CSS

Com a linguagem de estilo CSS, é possível elaborar animações que tornam o produto de website que está sendo desenvolvido muito mais interessante. Com essa linguagem, também é possível utilizar eventos para a criação de coisas, como animação, somente com o CSS. Vejamos alguns eventos mais conhecidos:

Evento	Descrição
animationstart	Acionado quando uma animação CSS é iniciada.
animationiteration	Disparado quando uma iteração de animação CSS termina e outra se inicia.
animationend	Acionado quando uma animação CSS é concluída.
animationcancel	Acionado quando uma animação CSS é abortada inesperadamente.

Quadro 1 | Eventos mais conhecidos. Fonte: adaptado de MDN Web Docs (2023).

Para que entenda melhor o uso desse tipo de recurso, vamos melhorar nosso projeto de tela de login, aquele que criamos em aulas anteriores. Agora, vamos utilizar uma regra CSS chamada `@keyframes`, que serve para controlar etapas intermediárias em uma sequência de animação CSS, em que é empregado o estilo por quadros-chave. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
67 /* ANIMAÇÃO DO FORMULÁRIO */
68 form{
69     animation-name: suave;
70     animation-duration: 0.7s;
71 }
72
73 @keyframes suave {
74     from{
75         opacity: 0;
76         transform: scale(0.8);
77     }
78     to{
79         opacity: 1;
80         transform: scale(1);
81     }
82 }
83
```

Figura 1 | Código CSS. Fonte: elaborada pelo autor.

Adicionamos, ao final do arquivo CSS que já havíamos construído, o código que está entre as linhas 68 e 82; chamamos a regra Keyframe e, nela, adicionamos o ponto de partida (from) e ponto de chegada (to) e definimos a opacidade e transform para criar o efeito de animação. A opacidade cria a sensação de que o formulário está surgindo e o transform modifica seu

# DESENVOLVIMENTO EM JAVASCRIPT

tamanho, aumentando-o levemente. Abra o seu projeto e veja o resultado dessa simples modificação. Não ficou mais elegante?

## Siga em Frente...

### Eventos HTML5

A linguagem HTML oferece muitos eventos, como já vimos anteriormente, e esses eventos podem ser associados à utilização de eventos em CSS e, com isso, resultar num projeto de qualidade indiscutível. De maneira muito simples, os eventos podem ser utilizados para atuar sobre a janela do documento em si, utilizando manipuladores de eventos sobre os elementos do documento ou o objeto window, que representa a própria janela aberta no browser.

Como pode ver, as possibilidades são variadas, por isso, faz-se importante conhecer mais alguns eventos indispensáveis na construção de aplicações web. Vejamos:

Evento	Chamado quando
ononline	O browser detecta conectividade e está trabalhando em modo online.
onoffline	O browser não detecta conectividade e está trabalhando em modo offline.
onresize	O evento é disparado quando a janela do navegador é redimensionada.
onload	Quando a página está carregada.
onunload	Quando a página é descarregada ou a janela é fechada.
onerror	O carregamento do recurso falhou (normalmente, por erro na rede).

Quadro 2 | Eventos indispensáveis. Fonte: adaptado de Flanagan (2013).

Vamos ver um exemplo de uso do onresize e mostrar as dimensões de largura e altura na página à medida que a janela é redimensionada. Veja o exemplo de código HTML para isso:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 <!DOCTYPE html>
2 <html lang="pt-br">
3
4 <head>
5   <meta charset="UTF-8">
6   <meta name="viewport" content="width=device-width, initial-scale=1.0">
7   <title>Evento onresize</title>
8 </head>
9
10 <body>
11   <p>Redimensione a janela do navegador para disparar o evento <code>resize</code>. </p>
12   <p>Altura da janela: <span id="altura"></span></p>
13   <p>Largura da janela: <span id="largura"></span></p>
14   <script src="eventoonresize.js"></script>
15 </body>
16
17 </html>
```

Figura 2 | Exemplo de código HTML. Fonte: adaptada de MDN (2023).

Agora, o código Javascript que vai chamar o evento onresize, veja:

```
1 const saidaAlt = document.querySelector("#altura");
2 const saidaLarg = document.querySelector("#largura");
3
4 function redimensiona() {
5   saidaAlt.textContent = window.innerHeight;
6   saidaLarg.textContent = window.innerWidth;
7 }
8
9 window.onresize = redimensiona;
```

Figura 3 | Código Javascript para chamar o evento onresize. Fonte: adaptada de MDN (2023).

Execute o código e veja como ficará o projeto direto no navegador, mas não se esqueça de redimensionar a janela do navegador para ver o resultado de uso do evento onresize.

## Vamos Exercitar?

Nós já havíamos construído um formulário de login em aulas anteriores, por isso, aproveitaremos o código já elaborado para conduzir os estudos propostos para esta aula. Aliás, vamos aproveitar a animação que já fizemos nesta aula e apenas complementá-la. A intenção, agora, é aplicar outro efeito de animação com CSS ao formulário do login, fazendo com que esse formulário suba na página, saindo dela, causando a impressão de que estamos entrando em outra área.

# DESENVOLVIMENTO EM JAVASCRIPT

Para tornar essa impressão mais realista, criaremos um arquivo html chamado site.html, que servirá para essa mudança de página. Como já tínhamos Javascript para validação do login, comentaremos esse código para verificar somente a animação que estamos fazendo, assim, evitaremos confusão com o estudo que estamos conduzindo agora. O nosso código HTML desse projeto ficará da seguinte forma:

```
1 <!DOCTYPE html>
2 <html>
3
4 <head>
5     <meta charset='utf-8'>
6     <meta http-equiv='X-UA-Compatible' content='IE=edge'>
7     <title>Login</title>
8     <link rel="stylesheet" href="login2.css">
9     <meta name='viewport' content='width=device-width, initial-scale=1'>
10 </head>
11
12 <body>
13     <div id="login-container">
14         <form action="">
15             <h1>Login</h1><br>
16             <label for="email">E-mail</label>
17             <input type="email" name="email" id="email">
18             <label for="senha">Senha</label>
19             <input type="password" name="senha" id="senha">
20             <a href="#" id="forgot-pass">Esqueceu a senha?</a>
21             <button type="button" id="button">Login</button>
22         </form>
23     </div>
24     <script src="login.js"></script>
25 </body>
26
27 </html>
```

Figura 4 | Código HTML do projeto. Fonte: elaborada pelo autor.

Agora, nosso código CSS não sofrerá alterações; aliás, nele, haverá novos códigos. Vamos construir as animações, lembrando que revisitaremos animações que já fizemos anteriormente. O código (somente a parte das animações) ficará da seguinte forma:

# DESENVOLVIMENTO EM JAVASCRIPT

```
67 /* ANIMAÇÃO DO FORMULÁRIO */
68 form{
69     animation: suave 0.7s;
70 }
71
72 @keyframes suave {
73     from{
74         opacity: 0;
75         transform: scale(0.8);
76     }
77     to{
78         opacity: 1;
79         transform: scale(1);
80     }
81 }
82
83 /* REMOVE FORMULÁRIO DA TELA AO CLICAR NO BOTÃO */
84 .oculta-form{
85     animation: top 0.5s;
86     animation-fill-mode: forwards;
87 }
88
89 @keyframes top {
90     from{
91         transform: translateY(0);
92     }
93     to{
94         transform: translateY(-100vh);
95     }
96 }
```

Figura 5 | Código CSS. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

A regra chamada “suave” nós criamos anteriormente, agora, foi adiciona a regra “top” (linha 89), que será responsável pelo movimento do formulário no sentido vertical, para cima. Será necessário trabalhar com o valor translateY, pois ele é o responsável por esse movimento vertical solicitado no desafio proposto para hoje.

Já o código javascript fica responsável pelos eventos de click e o animationend, que será executado ao final da animação. Veja o código:

```
1 // MOVIMENTO DO FORMULÁRIO
2 const btnLogin = document.querySelector('#button');
3 const form = document.querySelector('form');
4
5 btnLogin.addEventListener('click', event => {
6     event.preventDefault(); //impede carregamento da página
7     form.classList.add('oculta-form');
8 });
9
10 form.addEventListener('animationend', event =>{
11     if(event.animationName == 'top') {
12         form.style.display = 'none';
13         window.location.href = 'site.html';
14     }
15});
```

Figura 6 | Código Javascript. Fonte: elaborada pelo autor.

Ao final da animação (animationend), após o botão ser clicado, o formulário será ocultado (linha 12) e uma nova página será aberta, site.html, conforme consta na linha 13. Execute o projeto e veja o resultado.

## Saiba mais

Sugerimos a leitura do Capítulo 3, do livro *Projetos de Sistemas Web Conceitos, Estruturas, Criação de Banco de dados e Ferramentas de Desenvolvimento*, que trata de Gestão de projetos.

ALVES, W. P. [Projetos de sistemas web conceitos, estruturas, criação de banco de dados e ferramentas de desenvolvimento](#). 1. ed. São Paulo: Érica, 2015.

Veja também a obra *Desenvolvimento de Software com Metodologias Ágeis*.

# DESENVOLVIMENTO EM JAVASCRIPT

MASCHIETTO, L. G. et al. Desenvolvimento de software com metodologias ágeis. Porto Alegre: SAGAH, 2020.

## Referências

ALVES, W. P. **Projetos de sistemas web conceitos, estruturas, criação de banco de dados e ferramentas de desenvolvimento.** 1. ed. São Paulo: Érica, 2015.

FLANAGAN, D. **JavaScript:** o guia definitivo. Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MASCHIETTO, L. G. et al. **Desenvolvimento de software com metodologias ágeis.** Porto Alegre: SAGAH, 2020.

MDN WEB DOCS. **Elementos.** 2023. Disponível em: <https://developer.mozilla.org/en-US/docs/Web/API/Element>. Acesso em: 15 jan. 2024.

## Aula 5

Encerramento da Unidade

### Videoaula de Encerramento



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Chegada

# DESENVOLVIMENTO EM JAVASCRIPT

Olá, estudante, chegamos a mais um final! E que final, não é mesmo? A competência de implementar eventos com a linguagem Javascript pressupõe conhecer os eventos e suas finalidades. A implementação de eventos com essa linguagem é uma forma de melhorar o desenvolvimento de aplicações web, uma vez que esse caminho possibilita tornar páginas dinâmicas e interativas. Com a utilização de eventos, essa interação melhora drasticamente o que afeta diretamente a experiência do usuário.

Existe uma quantidade muito grande de eventos com finalidades variadas, e é preciso ter em mente que cada situação demandará uma abordagem distinta por parte da pessoa responsável pelo desenvolvimento do projeto web. Uma vez que você comprehende e conhece esses vários eventos, é possível aplicá-los adequadamente a uma determinada situação cujo problema deve ser resolvido. Diante disso, implementar eventos nas soluções que estão sendo escritas com a linguagem Javascript será fácil e orgânica.

## É Hora de Praticar!



### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

A programação orientada a eventos é uma prática que favorece muito o desenvolvimento de aplicações para web (Flanagan, 2013), e para que você fixe definitivamente esse conceito, criaremos um pequeno formulário e aplicaremos sobre ele uma forma de, por meio de eventos, realizar a sua validação. A intenção é mostrar ao usuário que os campos do formulário devem ser preenchidos para, então, poder avançar nas telas.

Diante disso, para esse exercício, crie um formulário com no máximo 4 campos, sendo: nome, telefone, e-mail e CPF. Ao clicar no botão do formulário, será exibida alguma forma de validação para orientar o usuário. Note que esse tipo de implementação é comum no mercado de trabalho, portanto, utilizar eventos na programação com Javascript é base para uma boa função na área.

Perceba que a solução apresentada é uma entre várias, por esse motivo, é valioso buscar outras formas de solucionar o problema. Tente, ao menos, encontrar outra maneira de resolvê-lo.

Antes de finalizarmos, lembre-se de que conhecer técnicas e ferramentas para auxiliar no gerenciamento de projetos de aplicações web também é extremamente relevante (Alves, 2015).

Entre o rol de possibilidades, estão as metodologias ágeis, que servem como abordagens de suporte ao processo de desenvolvimento, sobretudo quando esse processo acontece a várias mãos, ou seja, envolve um time repleto de conhecimentos (Maschietto *et. al.*, 2020).

- Você conseguiu associar os eventos às suas finalidades?

# DESENVOLVIMENTO EM JAVASCRIPT

- É capaz de identificar que tipo de evento está sendo utilizado em uma determinada aplicação?
- Consegue implementar os principais eventos que você conheceu?

A criação do formulário de cadastro segue o proposto no enunciado. Criamos 4 campos distintos e organizamos todos de forma a deixá-los dentro de elementos div, para controlar o estilo e, também, a programação com Javascript. O código proposto fica da seguinte forma:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1<!DOCTYPE html>
2<html lang="pt-br">
3<head>
4    <meta charset="UTF-8" />
5    <title>Validação</title>
6    <link rel="stylesheet" href="cadastro.css" />
7</head>
8<body>
9    <div class="container">
10        <form>
11            <h2>Cadastre-se</h2><br>
12            <div class="blocoInput">
13                <label for="">Nome</label>
14                <input type="text" name="nome" id="nome" class="form-input"><br>
15            </div>
16            <div class="blocoInput">
17                <label for="">Fone</label>
18                <input type="tel" name="fone" id="fone" class="form-input"><br>
19            </div>
20            <div class="blocoInput">
21                <label for="email">Email</label>
22                <input type="email" name="email" id="email" class="form-input"><br>
23            </div>
24            <div class="blocoInput">
25                <label for="cpf">CPF</label>
26                <input type="text" name="cpf" id="cpf" class="form-input"><br>
27            </div>
28            <button type="submit" class="btn">Login</button>
29        </form>
30    </div>
31    <script src="cadastro.js"></script>
32</body>
33</html>
```

Figura 1 | Código proposto. Fonte: elaborada pelo autor.

O CSS implementado para esse projeto leva em consideração questões como cor de fundo do projeto, centralização do formulário na página e padronização de campos do formulário no que diz respeito a tamanho e cores. O código proposto segue como nas imagens a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 * {  
2   padding: 0;  
3   margin: 0;  
4   font-family: sans-serif;  
5   box-sizing: border-box;  
6 }  
7 body {  
8   padding-top: 10%;  
9   background: -webkit-gradient(linear, left top, left bottom, from(rgba(70, 183, 236, 0.2)), to(rgba(24, 23, 27, 1))) fixed;  
10  width: 100vw;  
11  height: 30vw;  
12 }  
13 .container {  
14   display: block;  
15   align-items: center;  
16   margin-left: auto;  
17   margin-right: auto;  
18   width: 41em;  
19   padding: 2%;  
20   border-radius: 5px;  
21 }  
22 h1 {  
23   font-size: 32px;  
24   letter-spacing: 1px;  
25   margin: 20px 0;  
26   color: white;  
27 }  
28 form {  
29   background-color: white;  
30   padding: 30px 25px;  
31   border-radius: 10px;  
32 }  
33 form .blocoInput input {  
34   width: 100%;  
35   display: block;  
36   margin-top: 8px;  
37   padding: .75rem .75rem;  
38   color: rgb(0, 0, 0);
```

Figura 2 | Código CSS. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
39     border-radius: 2px;
40     font-size: 16px;
41     border: 1px solid #cccddef;
42     outline-color: rgb(70, 183, 236);
43 }
44 .btn{
45     width:40%;
46     height:40px;
47     border: none;
48     border-radius: 10px;
49     margin: 2.5%;
50     font-weight: 900;
51     font-size:large;
52     background-color: rgb(70, 183, 236);
53     color: rgb(255, 255, 255);
54 }
55 form.validateErro {
56     animation: preenche 200ms linear;
57     animation-iteration-count: 2;
58 }
59 @keyframes preenche {
60     0%,
61     100% {
62         transform: translateX(0);
63     }
64     35% {
65         transform: translateX(-15%);
66     }
67     70% {
68         transform: translateX(15%);
69     }
70 }
```

Figura 3 | Código CSS - cont. Fonte: elaborada pelo autor.

Agora, a programação com Javascript é a parte mais importante desse processo. A solução proposta não deixará o código muito extenso, contudo, faremos uso de eventos, escutador, seletores, métodos para obtenção de elementos do formulário escrito em HTML, estrutura de decisão e estrutura de repetição, basicamente. Veja o código a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 const btnLogin = document.querySelector(".btn");
2 const form = document.querySelector("form");
3
4 btnLogin.addEventListener("click", event => {
5   event.preventDefault();
6
7   const fields = [...document.querySelectorAll(".blocoInput input")];
8
9   fields.forEach(field => {
10     if (field.value === "") form.classList.add("validateErro");
11   });
12
13 const formError = document.querySelector(".validateErro");
14 if (formError) {
15   formError.addEventListener("animationend", event => {
16     if (event.animationName === "preenche") {
17       formError.classList.remove("validateErro");
18     }
19   });
20 } else {
21   alert('Tudo certo');
22 }
23 });
24});
```

Figura 4 | Código HTML. Fonte: elaborada pelo autor.

O evento escolhido para a solução do problema proposto é o evento click. Para utilizá-lo, adiciona-se um escutador ao código, o addEventListener().

Para propormos diferentes soluções, utilizamos os métodos para obtenção de elementos da página: o querySelector() e o querySelectorAll(). Note que, na linha 7, criamos um array com todos os elementos input do formulário para fazer a verificação do campo, se está preenchido ou

# DESENVOLVIMENTO EM JAVASCRIPT

não, e isso acontece na linha 10. Se estiver vazio, a animação validateErro será adicionada, executando um movimento e causando a impressão de tremor no formulário.

Além disso, foi feita uma verificação na linha 14 e adicionado um novo escutador, só que, agora, aplicando-se o animationend, que verifica se os campos foram preenchidos e remove a animação validateErro.

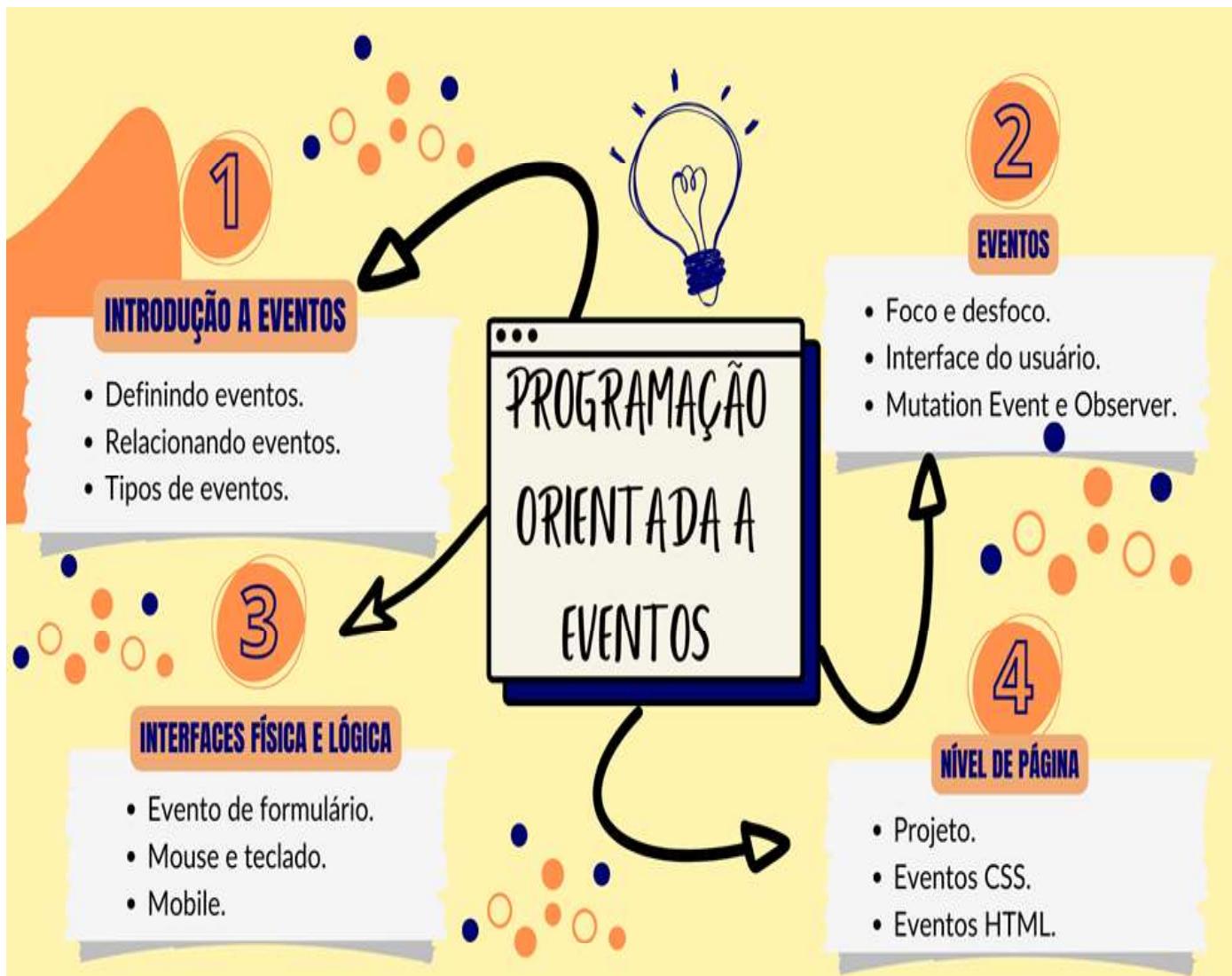


Figura | Programação orientada a eventos. Fonte: elaborada pelo autor.

ALVES, W. P. **Projetos de sistemas web conceitos, estruturas, criação de banco de dados e ferramentas de desenvolvimento.** 1. ed. São Paulo: Érica, 2015.

FLANAGAN, D. **JavaScript: o guia definitivo.** Tradução: João Eduardo Nóbrega Tortello. 6. ed. Porto Alegre: Bookman, 2013.

MASCHIETTO, L. G. et al. **Desenvolvimento de software com metodologias ágeis.** Porto Alegre: SAGAH, 2020.

# DESENVOLVIMENTO EM JAVASCRIPT

## Unidade 4

Frameworks - Bibliotecas Para Desenvolvimento Em Javascript

### Aula 1

Introdução a Frameworks em JavaScript

#### Introdução a frameworks em Javascript



##### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Partida

Olá, estudante! É com muito prazer que chegamos à fase final de nossa disciplina conhecendo os principais frameworks utilizados para o desenvolvimento de aplicações com a linguagem Javascript.

Neste primeiro momento, você conhecerá as características que tornam essas ferramentas indispensáveis no processo de desenvolvimento. É isso mesmo, muitos desses frameworks são utilizados no dia a dia de qualquer pessoa que atue com desenvolvimento, pois, de maneira geral, os frameworks oferecem celeridade e aumento da produtividade na tarefa de desenvolvimento.

Para ingressar no mundo dos frameworks Javascript, como desafio, você deverá montar uma pequena aplicação com um único objetivo: realizar a validação de um campo de e-mail. Esse desafio consiste em montar, utilizando Angularjs, um código que verifique, enquanto o usuário está digitando, se se trata de um e-mail válido ou não; caso o usuário informe o e-mail corretamente, ao final, a aplicação não exibirá nenhum tipo de mensagem.

# DESENVOLVIMENTO EM JAVASCRIPT

Vamos iniciar esta jornada de muito conhecimento acerca dos frameworks disponíveis no mercado? Espero que goste do que está por vir.

Bons estudos!

## Vamos Começar!

### Conceito de frameworks

Antes de iniciarmos, é importante que saiba que a linguagem Javascript é uma linguagem poderosa e cumpre bem o seu propósito, no entanto, a criação de frameworks é algo que tem acontecido com regularidade, pois utilizar esse tipo de ferramenta apresenta diversas vantagens, como veremos ao longo desta aula.

Frameworks são plataformas de desenvolvimento que fornecem modelos básicos com diferentes funcionalidades já implementadas, que podem ser utilizadas ou não pelo desenvolvedor (Zabot; Matos, 2020). Outra definição possível é que frameworks são um conjunto de códigos de uma linguagem de programação específica que auxiliam o desenvolvimento de projetos como de web, softwares, games, aplicativos, entre outros (Cardoso, 2021).

Os frameworks dão maior celeridade e agilidade ao processo de desenvolvimento, tornando vantajoso o seu uso por diversos motivos, e um dos principais é a diminuição de esforço (Marcolino, 2021). Outra característica importante dos frameworks é que eles trazem um conjunto de recursos, como grupos de bibliotecas, que possibilitam que os criadores executem alterações de operações de grande volume com maior agilidade e em pouco tempo (Carodoso, 2021).

### Diferença entre frameworks e APIs

As APIs são diferentes dos Frameworks. Você já comprehende o conceito de framework e entende as implicações que essa palavra traz a uma pessoa que trabalha com programação; Já as APIs é um acrônimo para *Application Programming Interface*, que, traduzido para o português, significa Interface de programação de aplicativos. De uma forma muito simples, quando falamos de uma Web API, dizemos que uma API é uma aplicação que serve recursos para outra aplicação. Por exemplo, você já fez uso de uma API, a API da ViaCEP, para que sua aplicação de cadastro de endereço consulte um dado endereço a partir do fornecimento do número do Código de Endereçamento Postal, o famoso CEP, você se recorda disso? Nesse caso, a API é uma aplicação que consulta os dados de endereço (rua, bairro, cidade e estado) a partir do CEP informado na sua aplicação e devolve para os dados mencionados, mas isso não significa que poucas são as APIs disponíveis para uso.

Existem várias APIs disponíveis para uso, muitas delas conhecidas pela maioria das pessoas. Existem APIs para análise de dados, APIs para Cloud, APIs de segurança e identidade e muitas

# DESENVOLVIMENTO EM JAVASCRIPT

outras (Google APIs Explorer, 2023).

## Siga em Frente...

## Principais frameworks em Javascript

Quando se trata de desenvolvimento Javascript, existe um número relativamente grande de frameworks disponíveis no mercado, no entanto, falaremos, basicamente, de 3, e o primeiro deles é o Vue.js

- **Vue.js**: trata-se de um framework progressivo para a construção de interfaces de usuário e tem uma peculiaridade muito importante: é um framework que foi projetado para ser utilizado de modo incremental, ou seja, apenas parte dele pode ser adotada (Marcolino, 2021).
- **Angular**: esse framework passou por várias modificações, iniciando-se como AngularJS – versão que ainda é utilizada em várias aplicações legadas. Uma vez que sofreu com muitos problemas, ao longo de sua história, evoluiu para sua versão atual, chamada somente angular (Marcolino, 2021). Essa ferramenta é considerada parte integrante da chamada Plataforma de Desenvolvimento Angular, que também inclui framework baseado em componentes para criação de aplicações web escaláveis, bem como conta, ainda, com uma coleção de bibliotecas bem definidas que servem para lidar com uma variedade de necessidades, incluindo rotinas, gerenciamento de formulários e muito mais (Marcolino, 2021).
- **React**: essa ferramenta, na verdade, está mais para uma biblioteca declarativa de interface de usuário, sendo adequada para o uso do padrão MVC (*Model-View-Controller*), além de apresentar uma performance muito boa, uma vez que permite atualizar o DOM apenas nos elementos que cria/altera (Marcolino, 2021).

Características	Angular	Vue	React
Quanto ao tipo de padrão arquitetural indicado.	Não segue um padrão específico em sua documentação.	É inspirado no padrão Modelo-Visão-VisãoModelo.	É baseado no padrão <i>Model-View-Controller</i> .
Quanto ao suporte da documentação e da comunidade.	Destaca-se negativamente por não possuir fórum para dúvidas.	Possui comunidade no Discord.	Possui comunidade no Discord.
Tamanho de pacotes de arquivos.	Num mesmo projeto, o angular gera arquivos maiores.	Apresenta tamanho intermediário entre os 3 comparados.	É o que apresenta o menor tamanho de arquivos de projeto.

# DESENVOLVIMENTO EM JAVASCRIPT

Tempo de renderização.	Apresenta o pior desempenho na renderização.	Apresenta desempenho intermediário na renderização.	Renderiza mais rápidos seus projetos.
------------------------	--	---	---------------------------------------

Quadro 1 | Comparativo. Fonte: adaptado de Marcolino (2021).

## Vamos Exercitar?

Para o estudo introdutório dos frameworks e de acordo com o desafio apresentado no início da aula, devemos montar uma pequena aplicação, algo simples, que servirá para validação de um campo do tipo e-mail, de formulário. Para cumpri-lo, vamos utilizar o framework Angular, mas note que, nesse momento, não é necessário instalar nada. Quanto a esse projeto, temos algumas maneiras de solucioná-lo, contudo, agora, vamos trabalhar com algumas diretivas disponíveis no Angular. São elas:

Diretiva	Descrição
ng-controller	Atribui uma classe de controller à uma View.
ng-disabled	Define atributo de um elemento como desativado.
ng-app	Designa o elemento root da aplicação e é utilizado próximo ao elemento raiz da página (<body> e <html>).
ng-show	Mostra ou oculta um elemento HTML de acordo com a expressão fornecida nessa diretiva.

Quadro 2 | Diretivas disponíveis no Angular. Fonte: adaptado de Angular (2023).

Para tornar o projeto o mais simples possível, não criaremos um arquivo para alojar o código Javascript; faremos tudo dentro do código HTML e utilizaremos a rede distribuída de conteúdo do Angularjs, que está disponível no site de mesmo nome, angularjs.org. Aliás, vale lembrar que temos, também, muito material disponível em Angular.io.

Montamos a estrutura do código HTML que ficará da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT

```

1 <!DOCTYPE html>
2 <html>
3
4 <head>
5   <script src="https://ajax.googleapis.com/ajax/libs/angularjs/1.6.9/angular.min.js"></script>
6   <title>Validação de e-mail</title>
7   <link href="https://cdn.jsdelivr.net/npm/bootstrap@5.3.2/dist/css/bootstrap.min.css" rel="stylesheet"
8       integrity="sha384-T3c6CoIi6uLrA9TneEoa7RxatzjcDSCmG1MXxSR1GAsXEVDwykc2MPK8M2HN" crossorigin="anonymous">
9   <style>
10    h2 {
11      text-align: center;
12    }
13   </style>
14 </head>

```

Figura 1 | Código HTML. Fonte: elaborada pelo autor.

Nessa primeira parte do código, inserimos o caminho do script do angularjs que está hospedado na estrutura do Google, mas note que vamos utilizar, também, o bootstrap 5, para fazer a estilização do projeto.

```

15 <body>
16   <h2>Validação de e-mail</h2>
17   <div class="container align-self-center d-flex justify-content-center">
18     <div class="align-middle">
19       <form ng-app="myApp" ng-controller="validateCtrl" name="myForm" novalidate>
20
21         <p>Email:<br>
22           <input type="email" name="email" ng-model="email" required>
23           <span style="color:red" ng-show="myForm.email.$dirty && myForm.email.$invalid">
24             <span ng-show="myForm.email.$error.required">Um e-mail é necessário.</span>
25             <span ng-show="myForm.email.$error.email">Endereço de e-mail inválido.</span>
26           </span>
27         </p>
28         <p>
29           <input type="submit" ng-disabled="myForm.user.$dirty && myForm.user.$invalid || myForm.email.$dirty && myForm.email.
$invalid" value="Testar">
30         </p>
31       </form>
32     </div>
33   </div>
34 </body>
35
36 </html>

```

Figura 2 | Código HTML – cont. Fonte: elaborada pelo autor.

No corpo da página, inserimos as classes que utilizamos do bootstrap e invocamos as diretivas que são necessárias para a criação da validação de e-mail; agora, vamos inserir um bloco de script, antes do fechamento do <body>, para criar o módulo cujo parâmetro se refere ao elemento HTML no qual a aplicação será executada. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
33    </div>
34    <script>
35        var app = angular.module('myApp', []);
36        app.controller('validateCtrl', function ($scope) {
37            $scope.email = '';
38        });
39    </script>
40
41 </body>
```

Figura 3 | Código HTML – cont. Fonte: elaborada pelo autor.

**Saiba mais:** O projeto está disponível no endereço [bit.ly/validaEmail-Angular](https://bit.ly/validaEmail-Angular) e pode ser baixado na íntegra. Nesse repositório, também é possível testar a aplicação funcionando.

## Saiba mais

Conheça algumas das principais diretivas do Angular:

ANGULAR. [Directive components in ng](#). 2022.

Sugerimos a leitura de *Frameworks back end*, de Leandro da Conceição Cardoso.

CARDOSO, L. da C. [Frameworks back end](#). São Paulo: Platos Soluções Educacionais S.A., 2021.

Conheça a seguir um conjunto de APIs fornecido só pela Alphabet, empresa dona de ferramentas como Google e Gmail. Disponível em:. Acesso em: 16 jan. 2024.

GOOGLE APIs EXPLORER. [APIs Explorer do Google](#). [s. d.]

Sugerimos, também, a leitura do capítulo *Diferenciando e escolhendo um framework front end*, do livro *Frameworks Front End*, de Marcolino.

MARCOLINO, A. da S. [Frameworks front end](#). São Paulo: Platos Soluções Educacionais S.A., 2021.

## Referências

ANGULAR. [AngularJS to Angular concepts](#): quick reference. [s. d.]. Disponível em: <https://angular.io/guide/ajs-quick-reference>. Acesso em: 16 jan. 2024.

# DESENVOLVIMENTO EM JAVASCRIPT

CARDOSO, L. da C. **Frameworks backend**. São Paulo: Platos Soluções Educacionais S.A., 2021.

GOOGLE APIs EXPLORER. **APIs explorer do Google**. [s. d.]. Disponível em: <https://developers.google.com/apis-explorer?hl=pt-br>. Acesso em: 16 jan. 2024.

MARCOLINO, A. da S. **Frameworks front end**. São Paulo: Platos Soluções Educacionais S.A., 2021.

W3C. **Web APIs: introduction**. 2023. Disponível em: [https://www.w3schools.com/js/js\\_api\\_intro.asp](https://www.w3schools.com/js/js_api_intro.asp). Acesso em: 16 jan. 2024.

ZABOT, D.; MATOS, E. **Aplicativos com bootstrap e angular**: como desenvolver apps responsivos. São Paulo: Erica, 2020.

## Aula 2

Angular

### Angular



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Partida

Olá, estudante! Nesta aula, vamos fazer uma imersão no framework Angular, conhecer um pouco sua história, como surgiu e entender em que momento estamos com relação à ferramenta, no que diz respeito à sua versão e funcionalidades. Além disso, faz-se importante, também, entender onde encontrar material sobre esse framework, pois a documentação faz toda a diferença quando alguém deseja adotar o Angular em sua rotina de programação.

# DESENVOLVIMENTO EM JAVASCRIPT

Para assegurarmos de que você entendeu bem como utilizar esse framework, você terá um novo desafio, deverá criar o ambiente de desenvolvimento utilizando Angular. Caso queira, poderá, também, utilizar o Bootstrap para estilizar o seu projeto. A ideia é ajustar o nosso formulário de login, aquele que já desenvolvemos anteriormente, ao framework em questão. Fique à vontade para usar sua criatividade e não se esqueça de utilizar o bootstrap na versão 5.x.

E aí, preparado para essa jornada de conhecimento e aprofundamento em tecnologias Javascript? Esperamos que sim. Vemo-nos na linha de chegada.

## Vamos Começar!

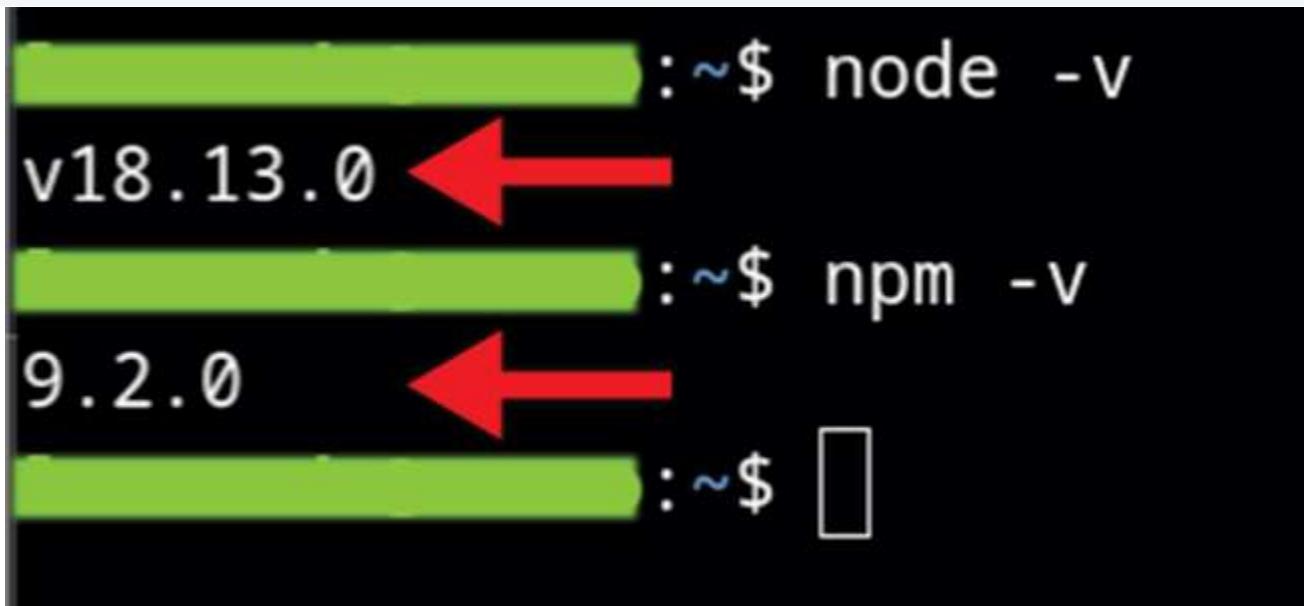
### Introdução e configuração do ambiente

O Angular tem uma longa história desde sua criação; ele foi lançado e por muito tempo mantido pela empresa Google (Zabot, 2020), tendo como propósito servir para desenvolvimento de aplicativos de página única (*single-page application*), sendo uma plataforma de desenvolvimento escrita em typeScript (Angular, 2023).

Para conseguir configurar o ambiente do Angular em seu computador, é necessário instalar o Node.js (dê preferência à versão LTS, que significa *Long-term support*, que nada mais é do que uma versão cuja estabilidade é de maior tempo) (NODEJS, 2023) e o npm, que é uma espécie de gerenciador de pacotes utilizado por desenvolvedores do mundo todo para se compartilhar e emprestar pacotes (Npm, 2023).

Após instalar o node.js e o npm, será a vez de instalar o Angular CLI, que permitirá que você crie projetos, gere aplicações e bibliotecas, bem como exerce uma variedade de tarefas de desenvolvimento contínuas, como teste, agrupamento e implantação (Angular, 2023). Para verificar se está tudo certo, basta que você digite os seguintes comandos no prompt de comando do seu computador:

# DESENVOLVIMENTO EM JAVASCRIPT



A screenshot of a terminal window on a black background. It shows the following text:

```
:~$ node -v  
v18.13.0 ←  
:~$ npm -v  
9.2.0 ←  
:~$ █
```

The text is in white, except for the red arrows pointing left from the version numbers. The prompt ':~\$' appears twice.

Figura 1 | Comandos. Fonte: captura de tela elaborada pelo autor.

Agora, instale o Angular CLI. Ainda utilizando o prompt de comando, digite o seguinte comando: `npm install -g @angular/cli`. Esse comando instalará o Angular CLI de forma global, ou seja, caso queira criar algum projeto, independentemente da pasta que esteja acessando, você poderá criá-lo (Angular, 2023).

Pronto, seu computador está preparado para iniciar a criação de aplicações utilizando o framework Angular.

## Componentes e módulos

A depender do projeto para o qual se pretende criar uma solução utilizando o Angular, é preciso pensar em conceitos próprios desse framework, como é o caso das rotas. Imagine que você pretende desenvolver um aplicativo para celular que tem como característica ser de página única, porque não há tempo para espera de carregamento de telas diferentes; nesse caso, você deverá utilizar o routing de páginas (Zabot, 2020). Com Angular, é possível criar sites com várias páginas que se parecem com aplicações dessa natureza de naveabilidade utilizando routing.

Após o processo de criação de um novo projeto, o Angular terá criado uma pasta do projeto com quase 30.000 arquivos (Zabot, 2020), em que a maioria será de definições da plataforma ou de utility, mas não se preocupe, nesses arquivos, não é necessário fazer qualquer alteração. O framework inclui um servidor para testar o app localmente, e isso é fundamental para se verificar o funcionamento.

Para que isso seja feito, no entanto, é necessário executar o comando `ng serve`, que inicializa um servidor na porta 4200 do localhost, permitindo o acesso à aplicação no seguinte endereço: <http://localhost:4200>.

# DESENVOLVIMENTO EM JAVASCRIPT

Um componente é composto de um arquivo CSS, um HTML e um arquivo Typescript. Veja um exemplo de estrutura de pastas e arquivos de um projeto feito com o framework:

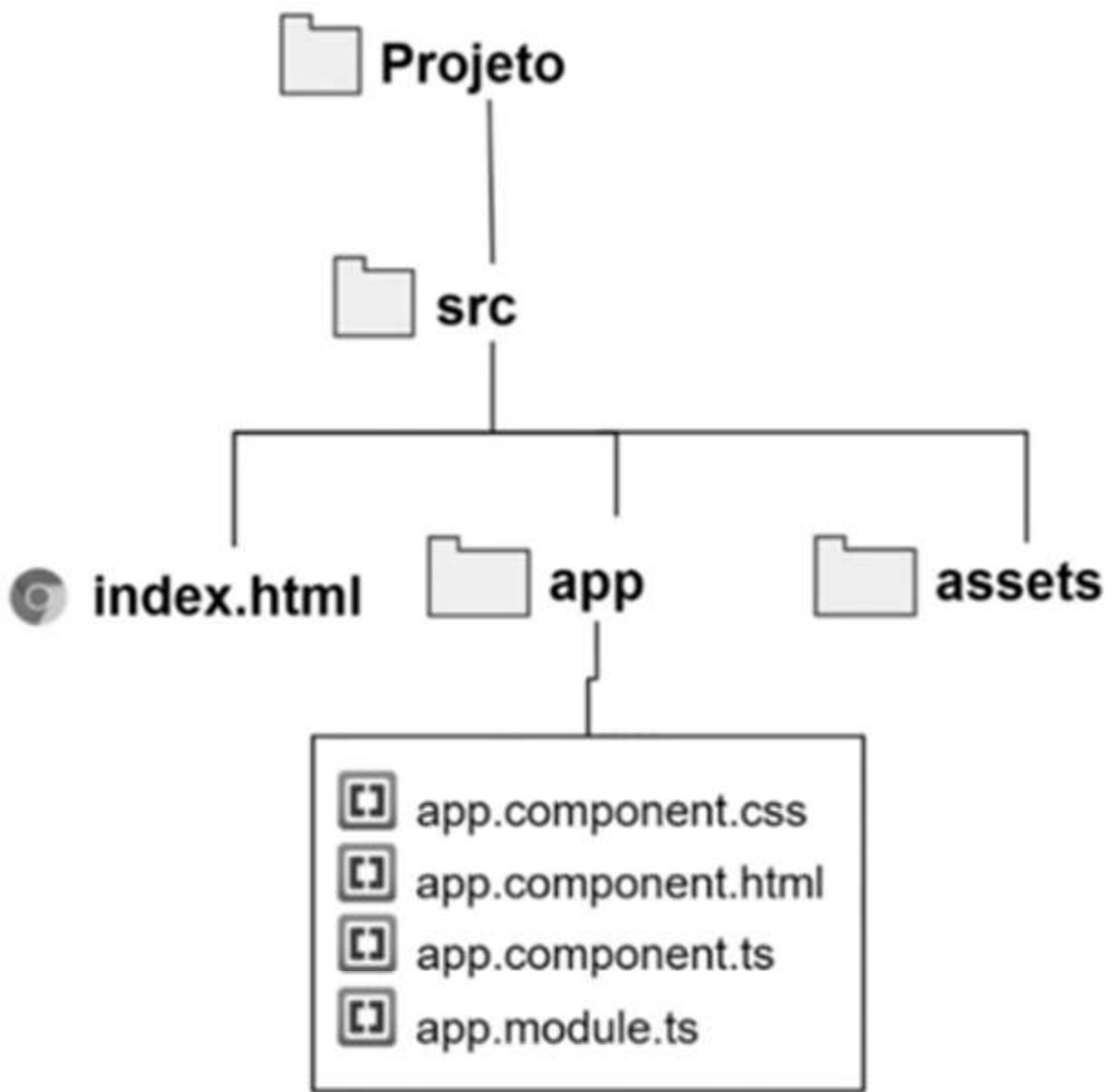


Figura 2 | Exemplo de estrutura de pastas e arquivos de um projeto feito com o framework. Fonte: Zabot (2020, p. 240).

De acordo com Zabot (2020), os arquivos constantes na pasta app significam:

- **app.module.ts**: módulo indicador das informações básicas do app.
- **app.component.css**: arquivo em que se encontra a configuração de estilo com CSS.
- **app.component.html**: contém o HTML do componente.
- **app.component.ts**: contém o controlador em código TypeScript.

# DESENVOLVIMENTO EM JAVASCRIPT

Siga em Frente...

## Criando um projeto com angular

Agora que já entendemos bem como funciona a configuração e a preparação de ambiente, bem como conhecemos os principais componentes de uma aplicação, vamos criar o nosso próprio projeto. O nome da nossa aplicação será formLogin, então, para criar essa aplicação, utilizaremos o seguinte comando: ng new formLogin.

Após o comando, o processo perguntará se desejamos utilizar routing e qual tipo de folha de estilo desejamos utilizar em nosso projeto. Basta, então, pressionar Enter e será escolhida a opção padrão, que é a que aparece em maiúsculo. Por exemplo (y/N) será escolhido o N, ou seja, não. Veja o resultado:

```
formContato$ ng new formContato
? Would you like to enable autocomplete? This will set up your terminal so pressing TAB while typing Angular CLI commands will show possible options and autocomplete arguments. (Enabling autocomplete will modify configuration files in your home directory.) Yes
Appended 'source <(ng completion script)>' to '/home/leonardo/.bashrc'. Restart your terminal or run the following to autocomplete 'ng' commands:
source <(ng completion script)
? Would you like to share pseudonymous usage data about this project with the Angular Team at Google under Google's Privacy Policy at https://policies.google.com/privacy. For more details and how to change this setting, see https://angular.io/analytics. No
Global setting: disabled
Local setting: No local workspace configuration file.
Effective status: disabled
? Would you like to add Angular routing? No
? Which stylesheet format would you like to use? CSS
CREATE formContato/README.md (1065 bytes)
CREATE formContato/.editorconfig (274 bytes)
CREATE formContato/.gitignore (548 bytes)
CREATE formContato/angular.json (2726 bytes)
CREATE formContato/package.json (1043 bytes)
CREATE formContato/tsconfig.json (901 bytes)
CREATE formContato/tsconfig.app.json (263 bytes)
CREATE formContato/tsconfig.spec.json (273 bytes)
CREATE formContato/.vscode/extensions.json (130 bytes)
CREATE formContato/.vscode/launch.json (470 bytes)
CREATE formContato/.vscode/tasks.json (938 bytes)
CREATE formContato/src/main.ts (214 bytes)
CREATE formContato/src/favicon.ico (948 bytes)
```

Figura 3 | Resultado do comando. Fonte: captura de tela elaborada pelo autor.

Abra a pasta criada no VSCode e você verá que a árvore de diretório do nosso projeto ficará da seguinte maneira:,

# DESENVOLVIMENTO EM JAVASCRIPT

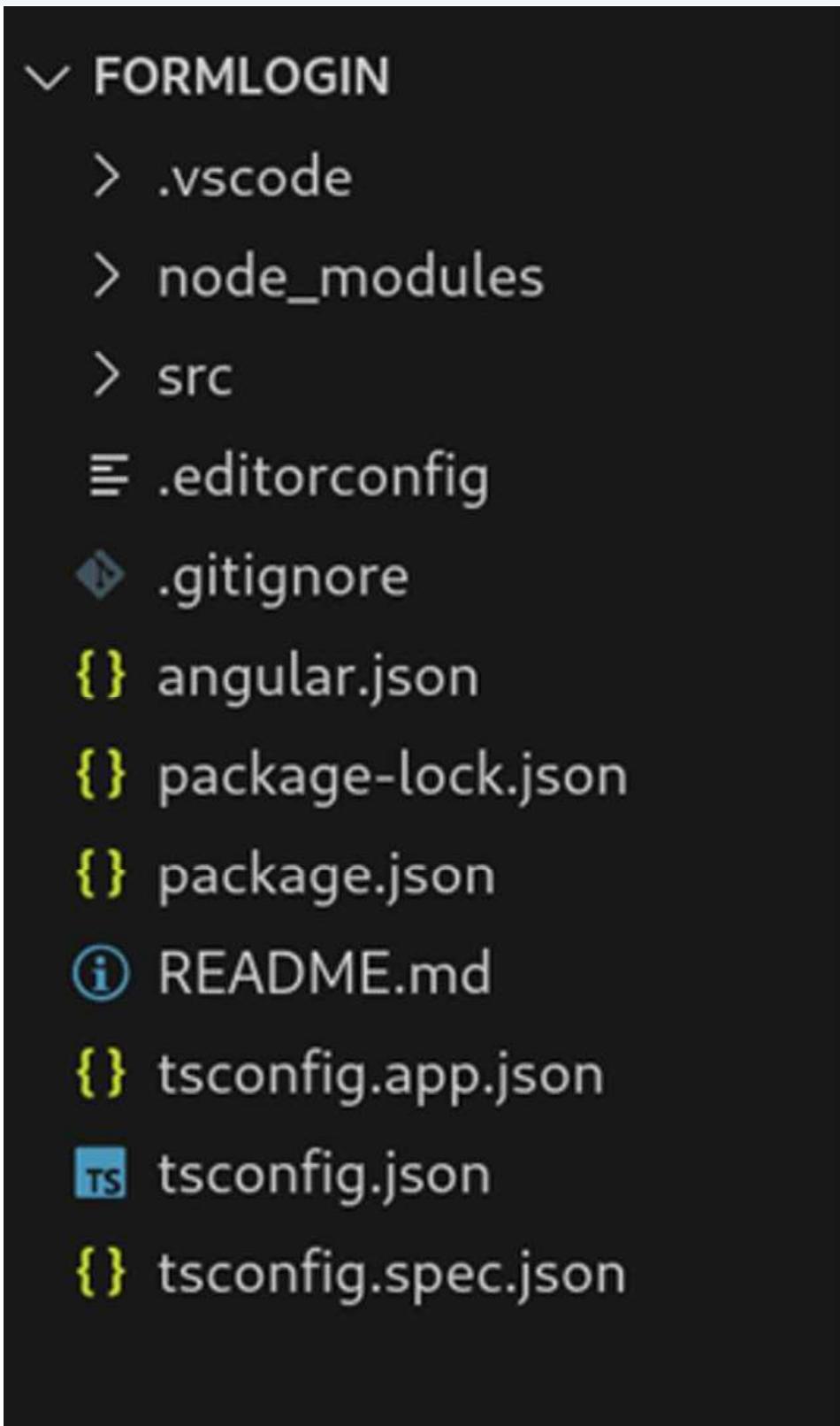


Figura 4 | Árvore de diretório do projeto. Fonte: captura de tela elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

Para deixar seu computador totalmente preparado, instale a extensão Angular Extension Pack, que facilitará o seu trabalho no desenvolvimento de aplicações no VSCode. Veja:

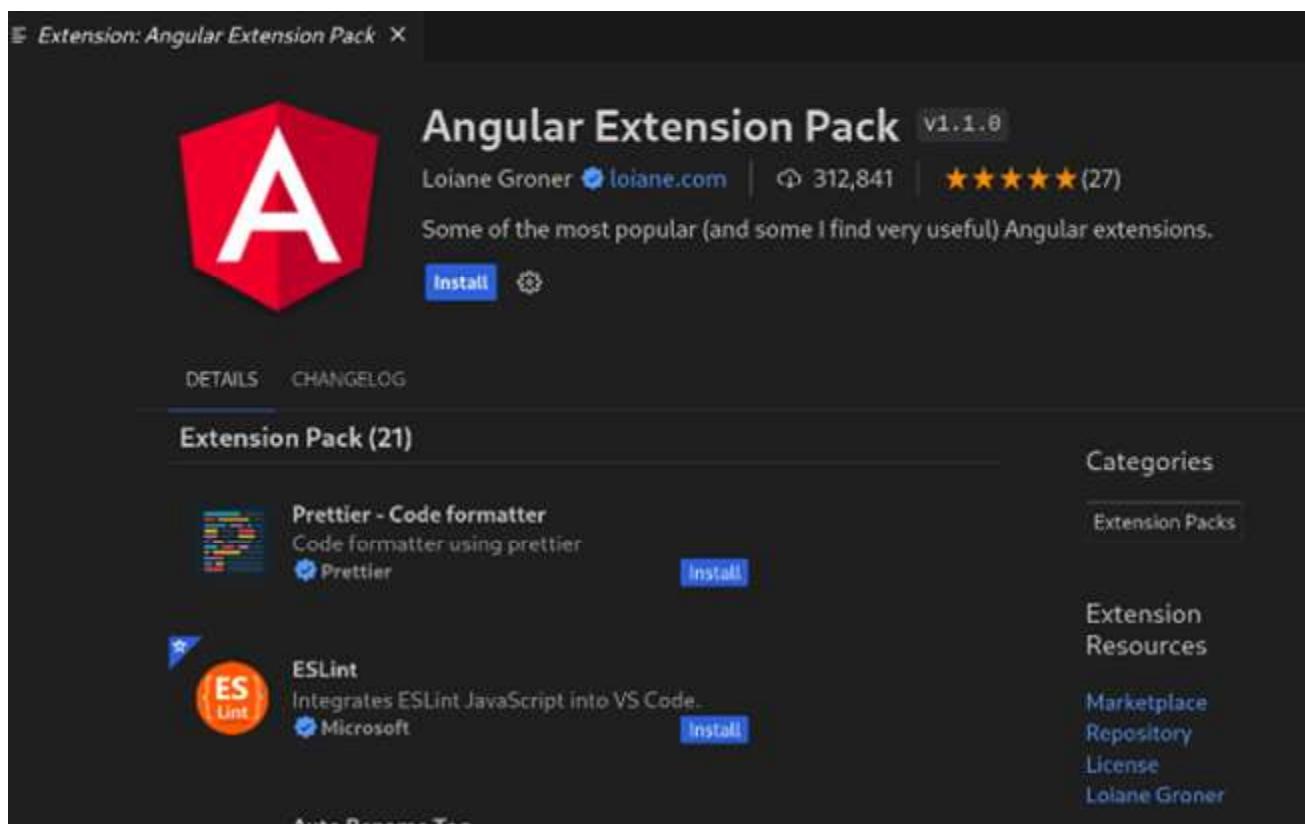


Figura 5 | Extensão Angular Extension Pack. Fonte: captura de tela elaborada pelo autor.

Pronto, agora é arregaçar as mangas e criar aplicações incríveis feitas com Angular.

## Vamos Exercitar?

Nós já construímos nossa aplicação ao longo desta aula, agora, vamos utilizar o projeto criado para adequá-lo à necessidade do nosso desafio, que é criar um formulário de Login utilizando Angular. Antes de mais nada, vamos levantar o servidor para verificar como está nossa aplicação, para isso, abra o terminal no VSCode no menu Terminal > New Terminal; em seguida, digite o comando `ng serve` e aguarde; será feita a build da aplicação e aparecerá o endereço para acessá-la. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
✓ Browser application bundle generation complete.

Initial Chunk Files      Names            Raw Size
vendor.js                 vendor           2.04 MB
polyfills.js               polyfills        333.17 KB
styles.css, styles.js     styles           230.45 KB
main.js                   main             46.19 KB
runtime.js                runtime          6.52 KB

Initial Total   2.64 MB

Build at: 2023-11-07T20:44:10.989Z - Hash: 7a246f612b573380 - Time: 11599ms

** Angular Live Development Server is listening on localhost:4200, open your browser on http://localhost:4200/ **

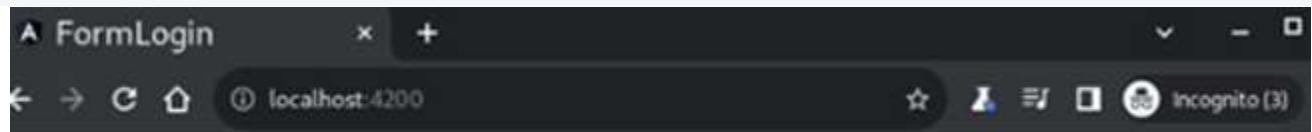
✓ Compiled successfully.
```

Figura 6 | Comando ng serve. Fonte: captura de tela elaborada pelo autor.

Abra a aplicação no navegador, no endereço <http://localhost:4200>, e veja o resultado. Vamos substituir o código gerado automaticamente pelo código de nossa tela de login; para isso, no VSCode, acesse o diretório **src > app > app.component.html** e apague todo o conteúdo desse arquivo; em seguida, basta colocar nele o código de sua tela de login, criado anteriormente. Lembre-se de trocar, também, o nome do arquivo CSS, porque, agora, no Angular, o nome deve ser **app.component.css**. Aproveite para colocar dentro desse arquivo o código css que você já havia feito. Caso prefira, basta utilizar o Bootstrap 5 para estilizar seu projeto.

Veja como ficará a aplicação:

# DESENVOLVIMENTO EM JAVASCRIPT



## Login

E-mail

Senha

[Esqueceu a senha?](#)

[LOGIN](#)

Figura 7 | Resultado da aplicação. Fonte: captura de tela elaborada pelo autor.

## Saiba mais

Sugerimos que você leia a respeito dos passos necessários para a criação da primeira aplicação utilizando Angular.

ANGULAR. [Build your first Angular app](#). 2023.

# DESENVOLVIMENTO EM JAVASCRIPT

Leia também o Capítulo 11 da obra *Aplicativos com bootstrap e angular – como desenvolver apps responsivos* sobre o Angular, em que você encontrará diversas informações, inclusive, sobre o processo de instalação.

ZABOT, D.; MATOS, E. [Aplicativos com bootstrap e angular – como desenvolver apps responsivos](#). São Paulo: Erica, 2020.

## Referências

ANGULAR. **Introduction to the Angular docs.** 2024. Disponível em: <https://angular.io/docs>. Acesso em: 16 jan. 2024.

NODEJS. **Node.js 20 changelog.** [s. d.]. Disponível em: [https://github.com/nodejs/node/blob/main/doc/changelogs/CHANGELOG\\_V20.md#20.9.0](https://github.com/nodejs/node/blob/main/doc/changelogs/CHANGELOG_V20.md#20.9.0). Acesso em: 10 nov. 2023.

NPM DOCS. **About npm.** 2023. Disponível em: <https://docs.npmjs.com/about-npm>. Acesso em: 16 jan. 2024.

ZABOT, D.; MATOS, E. **Aplicativos com bootstrap e angular:** como desenvolver apps responsivos. São Paulo: Erica, 2020.

## Aula 3

Vue

### Vue



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

# DESENVOLVIMENTO EM JAVASCRIPT

## Ponto de Partida

Olá, estudante! Desta vez, vamos ampliar nossos conhecimentos e falar um pouco sobre o framework Vue.js. Como acontece com outros frameworks disponíveis no mercado, Vue é uma dessas tecnologias que dispõe de diversas características que tornam o seu uso uma vantagem, a depender do projeto que se pretende desenvolver.

E para proporcionar uma prática que assegure a fixação dos conhecimentos propostos nesta aula, você terá de construir uma aplicação semelhante ao formulário de cadastro que já construiu, no entanto, faremos os recursos definidos em Javascript funcionarem dentro de um projeto feito com Vue.js.

O que você está achando desta oportunidade de estudar um novo framework e, ainda, um framework solicitado por várias empresas no mercado? Já pensou em ingressar numa grande empresa para atuar como responsável pelo desenvolvimento de aplicações utilizando framework Vue? Pois é, você pode! É só embarcar conosco no conteúdo desta aula.

Bons estudos!

## Vamos Começar!

## Introdução ao Vue

Em primeiro lugar, é importante que você entenda um pouco as características que o tornam um dos melhores frameworks para se utilizar na programação front-end.

Vue.js é um framework Javascript que, com Angular e React, integram o que chamamos de segunda geração de frameworks dessa linguagem (Marcolino, 2021). Segundo Marcolino, uma das principais características que tornam o Vue.js tão atrativo para os desenvolvedores é ser um framework progressivo, ou seja, não precisa ser adotado em sua totalidade no momento de desenvolver um dado projeto. Outro ponto apresentado é que Vue é um framework mantido e atualizado pela comunidade, por meio de doações.

O Vue também é considerado uma estrutura Javascript para construção de interfaces de usuário. Baseado em HTML, CSS e Javascript, ele fornece um modelo de programação declarativo e baseado em componentes que aumenta a eficiência no desenvolvimento de interfaces de usuário (VUE, 2023).

Assim como angular, para utilizar o Vue.js, é necessário instalar o Node.js. Com essa etapa superada, pode-se, então, instalar o Vue.js com o seguinte comando: `npm install vue`.

O resultado da instalação será algo semelhante à imagem a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
:~$ npm install vue
added 20 packages in 7s
2 packages are looking for funding
  run `npm fund` for details
:~$ █
```

Figura 1 | Resultado da instalação. Fonte: captura de tela elaborada pelo autor.

Faça a instalação com CLI globalmente, pois isso auxiliará na construção de projetos do tipo *single Page Applications*. Aliás, instale digitando o seguinte comando: `npm install -g @vue/cli`.

Agora, como um exemplo de como criar um novo projeto utilizando esse framework, vamos criar um com o nome `formCadastro`, para montar nosso primeiro projeto com `Vue.js`.

Note que existem, ao menos, três maneiras distintas de criar um projeto, e a primeira forma pode ser executada da seguinte maneira: `> npm create vue@latest`.

Também é possível utilizar o comando `npm create vue@next`, e ambos funcionam para construção de projetos.

Com o CLI instalado, é possível, ainda, criar um projeto de forma mais simples, utilizando o seguinte comando: `vue create <nome-do-projeto>`.

Se utilizar uma das duas maneiras apresentadas no comando, você será apresentado a vários recursos opcionais que poderão ser instalados, mas que, neste primeiro momento, marcaremos todos como “No”, ou seja, não instalaremos nenhum. Veja o resultado:

# DESENVOLVIMENTO EM JAVASCRIPT

```
1 $ npm create vue@latest
Need to install the following packages:
  create-vue@3.8.0
Ok to proceed? (y)

Vue.js - The Progressive JavaScript Framework

✓ Project name: formCadastro
✓ Package name: formcadastro
✓ Add TypeScript? No / Yes
✓ Add JSX Support? No / Yes
✓ Add Vue Router for Single Page Application development? No / Yes
✓ Add Pinia for state management? No / Yes
✓ Add Vitest for Unit Testing? No / Yes
✓ Add an End-to-End Testing Solution? No
✓ Add ESLint for code quality? No / Yes

Scaffolding project in [REDACTED] /formCadastro...

Done. Now run:

  cd formCadastro
  npm install
  npm run dev
```

Figura 2 | Resultado do comando de criação do projeto. Fonte: captura de tela elaborada pelo autor.

Veja como fica a árvore de diretórios de um projeto Vue.js dentro do VSCode:

# DESENVOLVIMENTO EM JAVASCRIPT

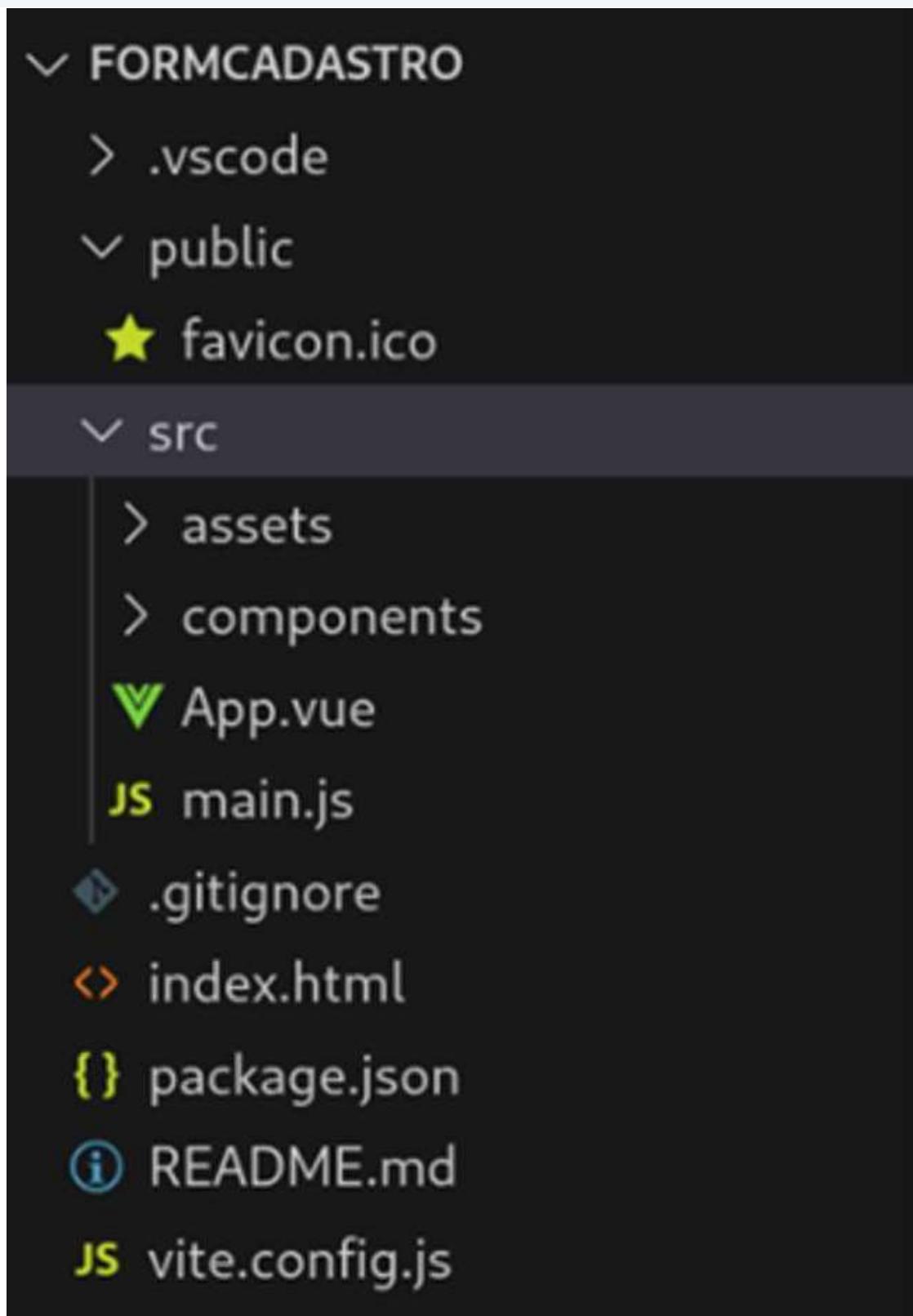


Figura 3 | Árvore de diretórios de um projeto Vue.js dentro do VSCode. Fonte: captura de tela elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

A criação de projeto utilizando o CLI pode perguntar qual versão do Vue.js você deseja escolher, e isso será feito de forma manual, como a imagem apresentada a seguir:

```
Vue CLI v5.0.8
? Please pick a preset: (Use arrow keys)
> Default ([Vue 3] babel, eslint)
  Default ([Vue 2] babel, eslint)
  Manually select features
```

Figura 4 | Definindo a versão do Vue.js. Fonte: captura de tela elaborada pelo autor.

Opte pela versão mais atual; em seguida, será feita a inicialização de plugins CLI, o que pode levar um tempo. Ao final, você verá como resultado algo semelhante à imagem que segue:

# DESENVOLVIMENTO EM JAVASCRIPT

```
4 moderate severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
  ⚡ Invoking generators...
  📦 Installing additional dependencies...


added 103 packages, and audited 972 packages in 9s

109 packages are looking for funding
  run `npm fund` for details

4 moderate severity vulnerabilities

To address all issues (including breaking changes), run:
  npm audit fix --force

Run `npm audit` for details.
  🛠️ Running completion hooks...

  📄 Generating README.md...

  🎉 Successfully created project form-cadastro.
  ➡️ Get started with the following commands:

    $ cd form-cadastro
    $ npm run serve
```

Figura 5 | Resultado. Fonte: captura de tela elaborada pelo autor.

Para finalizar, existem diversas extensões disponíveis no VSCode para serem utilizadas com Vue.js. Uma bem recomendada é a Vetur, conforme imagem a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

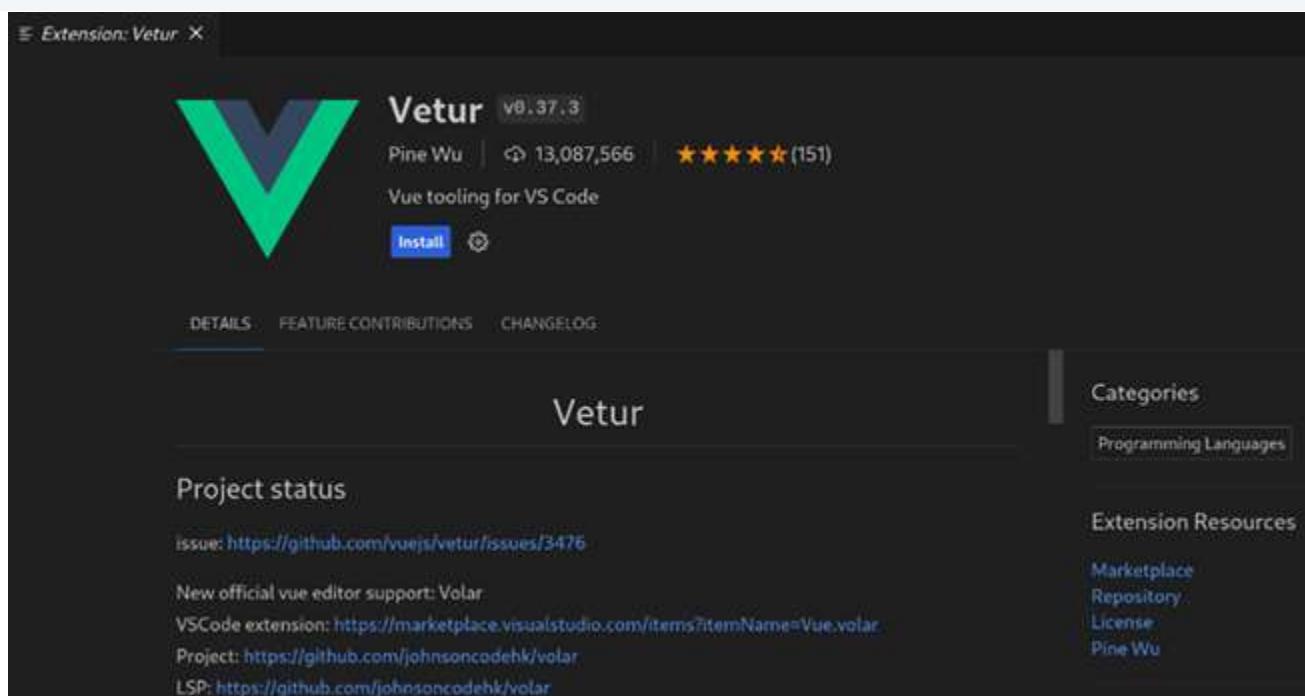


Figura 6 | Vetur. Fonte: captura de tela elaborada pelo autor.

Agora, você está pronto para trabalhar com Vue.js; é só começar a criar excelentes aplicações.

## Siga em Frente...

## Componentes VUE

O Vue.js oferece vários recursos, e um deles é a possibilidade de se trabalhar com componentes. Um componente é uma unidade de uma interface independente e pode possuir seu próprio estado, marcação (Marcolino, 2021). Um componente pode ser considerado uma maneira de dividir uma interface em pedaços, de modo a permitir sua montagem como se fosse um jogo de quebra-cabeça (Vue. Js Brasil, 2023).

Um bom exemplo de componente no desenvolvimento de aplicações com uso de Vue.js é permitir que o usuário, no contexto de um site de e-commerce, escolha um item de compra e verifique o estoque desse item antes de encaminhá-lo para área de pagamento da aplicação.

## Criando um projeto com VUE

Vamos trabalhar, neste momento, para adequar nosso formulário de cadastro, um projeto no qual já trabalhamos, ao framework Vue.js, para tanto, abriremos a aplicação no VSCode.

Como você deve ter notado, na tela de criação do projeto usando o CLI, ao final, é possível ver o comando necessário para rodar o build da aplicação. No VSCode, abra o terminal, digite npm run

# DESENVOLVIMENTO EM JAVASCRIPT

serve e veja que a aplicação será levantada na porta 8080, por padrão. O resultado será semelhante à imagem a seguir:

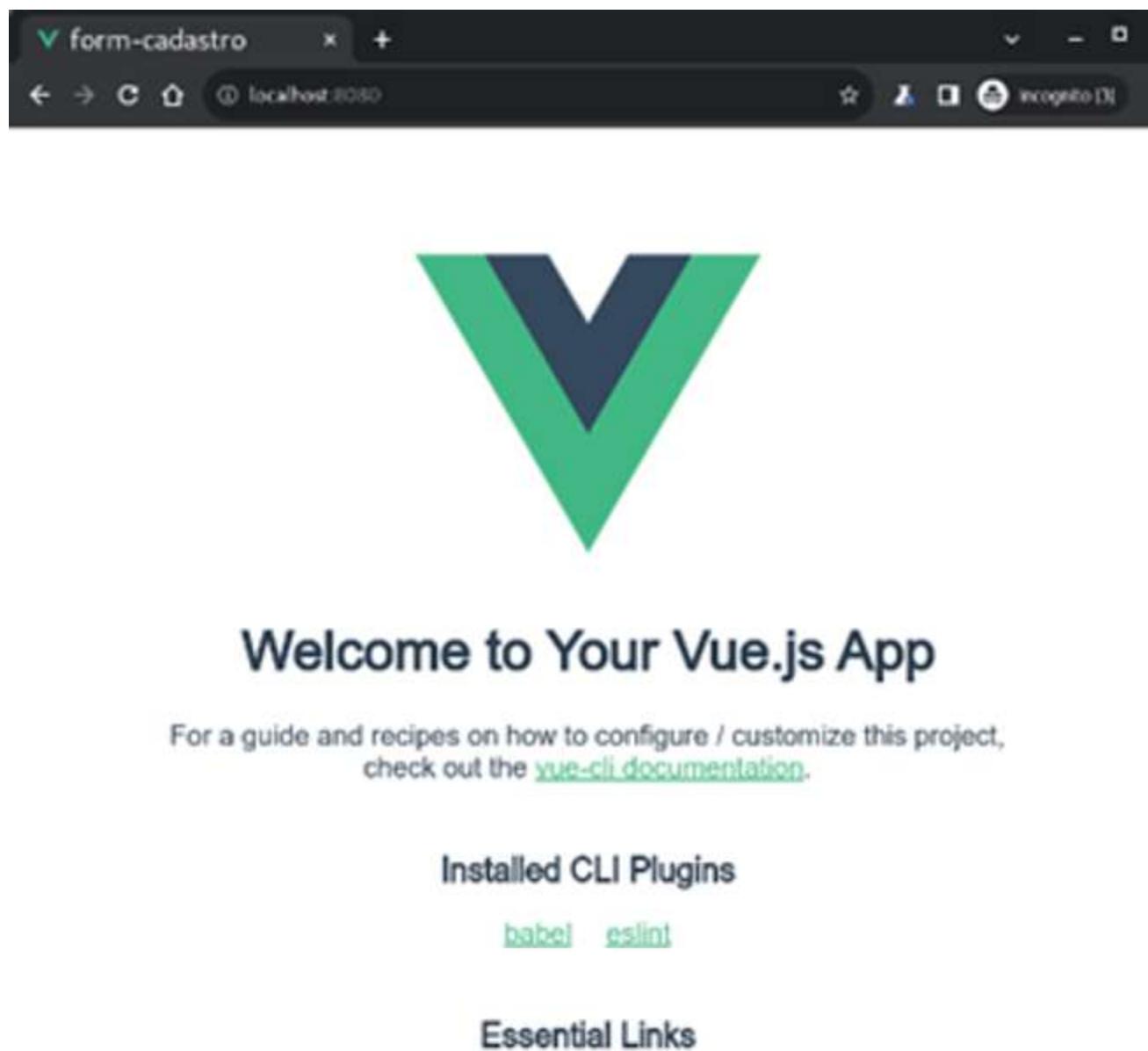


Figura 7 | Resultado do comando. Fonte: captura de tela elaborada pelo autor.

Agora, remova o arquivo que está no diretório src > components > HelloWorld.vue. No arquivo que está em src > App.vue, apague o conteúdo que está entre <template> </template>, remova a linha que aparece import HelloWorld from './components/HelloWorld.vue' e modifique o seguinte código:

Mude esse	Para esse
export default { name: 'App',	export default { name: 'App',

# DESENVOLVIMENTO EM JAVASCRIPT

```
components: {
  HelloWorld
}
}
```

Quadro 1 | Código a ser modificado. Fonte: elaborado pelo autor.

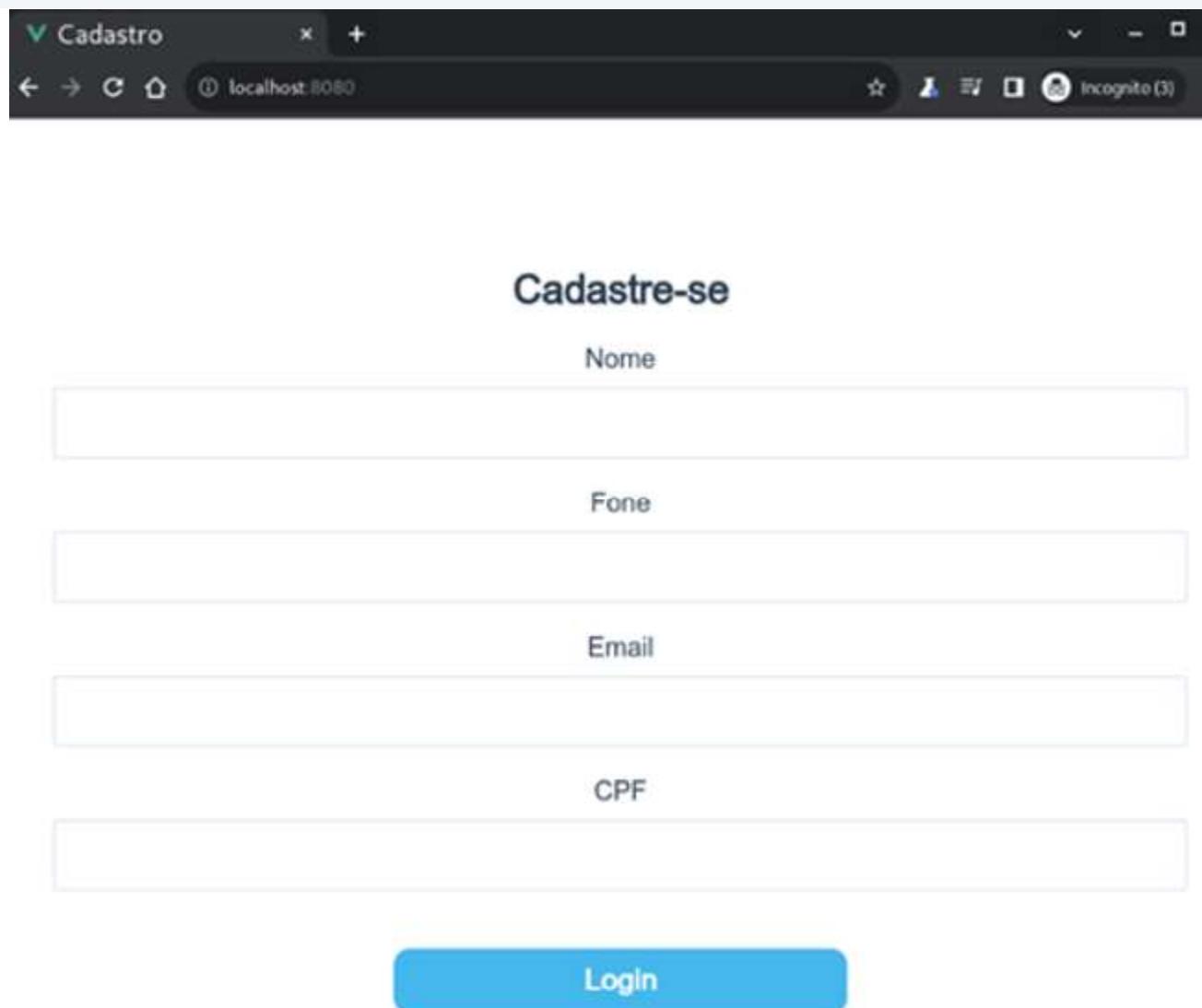
Agora, dentro dos elementos <template></template>, no arquivo App.vue, o código do formulário deve ser inserido, ficando da seguinte maneira:

```
1 <template>
2   <div class="container">
3     <form @submit.prevent="check()">
4       <h2>Cadastre-se</h2>
5       <br />
6       <div class="blocoInput">
7         <label for="">Nome</label>
8         <input type="text" name="nome" id="nome" class="form-input" /><br />
9       </div>
10      <div class="blocoInput">
11        <label for="">Fone</label>
12        <input type="tel" name="fone" id="fone" class="form-input" /><br />
13      </div>
14      <div class="blocoInput">
15        <label for="email">Email</label>
16        <input type="email" name="email" id="email" class="form-input" /><br />
17      </div>
18      <div class="blocoInput">
19        <label for="cpf">CPF</label>
20        <input type="text" name="cpf" id="cpf" class="form-input" /><br />
21      </div>
22      <button type="submit" class="btn">Login</button>
23    </form>
24  </div>
25 </template>
```

Figura 8 | Código HTML. Fonte: elaborada pelo autor.

Abaixo, no mesmo arquivo, entre os elementos <style scoped> </style>, insira o código CSS do projeto de formulário feito anteriormente. Agora, no arquivo que está em public > index.html, deixe dentro do <body> </body> apenas o seguinte código: <div id="app"></div> e substitua o comando que está nos elementos <title></title> para Cadastro. O resultado ficará assim:

# DESENVOLVIMENTO EM JAVASCRIPT



The screenshot shows a web browser window titled "Cadastro" with the URL "localhost:8080". The page content is a registration form titled "Cadastre-se". It contains four input fields: "Nome" (Name), "Fone" (Phone), "Email", and "CPF", each with a corresponding input box. Below the inputs is a large blue button labeled "Login".

Figura 9 | Formulário de cadastro. Fonte: captura de tela elaborada pelo autor.

E então, o que achou de montar uma aplicação de cadastro utilizando o framework Vue.js?

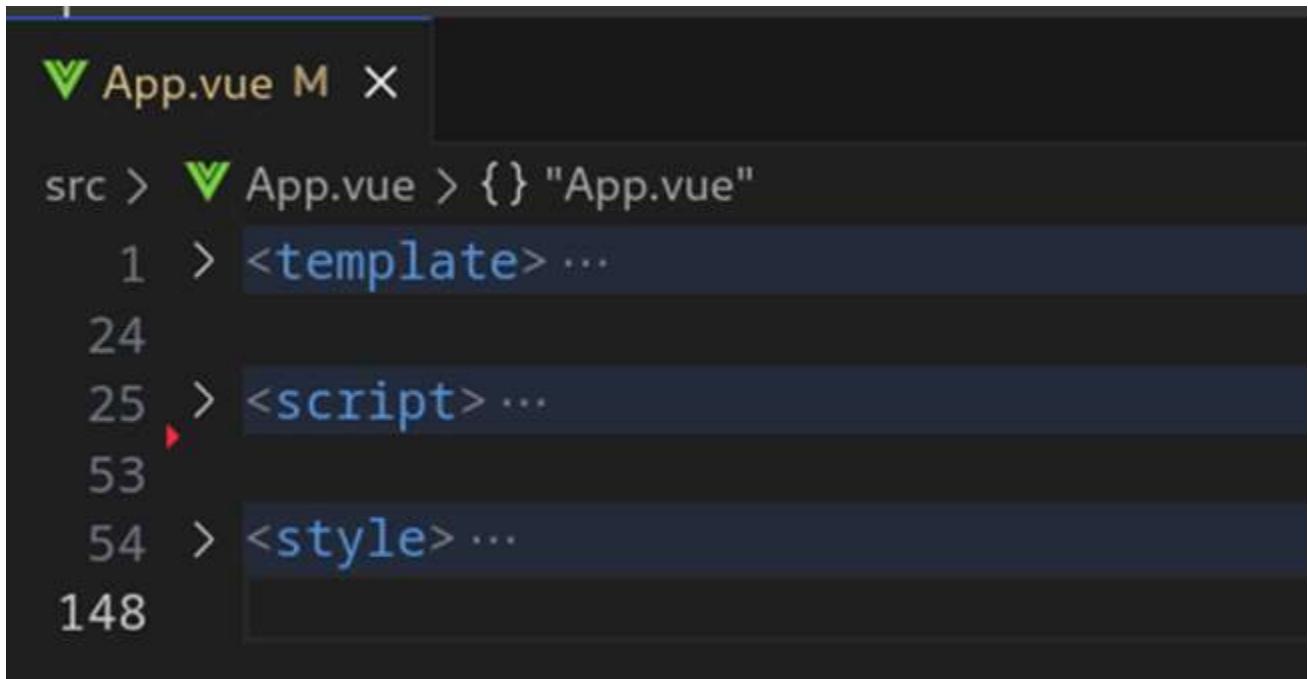
## Vamos Exercitar?

O projeto proposto para estudo nesta aula já está parcialmente construído, agora, nós vamos entender melhor como o principal arquivo do Vue.js e sua estrutura estão organizados, e essa compreensão é primordial para se construir aplicações com esse framework. O framework Vue tem uma estrutura e é necessário saber onde se deve colocar cada código de um projeto. Pensando, por exemplo, na linguagem CSS. Para auxiliar na fixação do que já vimos até agora, essa atividade consistirá na adaptação de um código criado previamente por você, em aulas passadas, à estrutura do Vue. Com isso, você compreenderá como esse framework funciona. Lembre-se que estamos fazendo o aproveitamento de código que já havíamos construído

# DESENVOLVIMENTO EM JAVASCRIPT

anteriormente, recorda? Agora, a intenção é entender como podemos utilizar esse código dentro de uma aplicação construída com o framework que estamos estudando nessa aula.

Até aqui, nós aprendemos a separar os arquivos de CSS do código principal da aplicação, no entanto, se você observar o arquivo constante em **src > App.vue**, perceberá que ele é organizado em 3 seções. Veja:



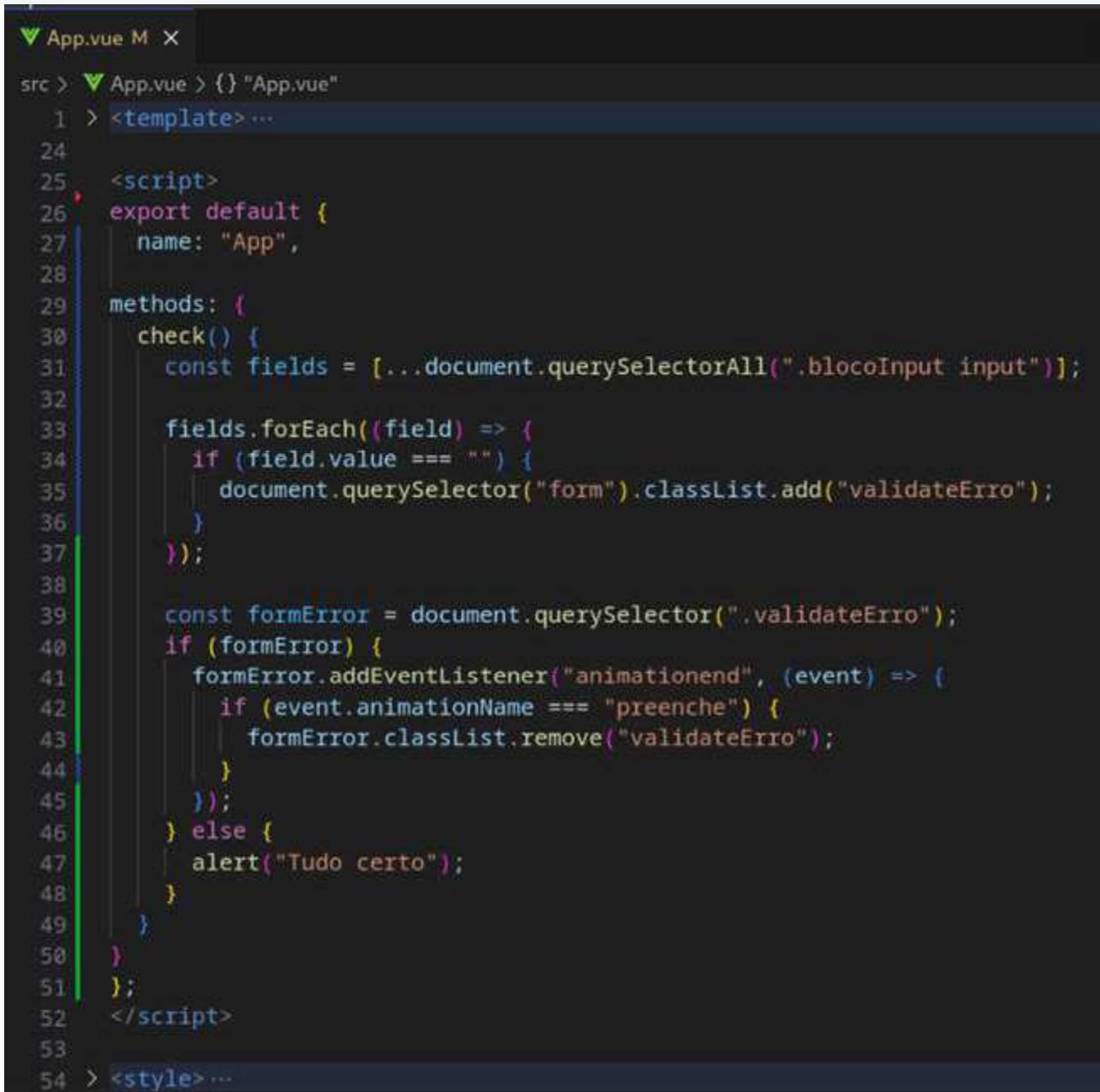
```
src > App.vue > {} "App.vue"
  1 > <template> ...
  24
  25 > <script> ...
  53
  54 > <style> ...
  148
```

Figura 10 | Organização das seções. Fonte: elaborada pelo autor.

Note que há o local específico para criação da estrutura com HTML (`<template>`), do estilo com CSS (`<style>`) e do código em Javascript, com (`<javascript>`). Claro que essa não é a única maneira, mas, no momento, é a forma como aplicaremos Javascript no projeto.

Lembre-se de que a ideia é aplicar animação no formulário para que ele mostre, de forma visual, que os campos não foram preenchidos e, caso contrário, apresente uma mensagem de alerta dizendo que tudo foi preenchido. Vamos colocar nosso código Javascript que cuidará disso. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT



```
App.vue M X
src > App.vue > {} "App.vue"
1 > <template>...
24
25 <script>
26 export default {
27   name: "App",
28
29   methods: {
30     check() {
31       const fields = [...document.querySelectorAll(".blocoInput input")];
32
33       fields.forEach((field) => {
34         if (field.value === "") {
35           document.querySelector("form").classList.add("validateErro");
36         }
37       });
38
39       const formError = document.querySelector(".validateErro");
40       if (formError) {
41         formError.addEventListener("animationend", (event) => {
42           if (event.animationName === "preenche") {
43             formError.classList.remove("validateErro");
44           }
45         });
46       } else {
47         alert("Tudo certo");
48       }
49     }
50   };
51 }
52 </script>
53 > <style>...
```

Figura 11 | Código Javascript. Fonte: elaborada pelo autor.

Nós criamos uma propriedade chamada methods (na linha 29) em que definimos qual ação realizar. Na seção <template>, mais precisamente dentro do elemento <form>, é preciso chamar o @submit que servirá para acessar algum evento ou método, sendo possível definir, ainda, a configuração que evitará que o formulário seja recarregado. Basta adicionar o prevent. Veja como deverá ficar a linha do elemento <form>: <form @submit.prevent="check()">.

Essa alteração permitirá executar o código da propriedade methods. Agora, teste a aplicação e veja como ficou sua validação e animação utilizando esse framework.

# DESENVOLVIMENTO EM JAVASCRIPT

## Saiba mais

Sugerimos a leitura do Capítulo *Compreendendo e utilizando o Framework Vue.js*, do livro *Frameworks Front End*.

MARCOLINO, A. da S. **Frameworks front end.** São Paulo: Platos Soluções Educacionais S.A., 2021.

Leia também o artigo que trata de componentes reutilizáveis em Vue.js.

VUE JS BRASIL. **Criando componentes reutilizáveis (mestre/detalhe).** 2018.

## Referências

MARCOLINO, A. da S. **Frameworks front end.** São Paulo: Platos Soluções Educacionais S.A., 2021.

VUE. JS BRASIL. **Criando componentes reutilizáveis (mestre/detalhe).** 2023. Disponível em: <https://vuejsbrasil.org/criando-componentes-reutilizaveis-mestre-detalhe/>. Acesso em: 16 jan. 2024.

VUE. JS. **Introdução.** 2022. Disponível em: <https://vuejsbr-docs-next.netlify.app/guide/introduction.html>. Acesso em: 11 nov. 2023.

## Aula 4

React

### React



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

# DESENVOLVIMENTO EM JAVASCRIPT

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

## Ponto de Partida

Olá, estudante! Estamos chegando ao final de nossa jornada de estudos e não poderíamos fechar nossa sequência de super frameworks sem falar de um dos mais conhecidos. Na aula de hoje, vamos conhecer o Framework React, uma excepcional ferramenta para o desenvolvimento de aplicações web; ver os detalhes que tornam esse framework uma ferramenta muito interessante de ser utilizada; e saber como criar um componente de maneira simples e prática.

No desafio desta aula, você será convidado a criar uma aplicação com validação, podendo se tratar de um projeto que sirva para validar um campo que deva receber como valor o nome a ser informado pelo usuário, e esse projeto deverá emitir alguma mensagem a ele quando o botão, que poderá se chamar “Verificar”, for pressionado, para o caso de o usuário não informar nenhum nome.

O React, assim como os demais frameworks disponíveis no mercado, oferece uma gama de possibilidades, e conhecer e até dominar um framework é garantir um bom emprego, na certa.

Vamos começar nossa trilha de estudos com React?

Bons estudos!

## Vamos Começar!

## Introdução e configuração do ambiente React

React é uma biblioteca robusta que pode ser adotada de forma incremental (Marcolino, 2021); com ela, é possível utilizar um conjunto de ferramentas que contribuem para o processo de desenvolvimento. Alguns exemplos, são:

Ferramenta	Descrição
Create React App	Aprender React ou criar um <i>single-page app</i> .
Next.js	Criar site renderizado no servidor (SSR) com Node.js.
Gatsby	Criar site estático orientado a conteúdo.
Nx, Parcel e outros	Utilizado por desenvolvedores mais experientes.

# DESENVOLVIMENTO EM JAVASCRIPT

Quadro 1 | Ferramentas fornecidas pelo React. Fonte: adaptado de Marcolino (2021).

É possível criar uma aplicação React sem qualquer instalação, no entanto, esse processo é muito mais moroso (React, 2023). A fim de que compreenda sua utilização e o processo de criação de aplicações, vamos adotar a ferramenta Create React App.

Assim como os demais Frameworks, com o Node.js instalado, basta a instalação da ferramenta mencionada; para isso, digite o seguinte comando no prompt em seu computador: `npm install -g create-react-app`.

A instalação será feita normalmente, de maneira muito semelhante aos últimos dois frameworks estudados. Com a utilização do Create React App, é possível levantar uma aplicação a partir de um modelo, no entanto, a ferramenta permite a criação de uma aplicação totalmente nova.

## Componentes React

A construção de componentes é um processo relativamente simples; geralmente, um componente recebe um nome qualquer seguido da extensão JS e serve para otimizar e dar celeridade ao desenvolvimento. O React suporta todos os componentes SVG e HTML integrados ao navegador (React, 2023).

Já sabemos que a integração entre HTML, CSS e Javascript nos permite criar funcionalidades como barra lateral, modal, dropdown e muitos outros, e isso é o que chamamos de recursos de Interface de usuário. Com React, é possível combinar tudo isso em “componentes” personalizados, elementos de interface de usuário reutilizáveis para a sua aplicação (React, 2023).

Vale mencionarmos que React não é uma ferramenta exclusiva para desenvolvimento web; temos como exemplo o React Native, que serve para a construção de aplicativos móveis (MDN, 2023).

De maneira tradicional, a pessoa que constrói uma aplicação web, estrutura o conteúdo com HTML e adiciona interações com a linguagem Javascript. Como existem sites e aplicações que lidaram com um volume extenso de interações, utilizar o React pode ser uma boa, pois o framework coloca a interatividade em primeiro lugar enquanto ainda usa a mesma tecnologia: um componente React é uma função Javascript que permite a adição de tags HTML (React, 2023). Aliás, a criação de um projeto é feita de forma simples; para se criar um projeto chamado hello-world, basta digitar o comando: `npx create-react-app hello-world`.

Ao abri-lo no VSCode, vemos a árvore de diretórios do projeto, como na imagem a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

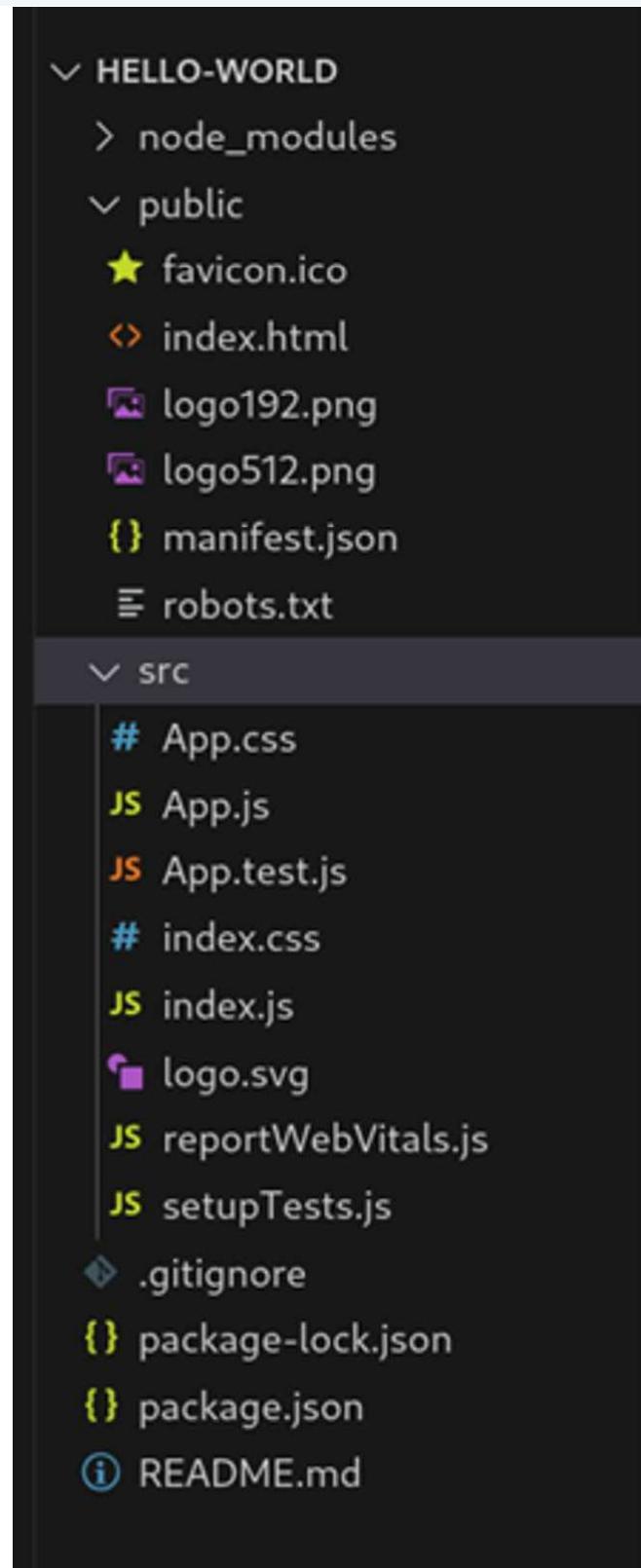


Figura 1 | Árvore de diretórios do projeto. Fonte: captura de tela elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

Se observarmos bem, veremos que a estrutura de pastas criada é muito semelhante aos frameworks vistos. Para rodar o projeto dentro do VSCode, abra o terminal e digite o comando: **npm start**. Será feito o build da aplicação e o resultado aparecerá no navegador, como na imagem a seguir:

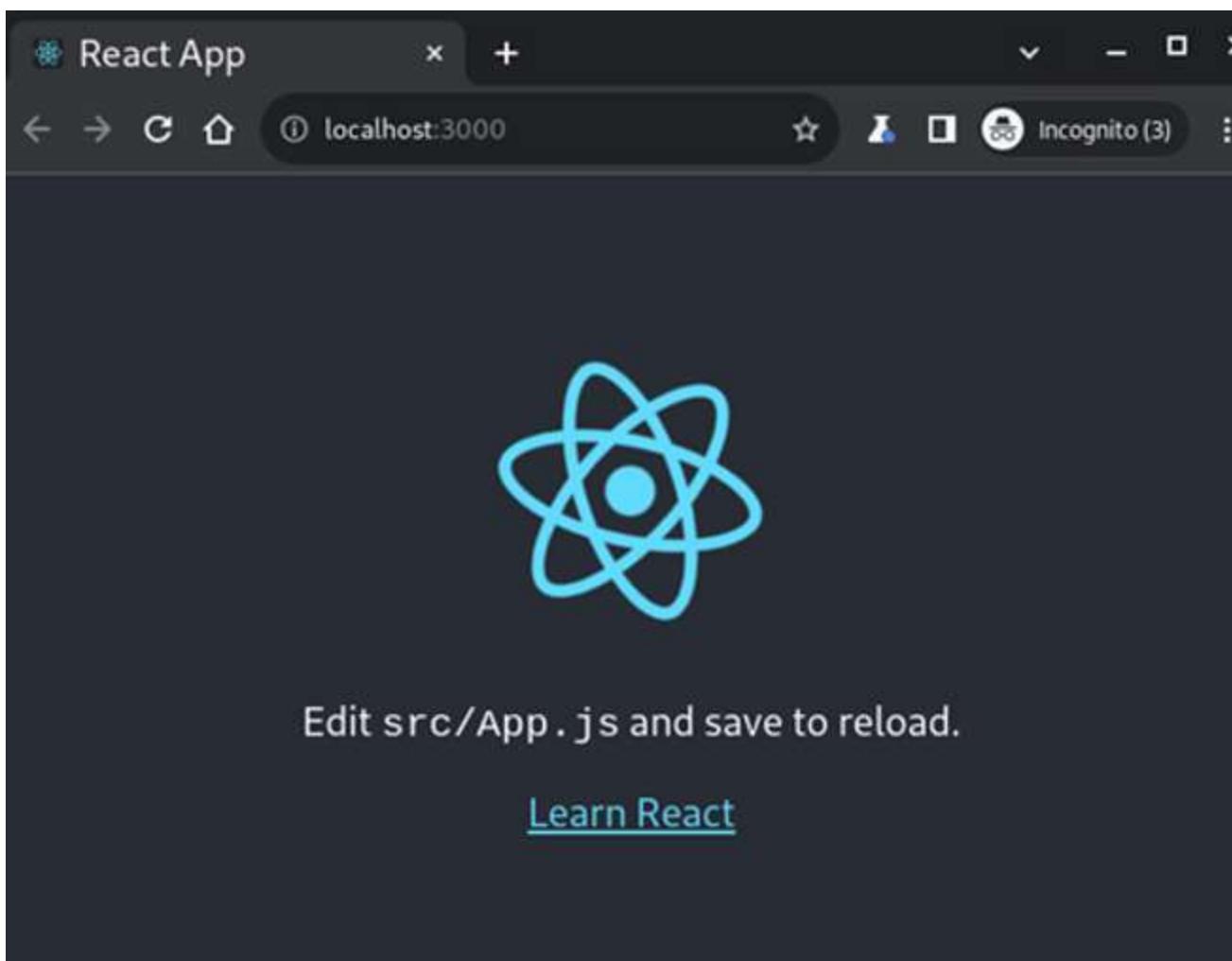


Figura 2 | Resultado. Fonte: captura de tela elaborada pelo autor.

Por fim, é indispensável saber que componentes do React utilizam *props* para se comunicarem uns com os outros. Todo componente pai pode passar alguma informação aos seus filhos por meio deles (React, 2023). Aliás, um *props* lembra um atributo HTML, mas não deixe de ler sobre a criação de componentes no **Saiba Mais**.

**Siga em Frente...**

**Criando um projeto com React**

# DESENVOLVIMENTO EM JAVASCRIPT

Para o exercício desta aula, vamos criar um projeto simples para validação de nome utilizando Javascript. Para isso, crie um projeto simples com React, pois, como se trata de um framework, o projeto simples já nos dará um bom trabalho. Execute o comando `npx create-react-app mouse-position` e acesse-o pelo VSCode. Esse projeto será usado para configurarmos a detecção da posição do ponteiro do mouse dentro da página. Agora, vamos remover os seguintes arquivos:

# DESENVOLVIMENTO EM JAVASCRIPT

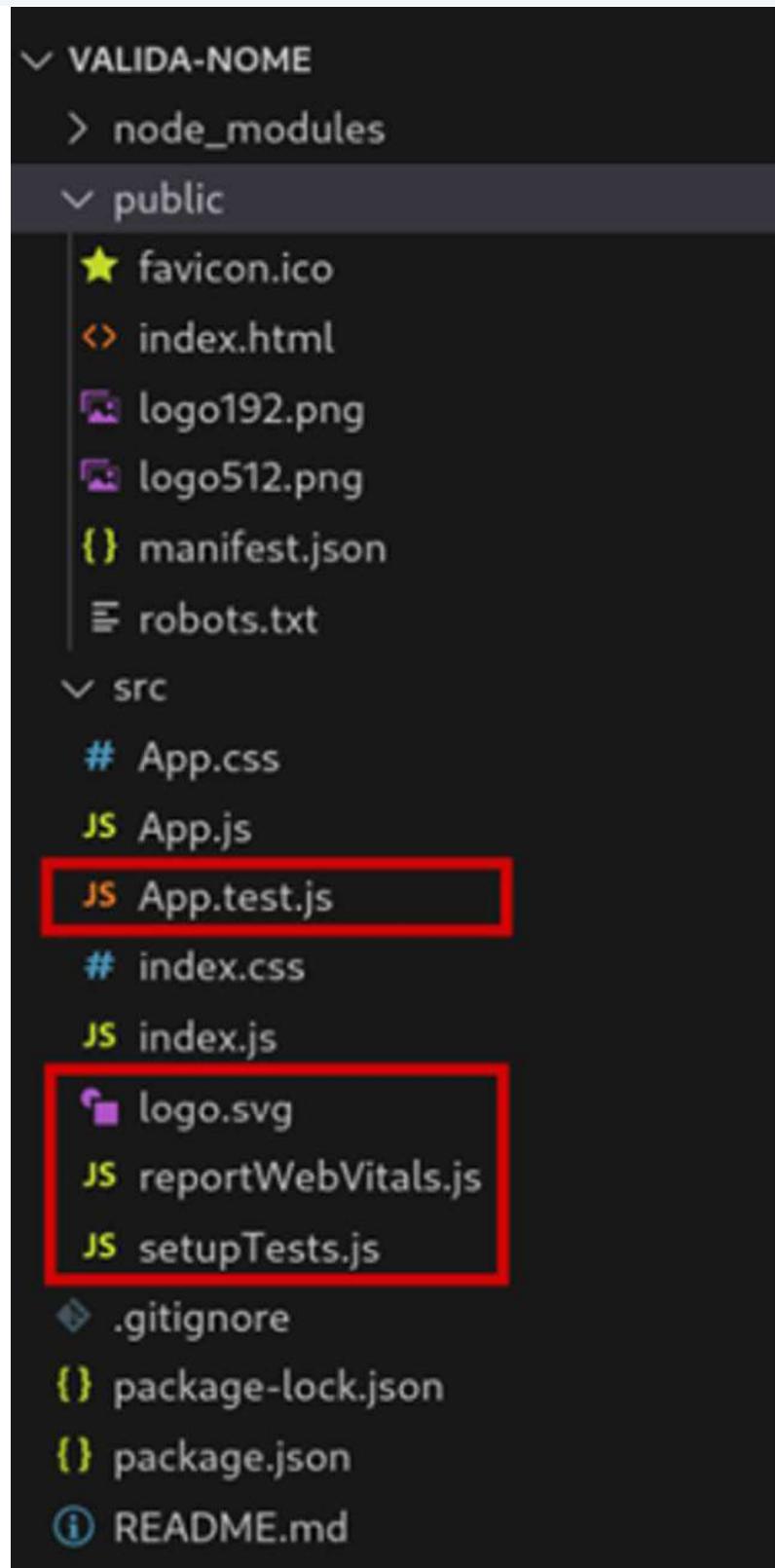


Figura 3 | Remover arquivos destacados. Fonte: captura de tela elaborada pelo autor.

O código do arquivo App.js ficará da seguinte forma:

# DESENVOLVIMENTO EM JAVASCRIPT

```
src > JS App.js > ...
1 import "./App.css";
2
3 function App() {
4     window.addEventListener("mousemove", (event) => {
5         const localiza = document.getElementById("position");
6         let posicaoX = document.getElementById("posicaoX");
7         let posicaoY = document.getElementById("posicaoY");
8         localiza.style.top = event.clientY + "px";
9         localiza.style.left = event.clientX + 5 + "px";
10        posicaoY.innerText = event.clientY + "px";
11        posicaoX.innerText = event.clientX + "px";
12    });
13
14    return (
15        <div className="App">
16            <header className="App-header">
17                <br />
18                <div id="position">
19                    <p>
20                        | x: <span id="posicaoX"></span>
21                    </p>
22                    <p>
23                        | y: <span id="posicaoY"></span>
24                    </p>
25                </div>
26            </header>
27        </div>
28    );
29}
30
31 export default App;
```

Figura 4 | Código do arquivo App.js. Fonte: elaborada pelo autor.

O código do arquivo App.css sofrerá algumas inclusões de código para estilizar a informação da posição do mouse na tela. Veja:

# DESENVOLVIMENTO EM JAVASCRIPT

```
src > # App.css > color
1 .App {
2   text-align: center;
3 }
4
5 .App-logo {
6   height: 40vmin;
7   pointer-events: none;
8 }
9
10 @media (prefers-reduced-motion: no-preference) {
11   .App-logo {
12     animation: App-logo-spin infinite 20s linear;
13   }
14 }
15
16 .App-header {
17   background-color: #rgb(223, 210, 173)39, 212);
18   min-height: 100vh;
19   display: flex;
20   flex-direction: column;
21   align-items: center;
22   justify-content: center;
23   font-size: calc(10px + 2vmin);
24   color: white;
25 }
26
27 .App-link {
28   color: #61dafb;
29 }
30
```

Figura 5 | Código do arquivo App.css - Parte 1. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
src > # App.css > ...  
31  @keyframes App-logo-spin {  
32    from {  
33      transform: rotate(0deg);  
34    }  
35    to {  
36      transform: rotate(360deg);  
37    }  
38  }  
39  *{  
40    margin: 0;  
41    padding: 0;  
42  }  
43  #position{  
44    width: 100px;  
45    height: 40px;  
46    color: ■rgb(0, 0, 0);  
47    padding: 5px;  
48    position: fixed;  
49    top: 0;  
50    left: 0;  
51  }  
52  body{  
53    background-color: □bisque;  
54  }  
55  p{  
56    margin: 0;  
57    font-family: sans-serif;  
58  }  
59  h2{  
60    text-align: center;  
61  }
```

Figura 6 | Código do arquivo App.css - Parte 2. Fonte: elaborada pelo autor

# DESENVOLVIMENTO EM JAVASCRIPT

Nosso projeto está finalizado. Para rodar e ver o resultado, execute o comando npm start no terminal do VSCode. O resultado será algo como o exibido na imagem a seguir:

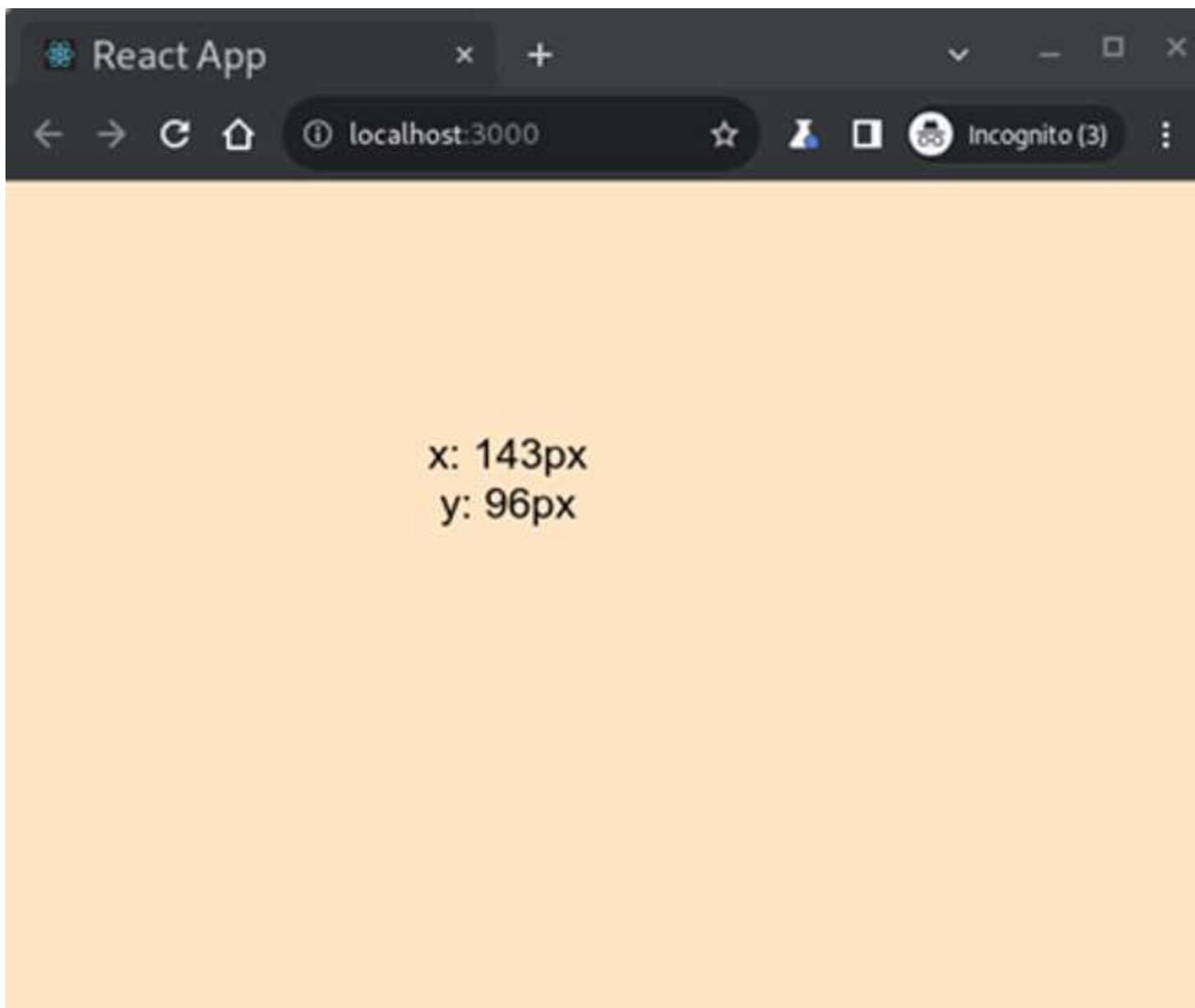


Figura 7 | Resultado do projeto. Fonte: elaborada pelo autor.

## Vamos Exercitar?

Para praticarmos nosso aprendizado, vamos ao desafio apresentado no início da aula, criação de uma aplicação com validação. Vamos criar uma aplicação chamada valida-nome, por meio do comando npx create-react-app valida-nome, e, ao final do processo, a estrutura do diretório do projeto.

Ao acessarmos o projeto pelo VSCode, de imediato, apagamos os mesmos arquivos que já vimos anteriormente: App.test.js, logo.svg, reportWebVitals.js e setupTests.js.

# DESENVOLVIMENTO EM JAVASCRIPT

Como sabemos, o primeiro arquivo que o React executa é o arquivo index.html, que está na pasta public; lá dentro, nós só temos uma div com id root, que é responsável por executar o código que está no arquivo index.js. Aliás, nele, você precisará remover umas linhas, e deverá ficar somente o código a seguir:

```
src > JS index.js > ...
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5
6
7 const root = ReactDOM.createRoot(document.getElementById('root'));
8 root.render(
9   <React.StrictMode>
10   |   <App />
11   </React.StrictMode>
12 );
```

Figura 8 | Código index.js. Fonte: elaborada pelo autor.

Esse arquivo renderiza o conteúdo do arquivo App.js, como pode ver entre as linhas 8 e 12, portanto, vamos alterar também o conteúdo de App.js e deixá-lo da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT

```
src > JS App.js > ...
1,   import './App.css';
2
3   function App() {
4     return (
5       <div className="App">
6         <header className="App-header">
7
8           </header>
9         </div>
10      );
11    }
12
13 export default App;
```

Figura 9 | Código App.js. Fonte: elaborada pelo autor.

Como pode ver, removemos o conteúdo que estava entre os elementos `<header>` e `</header>`. Aliás, também deve ser deletado o conteúdo da primeira linha, que faz a importação do logo, do projeto inicial, e é dentro desse espaço que vamos trabalhar com nosso código. Vamos inserir o seguinte código:

# DESENVOLVIMENTO EM JAVASCRIPT

```
src > JS App.js > ...
1,   import './App.css';
2
3   function App() {
4     return (
5       <div className="App">
6         <header className="App-header">
7           <div id="cadNome">
8             <h1>Validação</h1>
9             <input type="text" id="nome" />
10            <div id="msgerro"></div>
11            <button type="button" id="botao">Verificar</button>
12          </div>
13        </header>
14      </div>
15    );
16  }
17
18  export default App;
```

Figura 10 | Código App.js – cont. Fonte: elaborada pelo autor.

Ainda, dentro da função principal – a function App() –, nós vamos construir nossa função Javascript que fará a validação do campo. Essa validação consiste, basicamente, em verificar se o usuário inseriu o nome, e caso não tenha digitado nada e o botão “Verificar” for pressionado, será feita a validação, informando ao usuário que o nome precisa ser informado.

Logo depois da função principal, vamos inserir espaços e colocar o return mais para baixo, bem como inserir o código que fará a validação. O conteúdo desse arquivo ficará da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT

```
src > JS App.js > App
1 import "./App.css";
2
3 function App() {
4     const validaNome = () => {
5         let msg = document.getElementById("msgerro");
6         let nome = document.getElementById("nome");
7         if (nome.value.length === 0) {
8             msg.style.color = "red";
9             msg.textContent = "Informe o nome";
10        } else {
11            msg.textContent = "";
12            alert("Olá " + nome.value);
13        }
14    };
15
16    return (
17        <div className="App">
18            <header className="App-header">
19                <h1>Validação</h1>
20                <div>
21                    <label>Nome </label><br/>
22                    <input type="text" id="nome" />
23                    <div id="msgerro"></div>
24                    <button type="button" id="botao" onClick={validaNome}>
25                        Verificar
26                    </button>
27                </div>
28            </header>
29        </div>
30    );
31
32
33 export default App;
```

Figura 11 | Código App.js – cont. Fonte: elaborada pelo autor.

E para finalizar, vamos “limpar” o conteúdo do arquivo index.html e remover alguns comentários e linhas que não são mais necessários lá e esse arquivo fica na pasta public, do nosso projeto. Vamos aproveitar, também, para mudar o título da página que aparece no navegador e o idioma utilizado em nosso projeto; o resultado será o código apresentado a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

```
public > index.html > ...
1  <!DOCTYPE html>
2  <html lang="pt-BR">
3    <head>
4      <meta charset="utf-8" />
5      <link rel="icon" href="%PUBLIC_URL%/favicon.ico" />
6      <meta name="viewport" content="width=device-width, initial-scale=1" />
7      <meta name="theme-color" content="#000000" />
8      <meta
9        name="description"
10       content="Web site created using create-react-app"
11     />
12     <link rel="apple-touch-icon" href="%PUBLIC_URL%/logo192.png" />
13
14     <link rel="manifest" href="%PUBLIC_URL%/manifest.json" />
15
16     <title>Validação de nome</title>
17   </head>
18   <body>
19
20     <div id="root"></div>
21
22   </body>
23 </html>
```

Figura 12 | Resultado do código. Fonte: elaborada pelo autor.

Com isso, o projeto estará pronto e poderá ser testado. Para isso, lembre-se de que é necessário abrir o terminal do VSCode e executar o comando npm start, assim, teremos o resultado como o apresentado a seguir:

# DESENVOLVIMENTO EM JAVASCRIPT

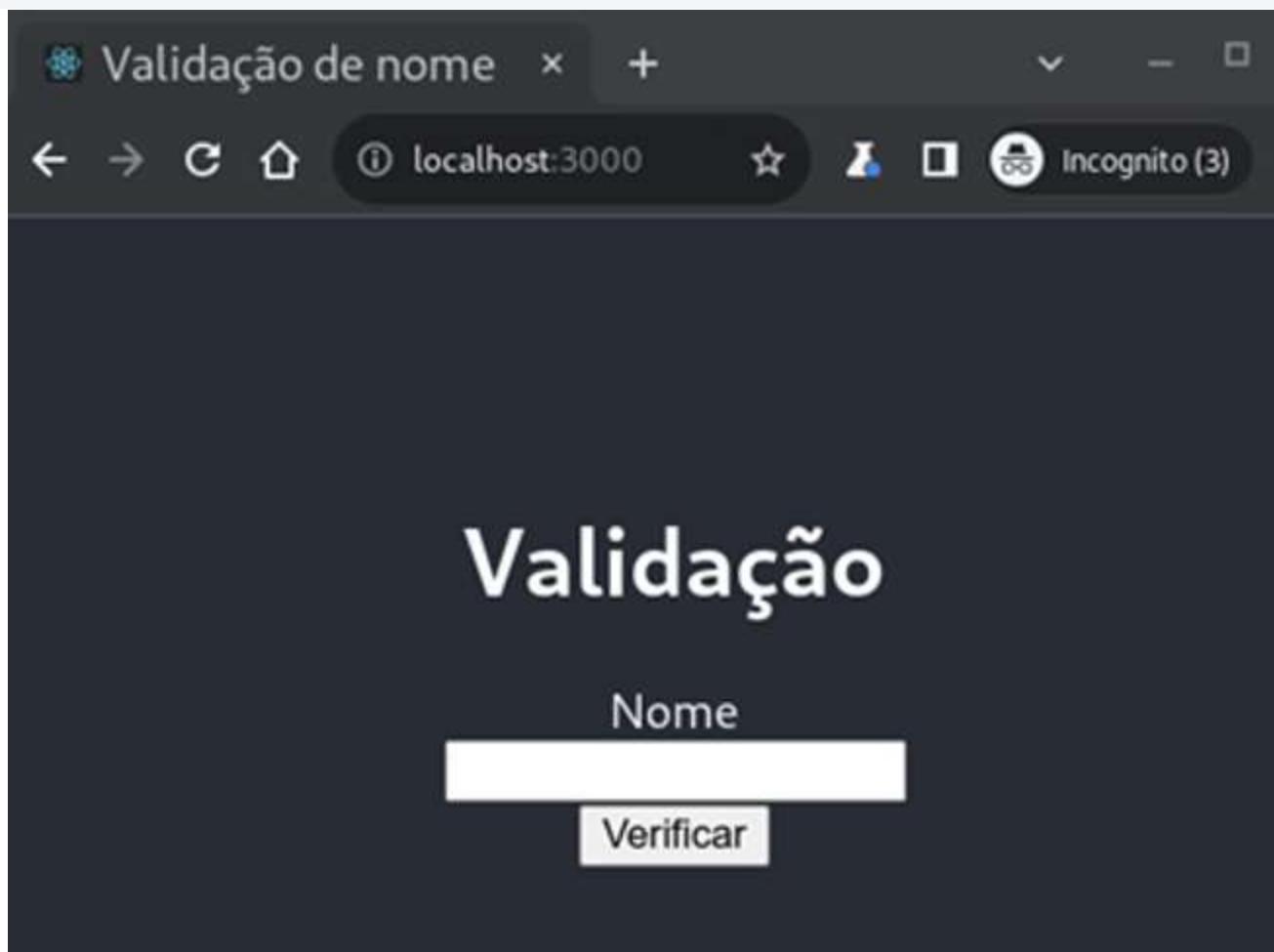


Figura 13 | Validação do nome. Fonte: elaborada pelo autor.

## Saiba mais

Sugerimos a leitura do Capítulo *React: desenvolvimento de componentes com framework do Facebook*, do livro *Frameworks Front End*.

MARCOLINO, A. da S. [Frameworks front end](#). São Paulo: Platos Soluções Educacionais S.A., 2021.

Leia também a respeito da criação de um primeiro componente no React.

REACT. [Seu Primeiro Componente](#). [s. d.].

## Referências

# DESENVOLVIMENTO EM JAVASCRIPT

MARCOLINO, A. da S. **Frameworks front end**. São Paulo: Platos Soluções Educacionais S.A., 2021.

MDN Web Docs. **Começando com React**. 2023. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Learn/Tools\\_and\\_testing/Client-side\\_JavaScript\\_frameworks/React\\_getting\\_started](https://developer.mozilla.org/pt-BR/docs/Learn/Tools_and_testing/Client-side_JavaScript_frameworks/React_getting_started). Acesso em: 12 nov. 2023.

REACT. **Quick start**. 2023. Disponível em: <https://pt-br.react.dev/learn>. Acesso em: 17 jan. 2023.

## Aula 5

Encerramento da Unidade

### Videoaula de Encerramento



#### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Estudante, esta videoaula foi preparada especialmente para você. Nela, você irá aprender conteúdos importantes para a sua formação profissional. Vamos assisti-la?

[Clique aqui](#) para acessar os slides da sua videoaula.

Bons estudos!

### Ponto de Chegada

Olá, estudante, o mercado de tecnologia é um mercado repleto de novidades e inovações que acontecem quase que instantaneamente, e nós temos visto, nos últimos tempos, o surgimento de tecnologias que estão ocupando um espaço significativo na vida das pessoas. Um bom exemplo é a famosa inteligência artificial, por meio de tecnologias lançadas no mercado, como é o caso do ChatGPT, apesar de a inteligência artificial ser mais antiga do que se pensa, tendo como início de desenvolvimento a década de 1950, com Dartmouth Summer Research Project on Artificial Intelligence, projeto desenvolvido por um colégio nos Estados Unidos (Silva et. al., 2019).

# DESENVOLVIMENTO EM JAVASCRIPT

Como vemos, essa evolução tecnológica não para, e tudo que é criado advém de constantes pesquisas e desenvolvimento de novos conhecimentos. Aliás, há uma demanda emergente e constante de pessoas com conhecimentos necessários para conduzir todo esse processo de construção de conhecimento. Por tudo isso, é muito importante sempre buscar novos conhecimentos e atualizações que lhe possibilitem uma boa colocação no mercado de trabalho. Conhecer novas ferramentas e tecnologias, como os vários frameworks que são criados, é fundamental para se tornar uma referência profissional nessa área.

Com a linguagem Javascript, frameworks como Angular, Vue.js e React são, sem dúvida nenhuma, os principais frameworks de mercado na opinião de profissionais da área (Camargos, 2019); eles estão moldando a forma como esses profissionais desenvolvem seus papéis no desenvolvimento de aplicações com a linguagem Javascript (Abdurakhimovich, 2023). Além disso, empresas e instituições de modo geral buscam pessoas com conhecimento em ferramentas como essas para alocação de projetos de clientes e parceiros, diante disso, busque estar sempre preparado para as oportunidades que surgirem. Portanto, os conteúdos estudados nesta Unidade são necessários para desenvolver a competência de reconhecer os principais frameworks no desenvolvimento com Javascript.

## É Hora de Praticar!



### Este conteúdo é um vídeo!

Para assistir este conteúdo é necessário que você acesse o AVA pelo computador ou pelo aplicativo. Você pode baixar os vídeos direto no aplicativo para assistir mesmo sem conexão à internet.

Agora que entendemos a importância que os Frameworks desempenham na atuação de profissionais que trabalham com desenvolvimento de aplicações, é chegada a hora de arregaçar as mangas e exercitar os seus conhecimentos.

Escolha um dos três principais frameworks de Javascript disponíveis (Angular, Vue.js e React) para montar uma aplicação que sirva de consulta a CEP na API da ViaCEP, mostrando, a partir dessa informação, endereço, bairro, cidade e estado.

Não é necessário montar componentes ou usar muitas diretivas nesse momento, pois não há a intenção de deixar essa tarefa muito longa, contudo, se você se sentir desafiado para isso, fique à vontade para colocar em prática todos os seus conhecimentos.

- Você é capaz de distinguir um framework de uma biblioteca e comprehende as sutis diferenças entre os principais frameworks de Javascript?
- Você é capaz de identificar, em sua região, qual dos frameworks de Javascript conhecidos é o mais utilizado por empresas?

# DESENVOLVIMENTO EM JAVASCRIPT

Para essa atividade, a resolução apresentada a seguir utilizará o React como framework, uma vez que vários estudos publicados na área mostram que ele tem sido o mais utilizado nos últimos tempos.

Como pedido no enunciado, não construiremos componentes, para que esse processo de criação do projeto no framework escolhido fique ainda mais familiar, mas a prática é fundamental para isso. Caso queira, adapte o código apresentado e coloque-o em componentes, para melhorar ainda mais sua aplicação.

A criação do projeto será feita com o comando `npx create-react-app consulta-cep`; após construídas e instaladas as dependências necessárias, abriremos o projeto no VSCode. De imediato, removeremos arquivos que não utilizaremos, e os arquivos são os marcados na imagem a seguir:

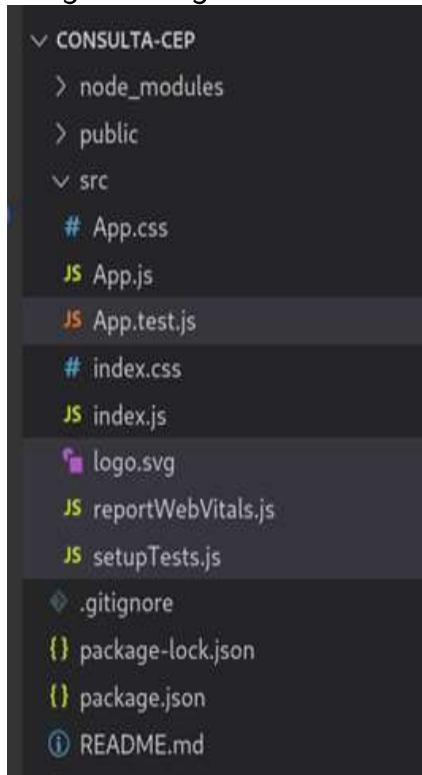


Figura 1 | Remover arquivos destacados. Fonte: captura de tela elaborada pelo autor.

Será necessário, também, remover resquícios dos arquivos que foram deletados. No arquivo `App.js`, remova a primeira linha, iniciada com `import logo...`, que busca um dos arquivos removidos. Depois, vamos deixar apenas a `div` e apagar o conteúdo. O arquivo deverá ficar assim:

# DESENVOLVIMENTO EM JAVASCRIPT

```
JS App.js M X
src > JS App.js > ...
1, import './App.css';
2
3 function App() {
4   return (
5     <div className="App">
6
7     </div>
8   );
9 }
10
11 export default App;
```

Figura 2 | App.js. Fonte: elaborada pelo autor.

Agora, no arquivo index.js, também vamos limpar algumas linhas e remover, da mesma maneira, algumas linhas de import. O arquivo deverá ficar da seguinte forma:

```
JS index.js M X
src > JS index.js > ...
1 import React from 'react';
2 import ReactDOM from 'react-dom/client';
3 import './index.css';
4 import App from './App';
5
6 const root = ReactDOM.createRoot(document.getElementById('root'));
7 root.render(
8   <React.StrictMode>
9     <App />
10    </React.StrictMode>
11 );
```

Figura 3 | index.js. Fonte: elaborada pelo autor.

Agora, no arquivo App.js, vamos inserir o código do HTML, e como se trata de um exercício para consumo de API da ViaCEP, utilizaremos quatro campos: CEP, endereço, bairro, cidade e estado. Quando digitarmos o número do CEP e sairmos do campo utilizando a tecla TAB ou clicando com o mouse em outro local, os demais campos serão preenchidos automaticamente. O arquivo deverá ficar da seguinte maneira:

# DESENVOLVIMENTO EM JAVASCRIPT

```
25 App.js M X
src > JS App.js > [0] default
1 | import "./App.css";
2 |
3 | function App() {
4 |   const preencherFormulario = (endereco) => {
5 |     document.getElementById("rua").value = endereco.logradouro;
6 |     document.getElementById("bairro").value = endereco.bairro;
7 |     document.getElementById("cidade").value = endereco.localidade;
8 |     document.getElementById("estado").value = endereco.uf;
9 |   };
10 |   const eNumero = (numero) => /^[0-9]+$/ .test(numero);
11 |   const cepValido = (cep) => cep.length === 8 && eNumero(cep);
12 |
13 |   const pesquisarCep = async () => {
14 |     const url = 'https://viacep.com.br/ws/${document.getElementById('cep').value}/json/';
15 |
16 |     if (cepValido(document.getElementById('cep').value)) {
17 |       const dados = await fetch(url);
18 |       const adres = await dados.json();
19 |
20 |       if (adres.hasOwnProperty("erro")) {
21 |         alert("CEP não encontrado!");
22 |       } else {
23 |         preencherFormulario(adres);
24 |       }
25 |     } else {
26 |
27 |       alert("CEP incorreto!");
28 |     }
29 |   };
30 |   document.getElementById('cep').addEventListener('focusout', pesquisarCep);
}
```

Figura 4 | Título - Parte 1  
Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

```
31 |
32     return (
33         <div className="App">
34             <form className="App-header">
35                 <label>CEP</label>
36                 <input type="text" id="cep" />
37                 <br />
38                 <label>Rua</label>
39                 <input type="text" id="rua" />
40                 <br />
41                 <label>Bairro</label>
42                 <input type="text" id="bairro" />
43                 <br />
44                 <label>Cidade</label>
45                 <input type="text" id="cidade" />
46                 <br />
47                 <label>Estado</label>
48                 <input type="text" id="estado" />
49                 <br />
50                 <br />
51                 <br />
52                 <button onclick="">Entrar</button>
53             </form>
54         </div>
55     );
56 }
57
58 export default App;
```

Figura 5 | Título - Parte 2. Fonte: elaborada pelo autor.

Como você pode ver, foram construídas 4 funções. A primeira, chamada preencherFormulario, é responsável por obter o dado que vem da API ViaCEP e inseri-lo no campo correto; já a função eNumero é responsável por aplicar ao número informado no campo CEP uma regra de expressão regular que verifica se só foram informados números (MDN, 2023); a função seguinte, cepValido, verifica se a quantidade de números informado é igual a 8 e chama a função criada anteriormente, passando para ela como argumento o número do CEP; por fim, a função principal, pesquisarCep, é uma função do tipo async, ou seja, uma função assíncrona que faz a consulta à API da ViaCEP e, se tudo estiver certo, retorna os dados em formato JSON da consulta, caso contrário, apresenta um erro.

Ao longo de nossos estudos, vimos todos os conceitos relacionados aos frameworks de mercado disponíveis para o desenvolvimento de aplicações e baseados na linguagem Javascript.

# DESENVOLVIMENTO EM JAVASCRIPT

O percurso leva em consideração mencionar os principais frameworks Javascript de mercado, aqueles que são opções claras de profissionais da área.

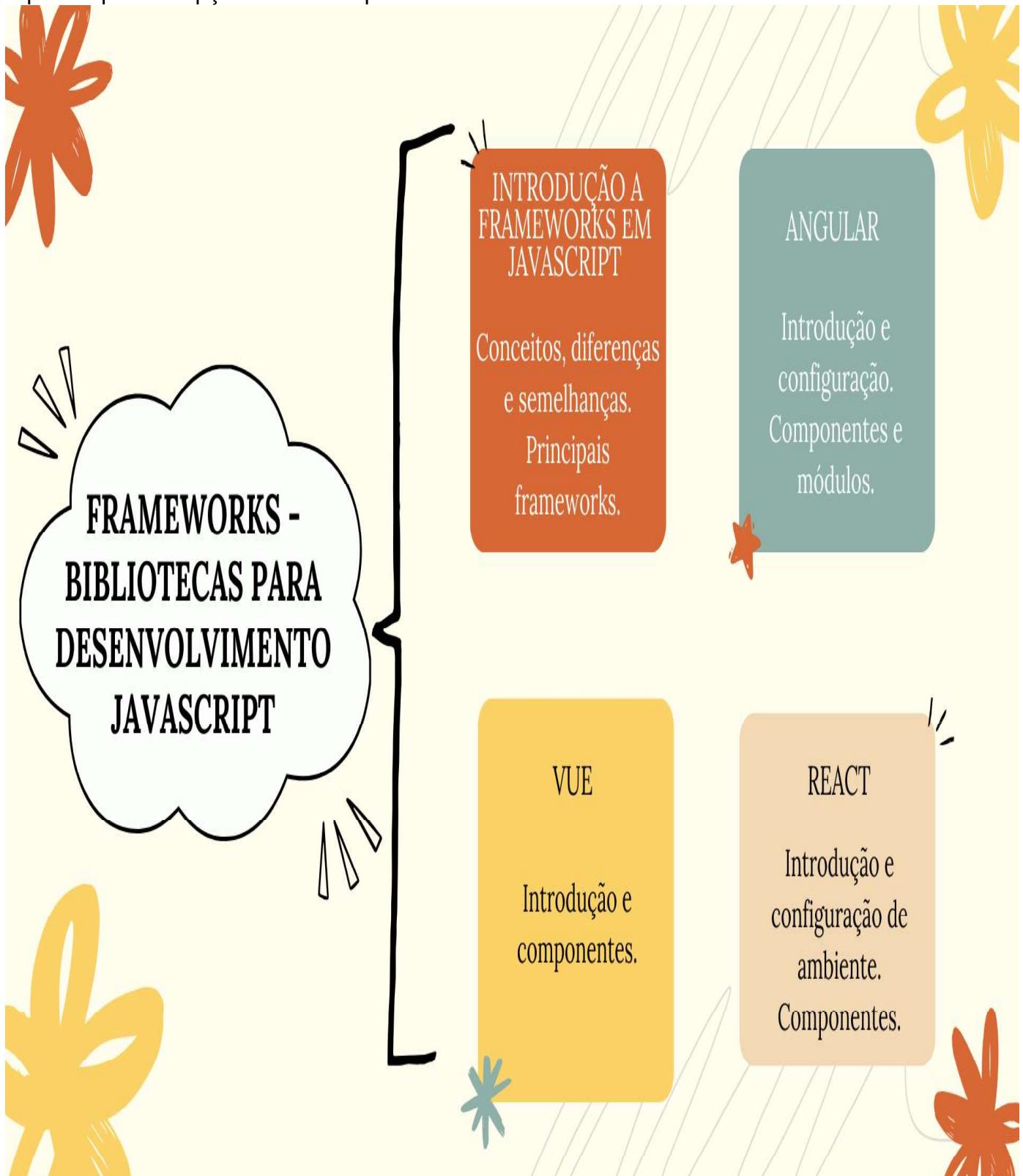


Figura | Frameworks – Bibliotecas para desenvolvimento Javascript. Fonte: elaborada pelo autor.

# DESENVOLVIMENTO EM JAVASCRIPT

ABDURAKHIMOVICH, U. A. The future of javascript: emerging trends and technologies. **Formation of psychology and pedagogy as interdisciplinary sciences**, [S. l.], v. 2, n. 21, p. 12-14, 2023.

CAMARGOS, J. G. C. *et al.* Uma análise comparativa entre os frameworks javascript angular e React. **Computação & Sociedade**, Belo Horizonte, v. 1, n. 1, 2019.

MDN WEB DOCS. **Expressões regulares**. 2023. Disponível em: [https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Regular\\_expressions](https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Regular_expressions). Acesso em: 17 jan. 2024.

SILVA, F. M. *et al.* **Inteligência artificial**. Porto Alegre: SAGAH, 2019.