

Roteiro

JavaScript

Tópicos

Opção 1

- DOM
- Tipos de dados
- Variáveis
- Operadores
- Estrutura de controles
- Objetos
- Arrays
- Funções
- Objetos customizados
- Closures
- Arrow functions
- Classes (propriedades, métodos e herança e entender o "this")
- let e const

Pré-Requisitos – React

<https://reactjs.org/tutorial/tutorial.html>

Prerequisites

We'll assume that you have some familiarity with HTML and JavaScript, but you should be able to follow along even if you're coming from a different programming language. We'll also assume that you're familiar with programming concepts like functions, objects, arrays, and to a lesser extent, classes.

If you need to review JavaScript, we recommend reading [this guide](#). Note that we're also using some features from ES6 — a recent version of JavaScript. In this tutorial, we're using [arrow functions](#), [classes](#), [let](#), and [const](#) statements. You can use the [Babel REPL](#) to check what ES6 code compiles to.

Links

<https://www.w3schools.com/js/default.asp>

<https://flaviocopes.com/javascript-introduction/>

<https://medium.com/trainingcenter/afinal-javascript-e-ecmascript-s%C3%A3o-a-mesma-coisa-498374abbc47>

<https://medium.freecodecamp.org/javascript-naming-conventions-dos-and-don-ts-99c0e2fdd78a>

O que é JavaScript?

Linguagem de programação

JavaScript versions

Let me introduce the term *ECMAScript* here. We have a complete guide dedicated to [ECMAScript](#) where you can dive into it more, but to start with, you just need to know that ECMAScript (also called **ES**) is the name of the JavaScript standard.

JavaScript is an implementation of that standard. That's why you'll hear about [ES6](#), [ES2015](#), [ES2016](#), [ES2017](#), [ES2018](#) and so on.

For a very long time, the version of JavaScript that all browser ran was ECMAScript 3. Version 4 was canceled due to feature creep (they were trying to add too many things at once), while ES5 was a huge version for JS.

[ES2015](#), also called [ES6](#), was huge as well.

Since then, the ones in charge decided to release one version per year, to avoid having too much time idle between releases, and have a faster feedback loop.

Currently, the latest approved JavaScript version is [ES2017](#).

Onde?

Back-end, Mobile, Front-End, Banco de Dados

Como?

Criar repositório GitHub – 3 sprint

2s2019-t2-sprint-3-frontend

Subir apresentação (somente do javascript) e pasta: 'gufos.base' (que contém somente o css e o html).

Apresentação – 40 minutos

HTML -> Estrutura

CSS -> Estilização

JavaScript -> Comportamento

https://cdn-images-1.medium.com/max/1600/0*ILZZpnli_R1FU3p4.gif

http-server

<https://imasters.com.br/desenvolvimento/escopos-em-javascript>

DOM

Representa o documento HTML na página.

Retomar o backend e quais endpoints que vocês irão construir.

Api/login

Api/categorias

C:_github\senai-dev-2s2019\sprint-3-frontend\backend.projeto\Senai.Gufos.WebApi\Senai.Gufos.WebApi

Abra o cmd e digite dotnet run.

Localhost:5000/Swagger

Fazê-los pensar quais passos deverão ser feitos para eles conseguirem enviar os dados do frontend para o backend.

Início

Manipulação do DOM e buscando a referência do botão.

[categorias.dom.html](#)

```

<script>
  // imprimir o documento
  var documento = document;
  console.log(documento);
  // buscar por nome da classe
  var vInputCategoria = document.getElementsByClassName('class__categoria');
  console.log(vInputCategoria);
  // buscar pelo id
  var vInputCategoria = document.getElementById('input__categoria');
  console.log(vInputCategoria);
  // buscar pelo queryselector e pela classe - introducao ao let
  let lInputCategoria = document.querySelector('.class__categoria');
  console.log(lInputCategoria);
  // irá aparecer um erro , por causa do reassign da variavel let
  let lInputCategoria = document.querySelector('#input__categoria');
  console.log(lInputCategoria);
  // nao declaro novamente o let, mas consigo trocar o valor da variavel
  lInputCategoria = document.querySelector('#input__categoria');
  console.log(lInputCategoria);
  // const e busca pelo id
  const cInputCategoria = document.querySelector('#input__categoria');
  console.log(cInputCategoria);
  // invalid assignment - falar sobre os 3 tipos de declaracoes
  cInputCategoria = document.querySelector('.class__categoria');
  console.log(cInputCategoria);
</script>

```

[categorias.botao.html](#)

Iniciar este exercício somente buscando a referência do botão anterior.

```

<script>
  const cInputCategoria = document.querySelector('#input__categoria');
  console.log(cInputCategoria);
</script>

```

Fazer a alteração para criar um cadastrarCategoria() dentro do script.

```

<button id="btn_cadastrar"
class="conteudoPrincipal-btn conteudoPrincipal-btn-cadastro"
onclick="cadastrarCategoria()">
  Cadastrar
</button>
</div>
</form>
</div>
</section>
</main>

<footer class="rodapePrincipal">
  <section class="rodapePrincipal-patrocinadores">
    <div class="container">
      <p>Escola SENAI de Informática - 2019</p>
    </div>
  </section>
</footer>
</div>

<script>

  const cInputCategoria = document.querySelector('#input_categoria');
  function cadastrarCategoria() {
    console.log(cInputCategoria);
  }

```

1

2

Evitar o comportamento padrão da página.

```

53     <div class="container">
54         <input type="text" class="class__categoria" id="input_categoria"
55         <button id="btn_cadastrar"
56         class="conteudoPrincipal-btn conteudoPrincipal-btn-cadastro"
57         onclick="cadastrarCategoria()">
58             Cadastrar
59         </button>
60     </div>
61 </form>
62 </div>
63 </section>
64 </main>
65
66 <footer class="rodapePrincipal">
67     <section class="rodapePrincipal-patrocinadores">
68         <div class="container">
69             <p>Escola SENAI de Informática - 2019</p>
70         </div>
71     </section>
72 </footer>
73 </div>
74
75 <script>
76
77     const cInputCategoria = document.querySelector('#input_categoria');
78     function cadastrarCategoria() {
79         event.preventDefault();
80         console.log(cInputCategoria);
81     }
82

```

Colocar um evento para que seja escutado.

```

51     </h2>
52     <form>
53         <div class="container">
54             <input type="text" class="class_categoria" id="input_categoria" p;
55             <button id="btn_cadastrar" class="conteudoPrincipal-btn conteudoPr;
56                 <!-- onclick="cadastrarCategoria()" -->
57                 Cadastrar
58             </button>
59         </div>
60     </form>
61 </div>
62 </section>
63 </main>
64
65 <footer class="rodapePrincipal">
66     <section class="rodapePrincipal-patrocinadores">
67         <div class="container">
68             <p>Escola SENAI de Informática - 2019</p>
69         </div>
70     </section>
71 </footer>
72 </div>
73
74 <script>
75
76     const cInputCategoria = document.querySelector('#input_categoria');
77     // function cadastrarCategoria() {
78     //     event.preventDefault();
79     //     console.log(cInputCategoria);
80     // }
81
82     const btnCadastrar = document.querySelector('#btn_cadastrar');
83
84     btnCadastrar.addEventListener("click", function (event) {
85         event.preventDefault();
86         console.log(cInputCategoria);
87     });
88 </script>

```

Handwritten annotations: A blue arrow points from the button text "Cadastrar" (line 57) to the `addEventListener` call (line 84). A blue "1" is next to the button text. A blue "2" is next to the `const cInputCategoria` line (line 76). A blue "3" is next to the `console.log(cInputCategoria);` line (line 86).

Para buscar o valor que o usuário digitou.

```

<script>

const cInputCategoria = document.querySelector('#input_categoria');
// function cadastrarCategoria() {
//     event.preventDefault();
//     console.log(cInputCategoria);
// }

const btnCadastrar = document.querySelector('#btn_cadastrar');

btnCadastrar.addEventListener("click", function (event) {
    event.preventDefault();
    console.log(cInputCategoria);
    console.log(cInputCategoria.value);
});
</script>

```

Handwritten annotations: A blue arrow points from the `console.log(cInputCategoria.value);` line (line 17) to the `value` property access (line 17). A blue "1" is next to the `value` property access.

[categorias.input.evento.html](#)

Validando o valor de input do usuário.

Observação: keyup não é obrigatório. Após, terá atividade com desafios para eles verem essa parte.

```
<script>

  const cInputCategoria = document.querySelector('#input__categoria');

  cInputCategoria.addEventListener("keyup", function () {
    if (cInputCategoria.value.length < 3) {
      cInputCategoria.style.border = "thick solid red";
    } else {
      cInputCategoria.style.border = "thick solid #e6e6e6";
    }
  });

  const btnCadastrar = document.querySelector('#btn__cadastrar');

  btnCadastrar.addEventListener("click", function (event) {
    event.preventDefault();
    console.log(cInputCategoria);
    console.log(cInputCategoria.value);
  });

</script>
```

4 1

[categorias.post.html](#)

Construir o objeto que será enviado para a API.


```

<script>

const cInputCategoria = document.querySelector('#input__categoria');

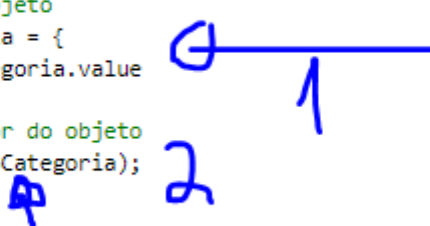
cInputCategoria.addEventListener("keyup", function () {
  if (cInputCategoria.value.length < 3) {
    cInputCategoria.style.border = "thick solid red";
  } else {
    cInputCategoria.style.border = "thick solid #e6e6e6";
  }
});

const btnCadastrar = document.querySelector('#btn__cadastrar');

btnCadastrar.addEventListener("click", function (event) {
  event.preventDefault();
  // construir um objeto
  let objetoCategoria = {
    nome: cInputCategoria.value
  };
  // imprimir o valor do objeto
  console.log(objetoCategoria);
});

</script>

```



Criar o evento que será feito para a API.

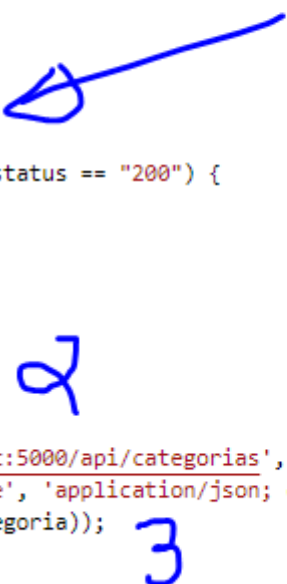
```

btnCadastrar.addEventListener("click", function (event) {
  event.preventDefault();
  // construir um objeto
  let objetoCategoria = {
    nome: cInputCategoria.value
  };
  // imprimir o valor do objeto
  console.log(objetoCategoria);

  // criar uma requisição
  var xhr = new XMLHttpRequest();
  xhr.onload = function () {
    if (xhr.readyState == 4 && xhr.status == "200") {
      console.log('Sucesso');
    } else {
      console.log('Erro');
    }
  };
  xhr.onerror = function () {

  };
  xhr.open('POST', 'http://localhost:5000/api/categorias', true);
  xhr.setRequestHeader('Content-Type', 'application/json; charset=utf-8');
  xhr.send(JSON.stringify(objetoCategoria));
});

```



Exercícios

3.frontend.js.ex1.validacoes.docx – formulários de input, validações de campo

3.frontend.js.ex1.post.docx – sstop – api, post

categorias.html

Manipulação do DOM e declaração de variáveis.

API

Mostrar uma lista fixa na tabela – criando elementos e mostrando na tela.

```
<script>

  let categorias = [{ idCategoria: 1, nome: "Categoria A" },
    { idCategoria: 2, nome: "Categoria B" },
    { idCategoria: 3, nome: "Categoria C" }];

  const tabela = document.getElementById("tabela-lista-corpo");
  categorias.forEach(element => {
    let linha = document.createElement("tr");
    let itemDaLinhaIdCategoria = document.createElement("td");
    itemDaLinhaIdCategoria.innerText = element.idCategoria;
    let itemDaLinhaNomeCategoria = document.createElement("td");
    itemDaLinhaNomeCategoria.innerText = element.nome;
    linha.appendChild(itemDaLinhaIdCategoria);
    linha.appendChild(itemDaLinhaNomeCategoria);
    tabela.appendChild(linha);
  });
</script>
```

C A T E G O R I A S	
#	Título
1	Categoria A
2	Categoria B
3	Categoria C

```
let listaTiposEventos = [{ "id" : 1, "nome" : "Marketing Digital" }, { "id": 2, "nome": "Tecnologia" }, { "id": 3, "nome": "Design" }];
var listing_table = document.getElementById("tabela-lista-corpo");

listaTiposEventos.forEach(element => {
  let linha = document.createElement("tr");
  let itemDaLinhaId = document.createElement("td");
  itemDaLinhaId.innerText = element.id;
  let itemDaLinhaNome = document.createElement("td");
  itemDaLinhaNome.innerText = element.nome;
  let itemDaLinhaOpcao = document.createElement("td");
  itemDaLinhaOpcao.innerText = "Edição";
  linha.appendChild(itemDaLinhaId);
  linha.appendChild(itemDaLinhaNome);
  linha.appendChild(itemDaLinhaOpcao);
  listing_table.appendChild(linha);
});
```

Comunicando com a API.

<https://medium.com/beginners-guide-to-mobile-web-development/the-fetch-api-2c962591f5c>

```

1  function success() {
2      var data = JSON.parse(this.responseText);
3      console.log(data);
4  }
5
6  function error(err) {
7      console.log('Error Occurred :', err);
8  }
9
10 var xhr = new XMLHttpRequest();
11 xhr.onload = success;
12 xhr.onerror = error;
13 xhr.open('GET', 'https://api.github.com/users/swapnilbangare');
14 xhr.send();

```

XMLHttpRequest.js hosted with ❤ by GitHub

[view raw](#)

<https://developers.google.com/web/fundamentals/primers/promises>

XML é somente modelo caso eles queiram.

XMLHttpRequest

```

var xhr = new XMLHttpRequest();
// em caso de sucesso
xhr.onload = sucesso;
// em caso de erro
xhr.onerror = error;
// verbo + url
xhr.open('GET', 'http://localhost:5000/api/categorias');
xhr.send();

```

Adicionar as duas funções, de erro e sucesso.

```

function sucesso() {
    var data = JSON.parse(this.responseText);
    console.log(data);
}

function error(err) {
    console.log('Um erro ocorreu: ', err);
}

var xhr = new XMLHttpRequest();
// em caso de sucesso
xhr.onload = sucesso;
// em caso de erro
xhr.onerror = error;
// verbo + url
xhr.open('GET', 'http://localhost:5000/api/categorias');
xhr.send();

```

Criar uma função e atribuir o valor como parâmetro de entrada da função.

```

function sucesso() {
  var data = JSON.parse(this.responseText);
  console.log(data);
  preencherTabela(data);
}

function error(erro) {
  console.log('Um erro ocorreu: ', erro);
}

var xhr = new XMLHttpRequest();
// em caso de sucesso
xhr.onload = sucesso;
// em caso de erro
xhr.onerror = error;
// verbo + url
xhr.open('GET', 'http://localhost:5000/api/categorias');
xhr.send();

const tabela = document.getElementById("tabela-lista-corpo");
function preencherTabela(categorias) {
  categorias.forEach(element => {
    let linha = document.createElement("tr");
    let itemDaLinhaIdCategoria = document.createElement("td");
    itemDaLinhaIdCategoria.innerText = element.idCategoria;
    let itemDaLinhaNomeCategoria = document.createElement("td");
    itemDaLinhaNomeCategoria.innerText = element.nome;
    linha.appendChild(itemDaLinhaIdCategoria);
    linha.appendChild(itemDaLinhaNomeCategoria);
    tabela.appendChild(linha);
  });
}

```

FETCH

Listando tipos de eventos utilizando a Fetch API.

```
const tabela = document.getElementById("tabela-lista-corpo");

const URL = 'http://localhost:5000/api/categorias';
fetch(URL)
  .then(resposta => resposta.json())
  .then(data => preencherTabela(data))
  .catch(erro => console.error(error));

function preencherTabela(categorias) {
  categorias.forEach(element => {
    let linha = document.createElement("tr");
    let itemDaLinhaIdCategoria = document.createElement("td");
    itemDaLinhaIdCategoria.innerText = element.idCategoria;
    let itemDaLinhaNomeCategoria = document.createElement("td");
    itemDaLinhaNomeCategoria.innerText = element.nome;
    linha.appendChild(itemDaLinhaIdCategoria);
    linha.appendChild(itemDaLinhaNomeCategoria);
    tabela.appendChild(linha);
  });
}
```

<https://braziljs.org/blog/fetch-api-e-o-javascript/>

Exercícios

3.frontend.js.ex1.get.docx - sstop

3.frontend.js.ex4.get.docx - personagens

arrays

categorias.arrays.html

```
let tiposEventos = ["Evento A", "Evento B", "Evento C"];
console.log(tiposEventos[1]);
console.log(tiposEventos.length);
```

```
for (let i = 0; i < tiposEventos.length; i++)
{
  console.log(tiposEventos[i]);
}
```

```
tiposEventos.forEach(element => {
  console.log(element);
});
```

PUSH/POP

```
// ===== PUSH/POP =====

tiposEventos.push("Evento D");
console.log(tiposEventos);

tiposEventos.pop();
console.log(tiposEventos);
```

Funções pré-definidas

```
// ===== ARRAYS =====

console.log(tiposEventos.join(" - "));
console.log(tiposEventos.sort());
console.log(tiposEventos.reverse());
```

MAP

```
let numerosComDescricao = [{ numero: 1, descricao: 'um'}, { numero: 2, descricao: 'dois' }, { numero: 3, descricao: 'tres' }];
let somenteNumeros = [];

// utilizando for
for (let i = 0; i < numerosComDescricao.length; i++)
{
  somenteNumeros.push(numerosComDescricao[i].numero);
}
console.log(somenteNumeros);

// começando com map
somenteNumeros = numerosComDescricao.map(function(cadaNumeroDaLista) {
  return cadaNumeroDaLista.numero;
});
console.log(somenteNumeros);

// remover funcao
somenteNumeros = numerosComDescricao.map((cadaNumeroDaLista) => {
  return cadaNumeroDaLista.numero;
});
console.log(somenteNumeros);

// remover parenteses
somenteNumeros = numerosComDescricao.map(cadaNumeroDaLista => {
  return cadaNumeroDaLista.numero;
});
console.log(somenteNumeros);

// removendo o return
somenteNumeros = numerosComDescricao.map(cadaNumeroDaLista => cadaNumeroDaLista.numero);
console.log(somenteNumeros);
```

FILTER

```
let numeros = [10, 20, 30, 40];

let numerosMaiorDoQueTrinta = numeros.filter(numeroDaLista => numeroDaLista >= 30);
console.log(numerosMaiorDoQueTrinta);

let numeroBuscado = numeros.filter(numeroEscolhido => numeroEscolhido == 10);
console.log(numeroBuscado);
```

REDUCE

```
let numeros = [10, 20, 30, 40];
let soma = numeros.reduce((total, value) => total + value);
console.log(soma);
```

Exercícios

3.frontend.js.ex2.docx

Extra

Cadastrando uma categoria com FETCH.

Realize o cadastro de um tipo de evento.

```
<div class="container" id="conteudoPrincipal-cadastro">
  <h2 class="conteudoPrincipal-cadastro-titulo">
    Cadastrar Categoria
  </h2>
  <form>
    <div class="container">
      <input type="text" class="class__categoria" id="input__categor
      <button id="btn_cadastrar" class="conteudoPrincipal-btn conte
        onclick="cadastrarCategoria()" >
          Cadastrar
      </button>
    </div>
  </form>
</div>
</section>
</main>

<footer class="rodapePrincipal">
  <section class="rodapePrincipal-patrocinadores">
    <div class="container">
      <p>Escola SENAI de Informática - 2019</p>
    </div>
  </section>
</footer>
</div>

<script>

  function cadastrarCategoria() {
    event.preventDefault();
    console.log(document.getElementById("input__categoria").value);
  }

</script>
```

Construir o JSON.

```

<div class="container" id="conteudoPrincipal-cadastro">
  <h2 class="conteudoPrincipal-cadastro-titulo">
    Cadastrar Categoria
  </h2>
  <form>
    <div class="container">
      <input type="text" class="class__categoria" id="input__categoria" ;
      <button id="btn__cadastrar" class="conteudoPrincipal-btn conteudoPr
        onclick="cadastrarCategoria()" >
        Cadastrar
      </button>
    </div>
  </form>
</div>
</section>
</main>

<footer class="rodapePrincipal">
  <section class="rodapePrincipal-patrocinadores">
    <div class="container">
      <p>Escola SENAI de Informática - 2019</p>
    </div>
  </section>
</footer>
</div>

<script>

function cadastrarCategoria() {
  // evitar o comportamento padrao da pagina
  event.preventDefault();
  // pegar o valor que o usuario digitou
  console.log(document.getElementById("input__categoria").value);
  // construir o json para ser enviado
  let objetoCategoria = {
    nome: document.getElementById("input__categoria").value
  };
  console.log(objetoCategoria);
}

</script>

```

Fazer o envio dos dados a API.

Com Fetch.

<script>

```
function cadastrarCategoria() {  
  // evitar o comportamento padrao da pagina  
  event.preventDefault();  
  // pegar o valor que o usuario digitou  
  console.log(document.getElementById("input__categoria").value);  
  // construir o json para ser enviado  
  let objetoCategoria = {  
    nome: document.getElementById("input__categoria").value  
  };  
  console.log(objetoCategoria);  
  
  fetch('http://localhost:5000/api/categorias', {  
    method: 'POST',  
    body: JSON.stringify(objetoCategoria),  
    headers: {  
      'Content-Type': 'application/json'  
    }  
  }).then(response => response)  
    .then(data => console.log(data))  
    .catch(error => console.log(error));  
}
```

</script>