
DIVERSIDADE

Barbara Nery
Comunidade

Coordenadora de T.I.

Especialista SharePoint On-Premises
IT Service Management

https://www.linkedin.com/in/barbara-fernandes/
https://www.instagram.com/_barbara.js/
.

JavaScript
Aula 01

01 Definição:
- 01.1 O que significa script;
- 01.2 O que significa cliente;
- 01.3 O que significa Interpretado.

02 Origem;
03 Bibliotecas e frameworks;
04 Quem é o Javascript na fila do pão no front-end?
05 JavaScript no Front e no Server (Node);
06 Dinâmica no site do google – Apresentação da “função inspecionar elemento” .

JavaScript
Aula 02

01 JavaScript no lado do cliente - IDEs.
02 Estrutura HTML e CSS;
03 DOM:
04.Boas Práticas JS;
05 Construção de uma calculadora;
06 Construção de um calendário.

JavaScript
Aula 03

01 Caixinha de respostas – Dúvidas enviadas pelos Toters durante a semana;
02 Um pouco sobre API;
03 JavaScript Fecth Api (Promise, response e Then);
04 Contrução do código Javascript para consumir a API do Pokemon;
05 Exercício: Um Toter refazer o código.

JavaScript
Aula 04

01 Caixinha de respostas – Dúvidas enviadas pelos Toters durante a aula passada.
02 Revisão das Aulas 2 e 3;
03 Adaptação do código de consulta de API para consumir a API SW;
04 Exercício: Modificar o retorno da consulta.

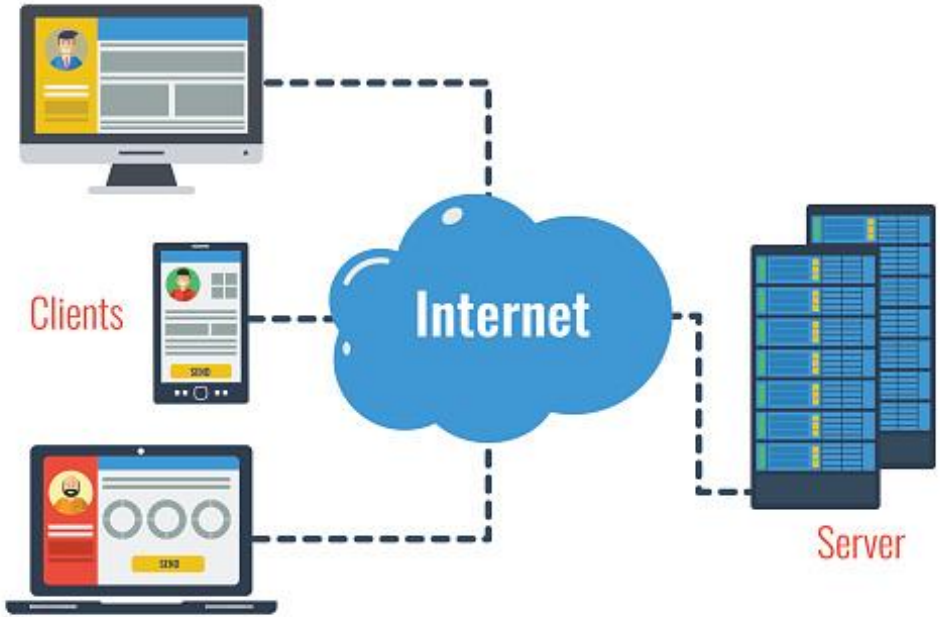

JS
“Que é,
onde vive,
de que se alimenta?"

AULA 01

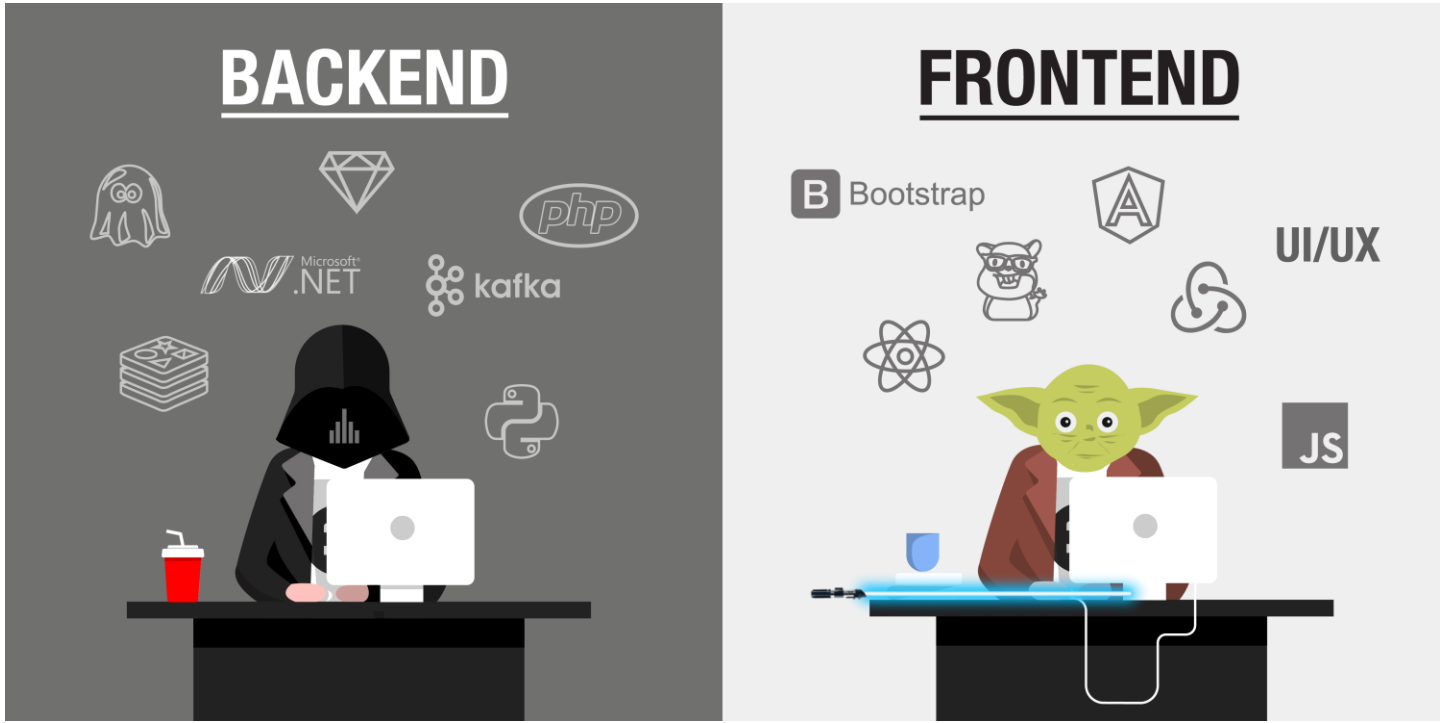
alert(“JavaScript – O que é?”);

JavaScript é uma linguagem de programação de scripts[1] e que é interpretado[2] e executado do lado cliente[3].

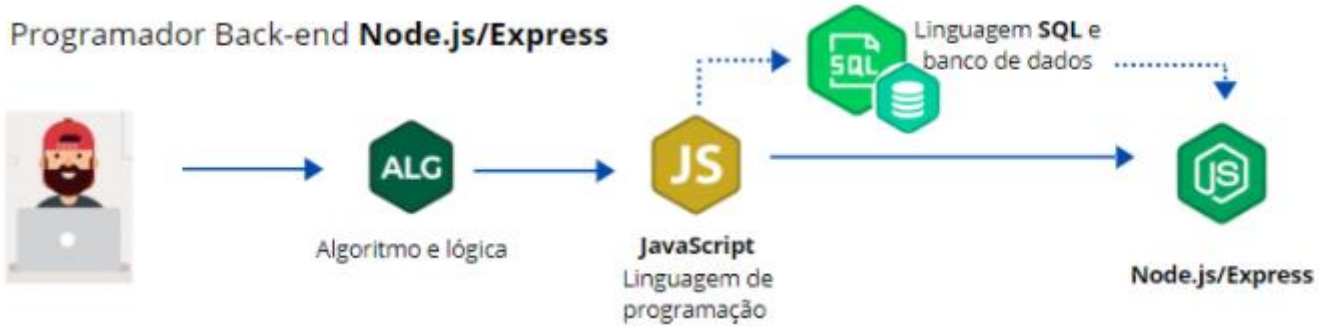
alert(“JavaScript – Como funciona?”);



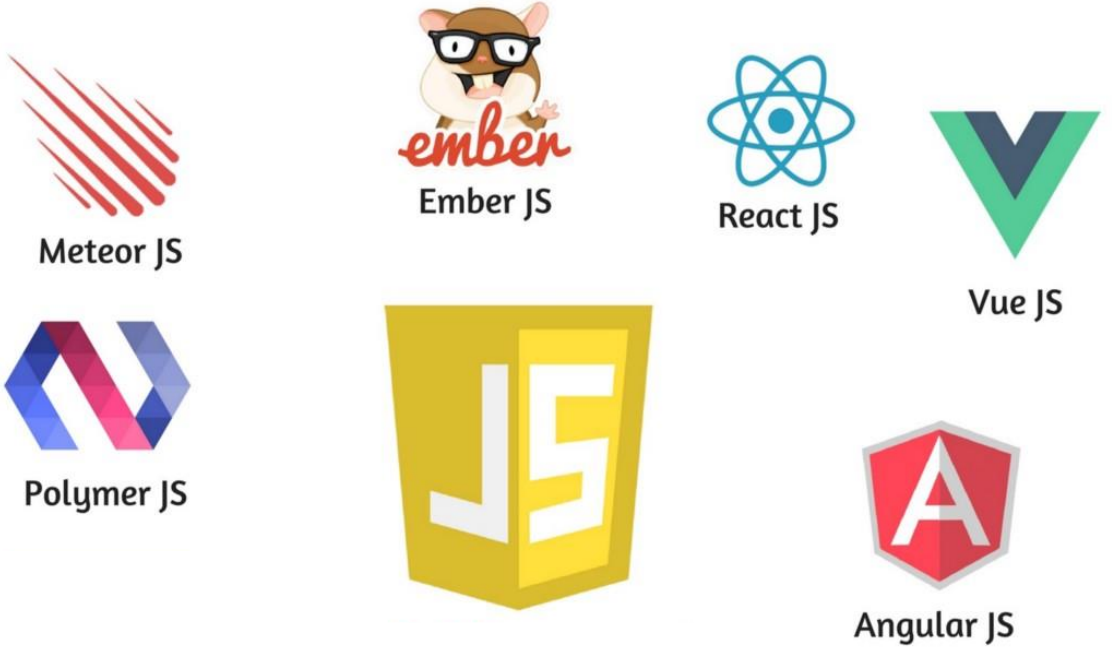
alert(“JavaScript é linguagem front - end ?”);



alert(“JavaScript – Como funciona?”);



alert(“JavaScript evoluções,bliotecas e frameworks?”);



alert(“JavaScript vem do Java?”);

Brendan Eich
em 1995

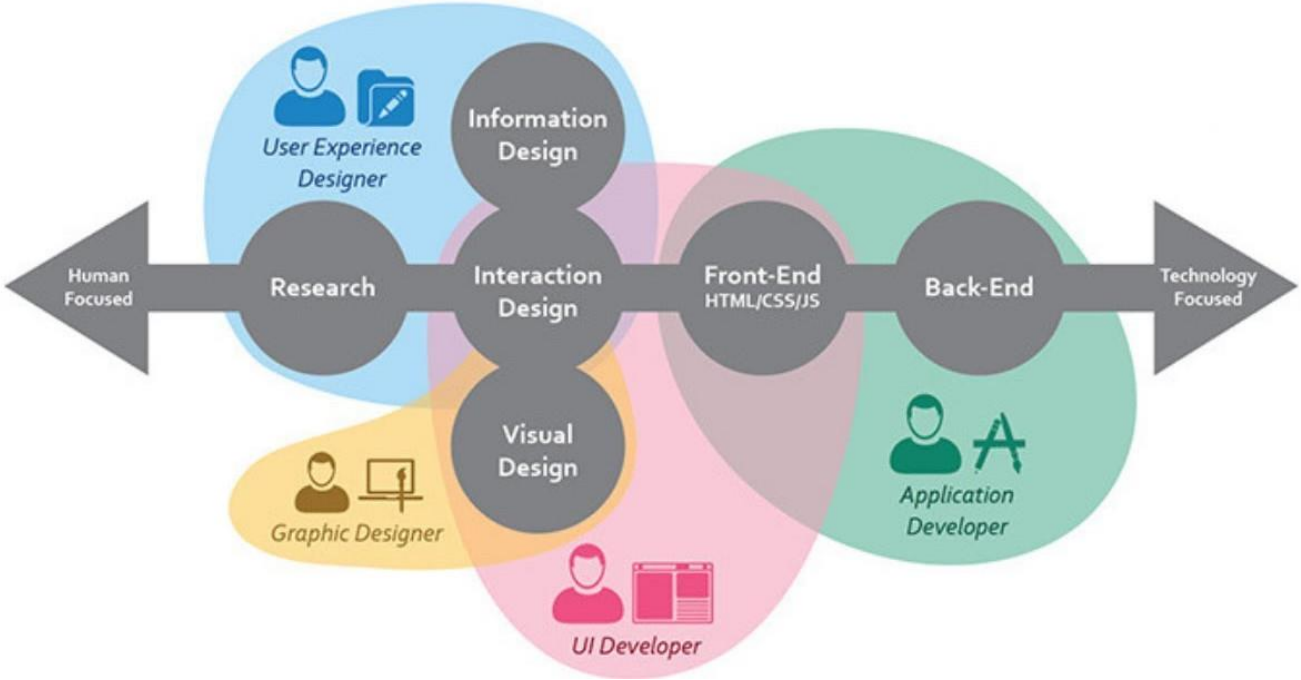
alert(“Por que Java?”);

Agora que sabemos que o nome da Linguagem JavaScript foi uma “carona” no sucesso da linguagem Java, precisamos saber o nome por trás da inspiração.

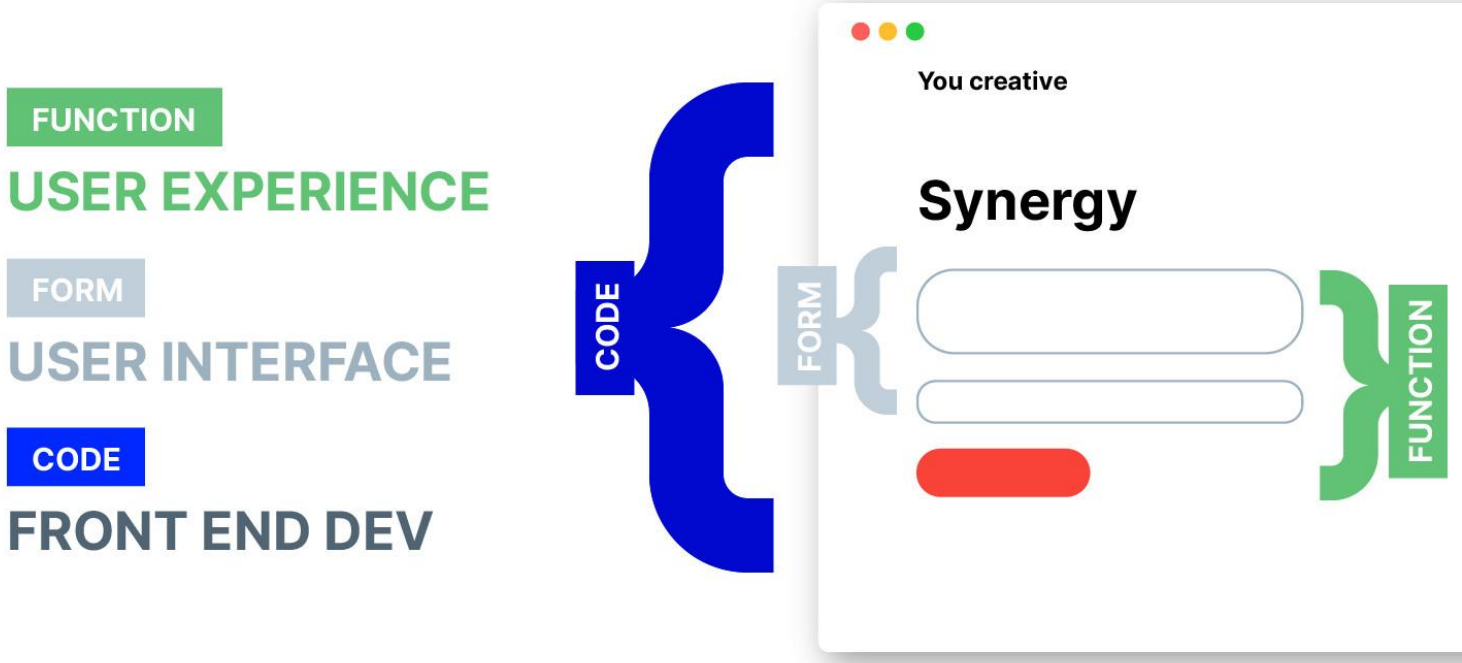
A logo da linguagem Java é uma xícara de café, certo? Isso é porque o criador da linguagem Java, fez inspirado no café que é combustível para todo programador, mas a palavra Java?

Nos Estados Unidos, um café forte ou um bom café é conhecido como Java, fazendo alusão a origem dos grãos, a Ilha de Java .

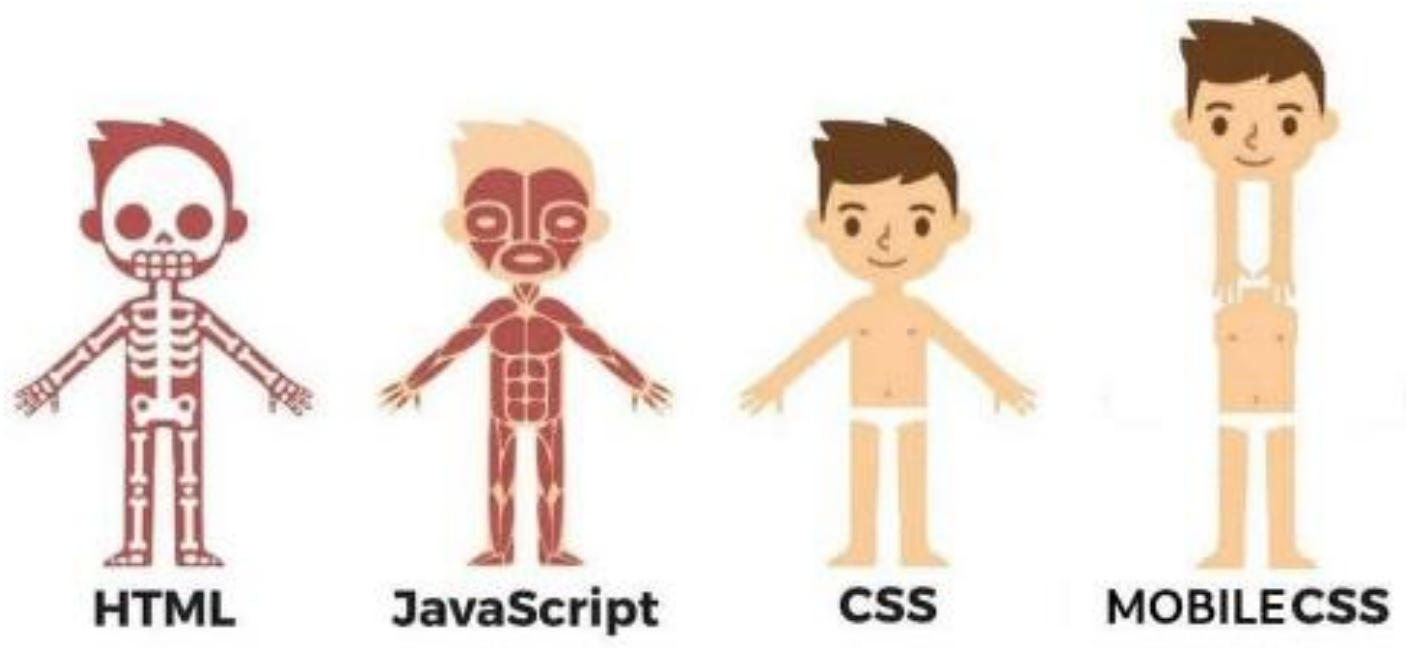
alert("JavaScript e o front - end que não é dev?");



alert(“JavaScript e o front - end que não é dev?”);

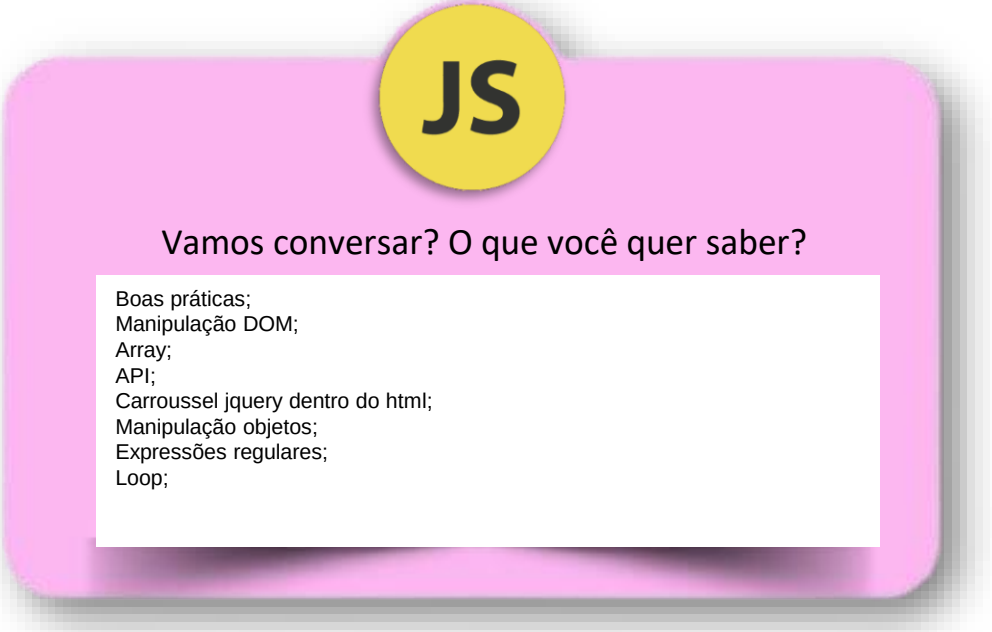


alert(“JavaScript e o front - end ?”);



alert(“JavaScript aonde?”);

alert(“JavaScript pra quê?”);



AULA 02

alert(“ Nosso projeto em JavaScript.”);

Requisitos:

“Vamos criar um projeto de uma página web que consuma um serviço de uma API.”

Essa página vai ter:
-Um carrossel;
-Formulário;
-Calculadora;
-Calendario;

Especificações Técnicas:

-Fazer no vscode;
-Boas práticas;
-Manipulação DOM;
Manipulação objetos;
Expressões regulares;
Array;
Loop;
-Carroussel jquery dentro do html;
-API;

Puxa-Frangos, personagem do desenho Pica-Pau.
Episódio 110: ***Não Puxem Minhas Penas , original Franken-Stymied***.
Temporada 4 - 1961.

alert(“JavaScript, boas práticas.”);

Comente seu Código: (// ou /* função */)

Pode até parecer desnecessário, no começo, mas acredite, você QUERERÁ comentar seus códigos da melhor forma possível. O que acontece quando você volta a trabalhar em um projeto, depois de um mês sem tocar nele, só para descobrir que você não é capaz de lembrar facilmentea linha de pensamento que tinha. E se um de seus colegas precisarem de revisar seu código? Sempre, sempre comente seções importantes do seu código

```
// Iterar sobre o vetor e imprime cada nome.
for(var i = 0, len = array.length; i < len; i++) {
 console.log(array[i]);
}
```

Sérgio Perez,Fórmula 1.
Segundo piloto da escuderia Red Bull - Quarto lugar geral em 2021.

alert(“JavaScript, boas práticas.”);

Variáveis e funções com nomeclatura que façam sentido para o négocio:

O Se você está contruindo uma funcionalidade para enviar e-mail, por favor, nomeie de forma adetaquada.

Isso:

```
function enviarEmail (){
    var validarEmail = window.document.getElementById("nome do id");
}
```

E não:

```
function x(){
    var y = window.document.getElementById("nome do id");
}
```

Valtteri Bottas,Fórmula 1.
Segundo piloto da escuderia Mercedes (2021) Alfa Romeo(2022)
Terceiro lugar geral em 2021.

alert(“JavaScript, boas práticas.”);

Remova o Atributo "Language":

Anos atrás, não era incomum encontrar o atributo "language" em tags script.

Porém, esse atributo, há tempos, é obsoleto, então, deixe-o de fora.

```
<script type="text/javascript" language="javascript">
...
</script>
```

Lewis Hamilton,Fórmula 1.
Primeiro piloto da escuderia Mercedes. Segundo lugar geral em 2021.

alert(“JavaScript, boas práticas.”);

Coloque os scripts na parte final da sua página:

O objetivo principal é fazer a página carregar o mais rápido possível para o usuário. Quando carregar um script, o navegador não pode continuar até que o arquivo inteiro tenha sido carregado. Assim, o usuário terá de esperar mais para perceber algum progresso.

Se o propósito dos seus arquivos JavaScript são, somente, para adicionar funcionalidades – por exemplo, após um botão ser clicado – vá em frente e coloque todos esses arquivos ao final da página, logo antes do fechamento da tag body. Isso é, definitivamente, uma boa prática.

```
<p>E, agora, você sabe os meus tipos favoritos de milho. </p>
<script type="text/javascript" src="path/to/file.js"></script>
<script type="text/javascript" src="path/to/anotherFile.js"></script>
</body>
</html>
```

Max Verstappen,Fórmula 1.
Primeiro piloto da escuderia Red Bull - Primeiro lugar geral em 2021.

alert(“JavaScript, boas práticas.”);

-Código indentado;
-Longas listas de parâmetros;
-Usar *“console.log();”* ao invéz de *“alert();” para validar* funcionalidade do código;
-Longas listas de parâmetros.

Saiba mais:
https://code.tutsplus.com/pt/tutorials/24-javascript-best-practices-for-beginners--net-5399

Pierre Gasly,Fórmula 1.
Primeiro piloto da escuderia AlphaTauri - 13º lugar geral em 2021.


DOM
“Que é,
onde vive,
de que se alimenta?"

alert(“DOM.”);

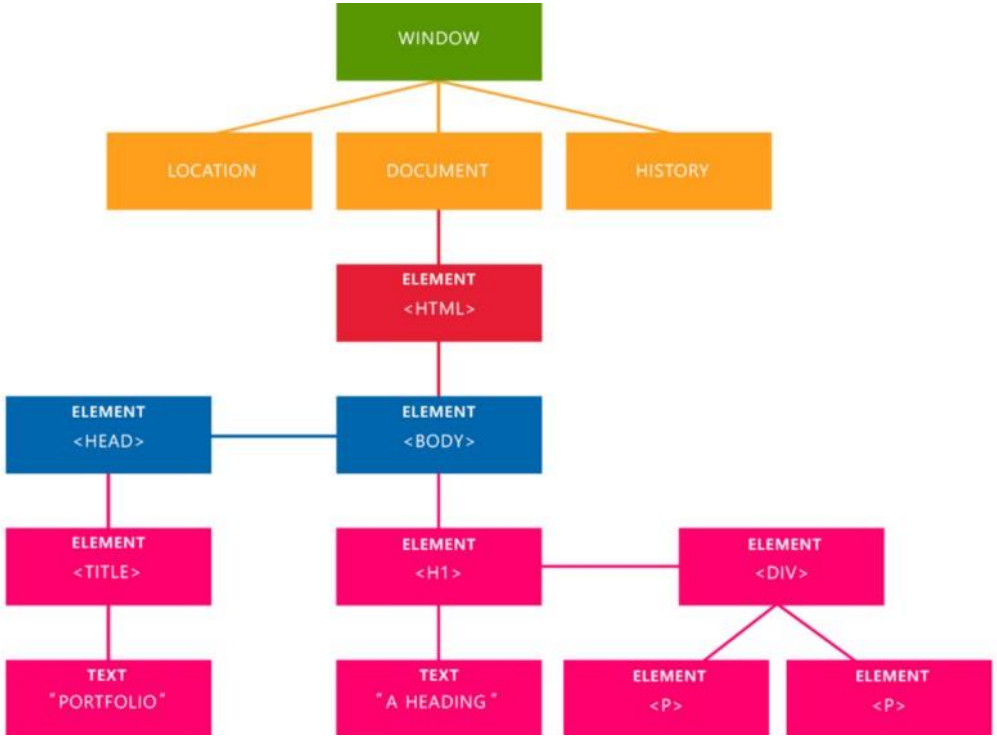
Modelo de Documento por Objetos (do inglês Document Object Model - DOM) é um dos conceitos mais importantes para quando precisamos interagir com a página web que desenvolvemos.

Para atualizar elementos, criar e até mudar texto do HTML pelo JavaScript nós utilizamos a DOM

O que é árvore DOM?

O Modelo de Objeto de Documento (DOM) é uma interface de programação para documentos HTML, XML e SVG . Ele fornece uma representação estruturada do documento como uma árvore. O DOM define métodos que permitem acesso à árvore, para que eles possam alterar a estrutura, estilo e conteúdo do documento..

alert(“Árvore DOM.”);

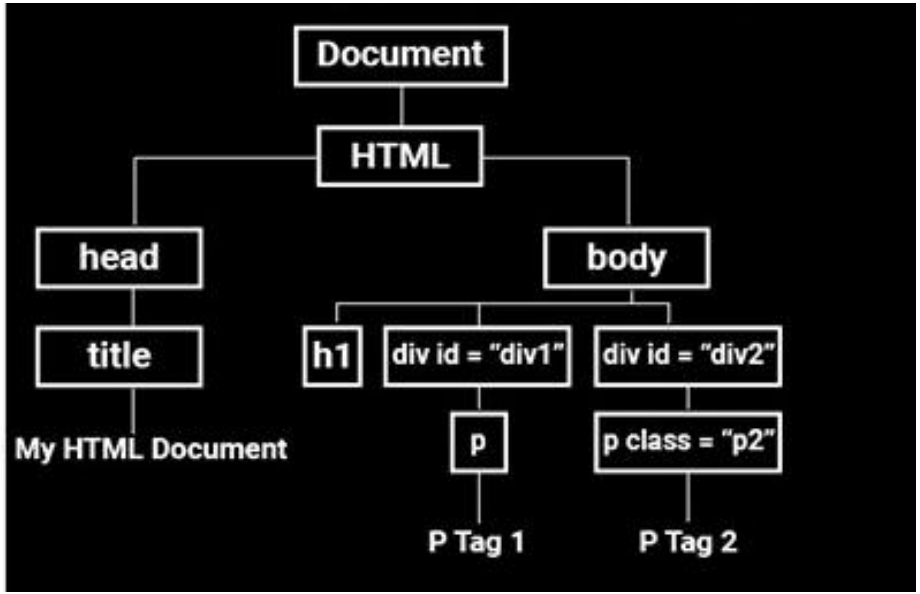


alert(“Árvore DOM na prática.”);

index.html

```
<html>
    <head>
        <title>My HTML Document</title>
    </head>

    <body>
        <h1>Heading</h1>
        <div id="div1">
            <p>P Tag 1</p>
        </div>
        <div id="div2">
            <p class="p2">P Tag 2</p>
        </div>
    </body>
</html>
```

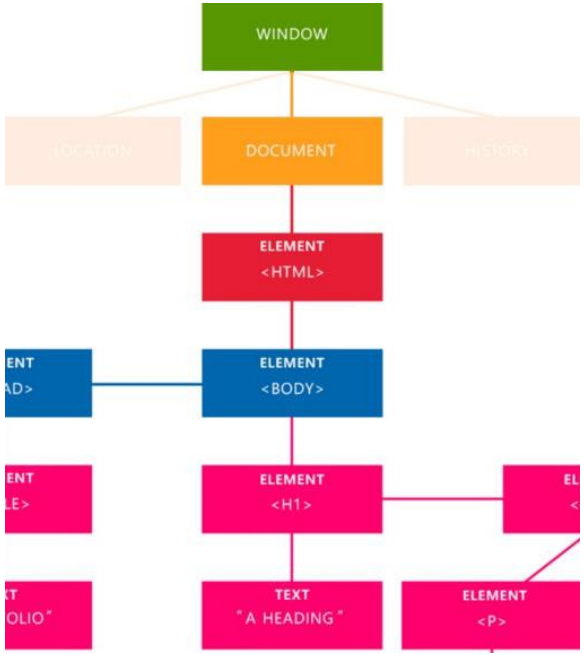


alert("DOM - Principais comandos : window.document.");

```
window.document.getElementById("nome do id");
       document.getElementsByClassName("nome da classe");
       document.getElementsByName("nome-dos-elementos");
       document.getElementsByTagName("p");
       document.querySelectorAll('seletor');
```

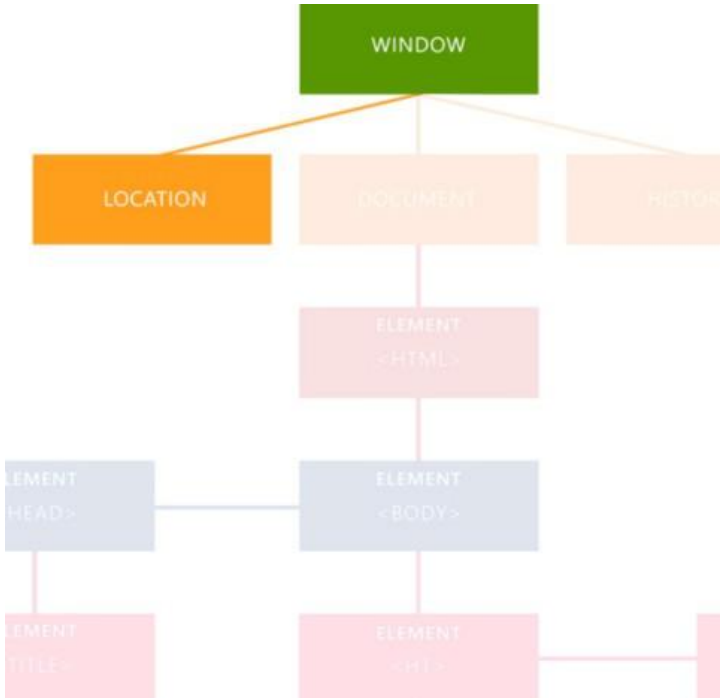
```
       document.getElementById("nome do id").innerHTML;
       document.getElementById("nome do id").nodeName;
```

Listagem completa: https://www.w3schools.com/jsref/dom_obj_all.asp



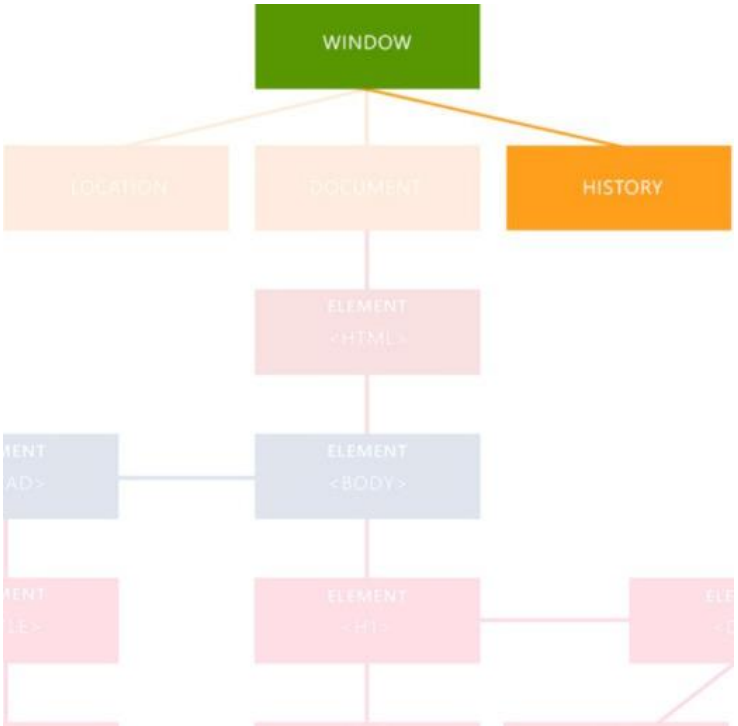
alert("DOM - Principais comandos : window.location.");

```
window.location.href
window.location.hostname
window.location.pathname
window.location.protocol
window.location.assign
```



alert("DOM - Principais comandos : window.history,");

```
window.history.back()
       history.forward()
```



alert(“DOM na prática.”);

```
<html>
    <button onclick="funcaoTesteDom()"> Clique aqui! </button>
</html>
<script>
funcaoTesteDom(){
            alert();
            document.getElementById("nome do id existente no html").innerHTML=" A Toti é Mara!";

            var p1 = window.document.getElementsByTagName('p')[0]
            window.document.write()

            var testeVar= document.getElementById("nome do id").nodeName;
            var testeVar= document.getElementById("nome do id").parentNode;
            var testeVar= document.getElementById("nome do id UL").parentNode;
            alert(testeVar);
            alert(testeVar.nodeName);
        }
</script>
```

alert(“Dúvidas: Array e forEach na prática.”);

```
let pessoas = [
  {
    nome: 'Joao',
    idade: 30
  },
  {
    nome: 'Maria',
    idade: 20
  }
];
pessoas.forEach((pessoa, index, array) => {
    console.log(`O nome é: ${pessoa.nome} e tem a idade ${pessoa.idade}`)
    //vai aparecer no console
    //O nome é: João e tem a idade 30
    //O nome é: Maria e tem a idade 20
});
```

alert(“ Dúvidas: Sobre a Calculadora >> Eval(). ”);

A função Eval avalia o expressão de cadeia de caracteres e retorna seu valor. Por exemplo, Eval("1 + 1") retorna 2.

Se você passar para a função Eval uma cadeia de caracteres que contém o nome de uma função, a função Eval retornará o valor de retorno da função.O mau uso do eval pode levar a ataques de injeção de código.

É mais dificil de fazer debug (uma vez que dentro do eval os error não têm indicação de linha), mais lento que código similar sem eval.Dificulta e limita a minificação do código.

```
function calc(e){

        var operacao = e.value;

        var n1=parseFloat(document.getElementById("n1").value);
        var n2=parseFloat(document.getElementById("n2").value);

        var calculo = eval(n1+operacao+n2);

        if(!isNaN(calculo)){
            alert(calculo);
        }
```

AULA 03

alert(“ Nosso projeto em JavaScript.”);

Especificações Técnicas => Check-list de temas já validados:

~~-Fazer no vscode;~~

~~-Boas práticas;~~

~~-Manipulação DOM;~~

~~*Manipulação objetos;*~~

~~*Expressões regulares;*~~

~~*Array;*~~

~~*Loop;*~~

-Carroussel jquery dentro do html;

-API;

Puxa-Frangos, personagem do desenho Pica-Pau.
Episódio 110: ***Não Puxem Minhas Penas , original Franken-Stymied***.
Temporada 4 - 1961.


API
“Que é,
onde vive,
de que se alimenta?"

alert(“ API.”);

Antes de aprender a consumir dados de uma API com JavaScript, precisamos saber o que é uma API.

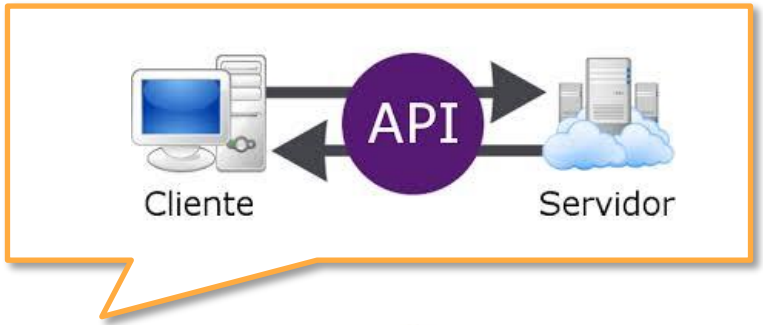
O conceito de API nada mais é do que uma forma de comunicação entre sistemas. Elas permitem a integração entre dois sistemas, em que um deles fornece informações e serviços que podem ser utilizados pelo outro, sem a necessidade de o sistema que consome a API conhecer detalhes de implementação do software.

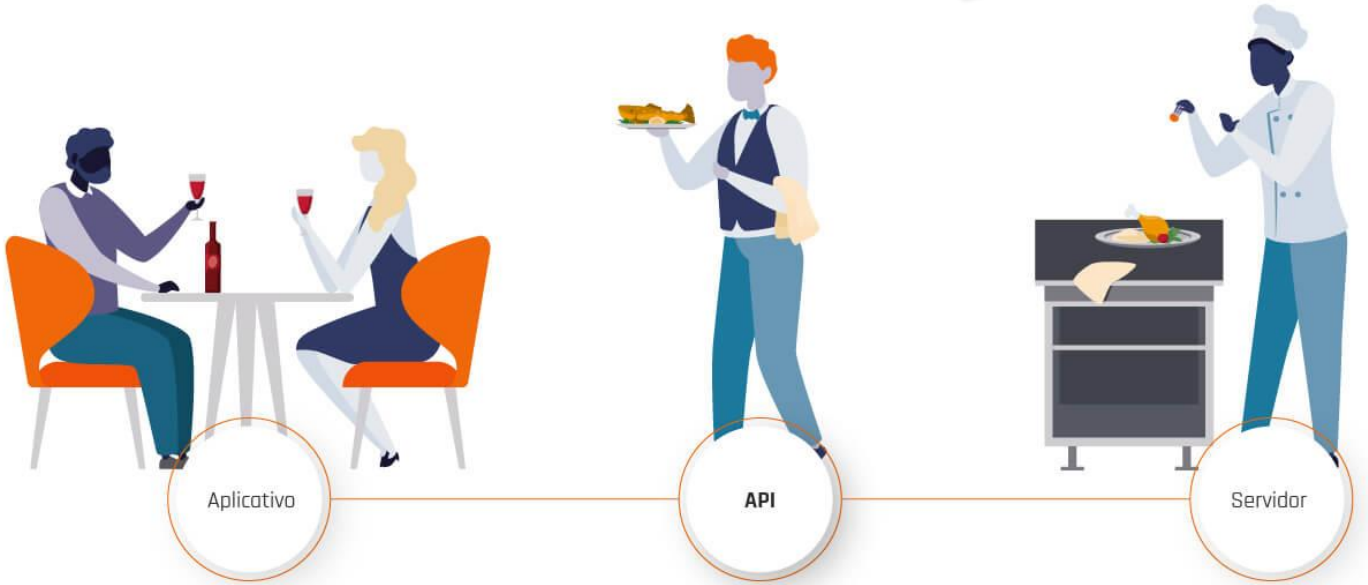
alert(“ API.”);

Metaforicamente, podemos pensar no que é API como um garçom. Quando estamos em um restaurante, buscamos o que desejamos no menu e solicitamos ao garçom. O garçom encaminha esse pedido à cozinha, que prepara o pedido. No fim, o garçom traz o prato pronto até a gente. Não temos detalhes de como esse prato foi preparado, apenas recebemos o que solicitamos.

Com os exemplos de API disponíveis, a ideia é a mesma do garçom. Ela vai receber seu pedido, levar até o sistema responsável pelo tratamento e te devolver o que solicitou (o que pode ser uma informação, ou o resultado do processamento de alguma tarefa, por exemplo).

alert(“ API.”);





alert(“ API – Curiosidades.”);

- As API não necessariamentes precisam trabalhar via WEB;
- Para WEB, usamos a API Rest utiliza o protocolo HTTP;
- Existem quatro tipos de APIs:
    - APIs públicas ou abertas;
    - APIs privadas ou internas;
    - APIs de parceiros;
    - APIs compostas;

-API REST funciona como uma interface que permite editar recursos no lado do servidor, funcionando como um CRUD (Create, Read, Update e Delete) através do HTTP.

alert(“ API – Verbos.”);

Verbo GET:
Sem passagem de ID: vai retornar todas as notas (ou as notas mais recentes, isso cabe a regra de negócio da aplicação).

Verbo POST:
Normalmente usado sem passagem de parâmetro – usado para criar uma nova nota.

Verbo DELETE:
Usado para remover o recurso (por exemplo uma nota): utilize com passagem de ID.

Verbo PUT:
Normalmente usado com parâmetro; Use para editar o recurso.

Verbo PATCH:
Usado para editar o recurso sem a necessidade de enviar todos os atributos – o consumidor envia apenas aquilo que de fato foi alterado (mais o ID como parâmetro, para que o serviço saiba o que vai ser alterado).

alert(“ API – REST X RESTful ”);

Por definição, uma API Restful é uma API REST que cumpre com todos os 6 requisitos propostos pelo Roy Fielding, que são:

Interface Uniforme:
Um recurso não deve ser muito grande e conter muita informação em seu conteúdo, ao invés disso, deve conter links para outros recursos quando necessário.

Cliente Servidor:
Backend e Frontend devem poder ser desenvolvidos separadamente e até mesmo trocados, contando que a interface entre eles não seja alterada.

Stateless:
Nenhum contexto do cliente deve ser armazenado no lado do servidor entre requisições. O cliente é responsável por armazenar o estado da aplicação.

alert(“ API – REST X RESTful ”);

Cacheable:
No REST, devemos aplicar cache quando possível, quando um recurso não muda muito com o tempo, por exemplo. Os recursos devem se declarar cacheaveis, através dos headers da requisição HTTP.
Uma estratégia de cache bem gerenciada reduz a carga de trabalho do backend, melhorando a escalabilidade e performance da aplicação.

Arquitetura em Camadas:
O estilo de sistema em camadas permite que uma arquitetura seja composta de camadas hierárquicas, restringindo o comportamento do componente de forma que cada componente não possa "ver" além da camada imediata com a qual está interagindo.

Código sob demanda (opcional):
Apesar de geralmente lidarmos com JSON nas respostas de nossa API, também é permitido responder código executável para suportar parte da aplicação.

alert(“ API – links para estudo:”);

Postman:
https://www.postman.com/
Postman é uma plataforma de API para desenvolvedores projetar, construir, testar e iterar suas APIs.

Swagger:
https://swagger.io/
O Swagger é, basicamente, um conjunto de ferramentas que nos ajuda a fazer o design, ou seja, fazer a modelagem, a documentar e até gerar código para desenvolvimento de APIs.

alert(“APIs disponivéis.”);

Dito tudo isso... Como nosso objetivo é consumir dados de uma API, vamos utilizar uma API disponível para desenvolvedores:

https://pokeapi.co/

https://swapi.dev/

imgur

https://api.imgur.com/

alert(“ Bora codar?!”);

Para consumir uma API com JavaScript precisamos conhecer mais sobre fetch() :

Fetch API é uma versão mais simples e mais fácil de usar da XMLHttpRequest para consumir recursos de modo assíncrono. Fetch permite que você trabalhe com as APIs REST com opções adicionais como cache de dados, leitura de respostas em streaming e mais. A maior diferença está no fato de que Fetch trabalha com promises em vez de callbacks.

fetch() retorna uma classe Promise para que você possa usar os métodos then() e catch().

XMLHttpReques:
É uma API disponível em linguagens de script para navegadores web tais como JavaScript. É utilizada para enviar requisições HTTP ou HTTPS diretamente para um servidor web e carregar os dados de resposta do servidor diretamente de volta ao script

Promise:
Permite a construção de funções de processamento assíncrono representando um valor que poderá estar disponível no futuro.

then:
Permite definir o bloco executado mediante o cumprimento de uma promise retornando um objeto do tipo Response.

catch:
Permite definir o bloco executado mediante a rejeição de uma promise.

alert(“Resultado”);

```
function loadpk(){

    var valorPk= document.getElementById("numeroPk").value;

    var url="https://pokeapi.co/api/v2/pokemon-form/"+valorPk+"/";

    fetch (url)
        .then((response)=> {
            return response.json();
        })

        .then ((data) => {

            console.clear();
            console.log(data);
            document.getElementById('nome').innerHTML = data['name'];
            document.getElementById('habilidades').innerHTML = data['types']['0']['type']['name'];

            let img = data['sprites']['front_default'];
            document.getElementById('picFront').setAttribute('src',img);


            let imgBack = data['sprites']['back_default'];
            document.getElementById('picBack').setAttribute ('src',imgBack);
        })

        .catch((erro) => {
            console.log("Erro:"+erro);
        });

}
```

AULA 04

alert(“Reaproveitamento do código”);

Proposta:
Reaproveitar/Adaptar o código para consumir a API da aula anterior para um novo desenvolvimento.

Exercício:
Encrementar o portal aproveitantando todos os conceitos de Js, Html e Css.

alert(“Exercício: Alteração do escopo”);


MATONDO!
(OBRIGADO)
_toti
@toti.diversidade
/school/toti-diversidade
/toti.diversidade
_https://totidiversidade.com.br