SENAC

Campus Santo Amaro

.

TADS - Análise Desenvolvimento de Sistemas

PW - Programação Web

Aula #2
JavaScript:Introdução

Professor: Veríssimo - carlos.hvpereira@sp.senac.br

16/08/2022

Sobre este documento

Este documento objetiva deixar registrado o conteúdo abordado em sala de aula pelo professor. Importante destacar que a Nota de Aula serve como guia ao professor, bem como serve aos alunos como um norte, quanto ao conteúdo desenvolvido em sala de aula.
Este documento não tem a pretenção de ser uma única fonte para estudo. Para tal, o aluno deverá assistir às aulas e fazer uso (consulta) à bibliograa recomendada na ementa da disciplina, e à bibliograa complementar, apontada pelo professor.

Preâmbulo da Aula

Esta aula aborda coneitos fundamentais da linguagem JavaScript. Neste sentendido, trataremos dos seguintes tópicos:

- Entendendo JavaScript
- Manipulção de Variáveis (var, let, const)
- Comando de Entrada
- introdução a funçoes

Importante destacar que esta aula possui uma abordagem prática, na qual os elementos conceituai servem de guia para a parte prática da aula.

Contents

Chapter 1

JavaScript- Introdução

1.0.1 O que é JavaScript

O JavaScript **(JS)** é uma linguagem de programação que permite que implementemos itens complexos em páginas Web. **(JS)** é geralmente utilizada em páginas Web, onde possui um papel importante para que uma página HTML possua elementos dinâmicos, pois, o HTML/CSS por si só, não possuem esta caracterísitica (Lembramdo que HTML é uma linguagem de marcação, e não uma linguagem procedural).

O poder da linguagem JavaScript transcende ao ambiente Web pois, com esta poderosa linguagem podemos desenvolver **App** Android/IoS, bem como também desenvolver servidores, **API-Rest** (Back-end).

1.0.2 Algumas Caracterísiticas Importantes do JS

Atenção a pontos importantes a considerar sobre a linguagem (**JS**):

- É uma linguagem Interpretada (Não é compilada)
- É **Case Sentisitve**
- É uma linguagem **não-tipada**
- Linguagem baseada em protótipos
- Suportando estilos de orientação a objetos, imperativos e declarativos (programação funcional)

Chapter 2

Utilizando JavaScript

O **JS** é uma linguagem de programação que permite implementar funcionalidades mais complexas em páginas web **(Front-end)**, bem com em ambientes de servidores **(Back-end)**

2.0.1 Ambiente Web - Font-End

A figura 2.1 ilustra a utilização da linguagem **JS** em uma página **HTML**. Observe que a página HTML aciona a função **clicou()**, a partir do envento **onclick** - **"Button"**

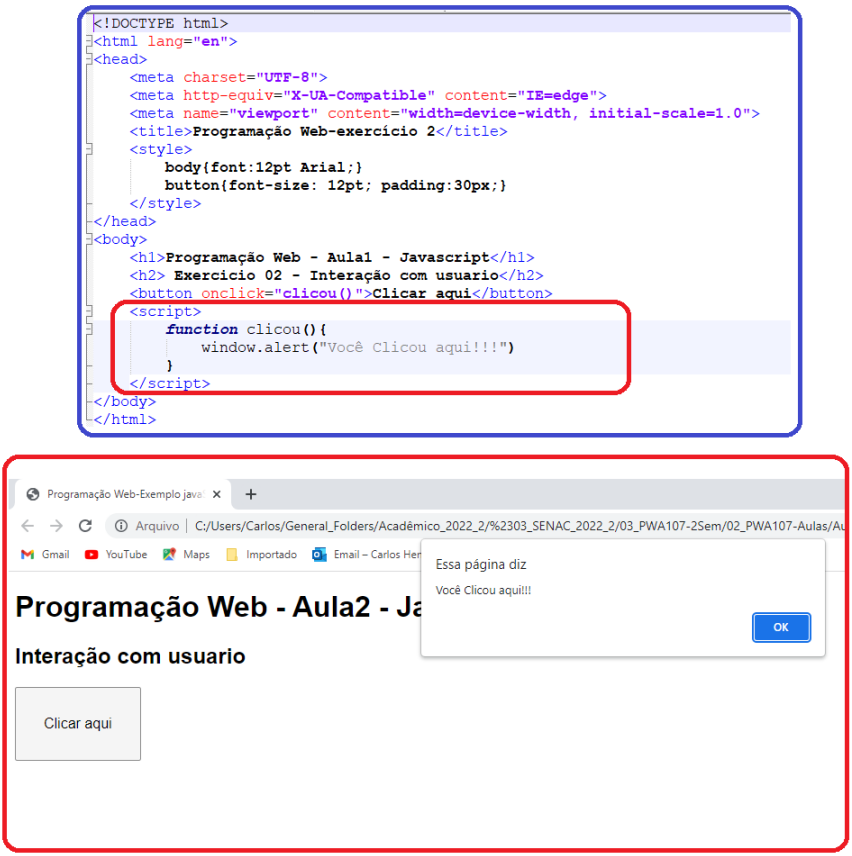

Figure 2.1: Exemplo JS em Página HTML

2.0.2 Ambiente Servidor - Back-End

A figura 2.2 ilustra a utilização da linguagem **JS** em uma Aplicação no Servidor **API**. Neste exemplo a API responde a uma requisição: Devolve em formato de protocolo **JSON**

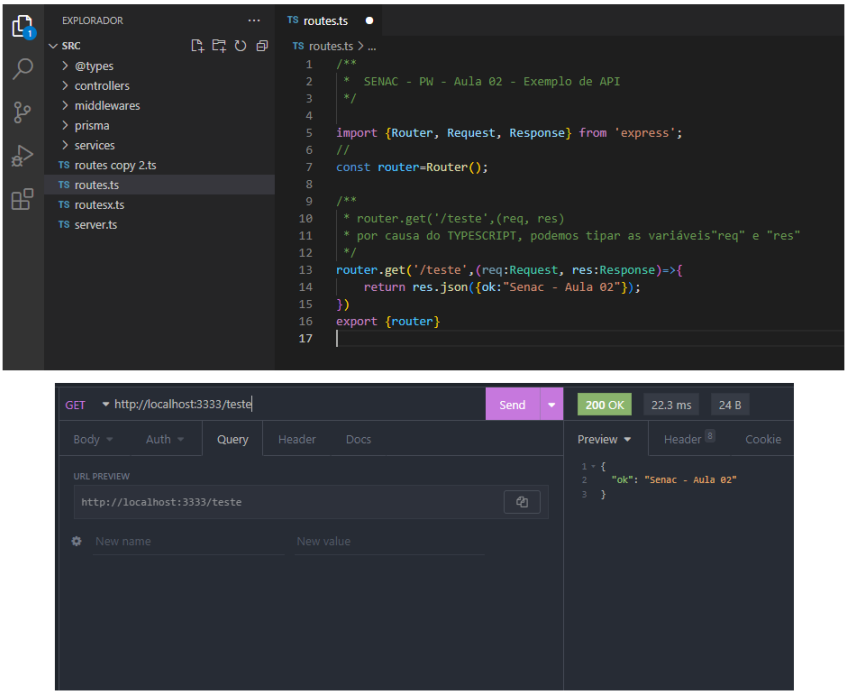

Figure 2.2: Exemplo JS no Servidor - API

Chapter 3

JavaScript - Manipulando Variáveis

3.0.1 Tipos de Dados

- Boolean: entidade lógica e pode ter dois valores: verdadeiro(true) ou falso(false).
- Null:tem exatamente um valor: null
- Undefined:Uma variável que não foi atribuída a um valor específico
- Number:O tipo number possui apenas um inteiro que tem duas representações: 0 é representado como -0 ou +0.
- BigInt:é um tipo de dado numérico que representa inteiros no formato de precisão arbritrária.
- String:O tipo String em JavaScript é usado para representar dados textuais. Isto é um conjunto de "elementos" de valores de 16-bits

- Symbol: Um Symbol é um valor primitivo único e imutável e pode ser usado como chave de uma propriedade[1] de Object
- **Object**: refere-se a uma estrutura de dados contendo dados e instruções para se trabalhar com estes dados.

O operador **typeof** retorna uma string indicando o tipo de um operando.
O operador typeof pode ser utilizado das seguintes maneiras:

- **typeof** operando

Exemplo:

- console.log(typeof "3.14");

Resultado: **string**

[1]Uma propriedade Javascript é uma característica de um objeto, frequentemente descrita como atributos associados à uma estrutura de dados.

3.0.2 Regras para os Identificadores

Os nomes de variáveis [2], chamamos de **"Identificadores"**. Temos que atentar para as seguintes regras:

- Podemos começar com letra, $ ou _ (Underline)
- Não pode começar com **números**
- podem conter letras e números
- é possível usar acentos e símbolos
- Não podem conter espaços
- não podem ser palavras reservadas

3.0.3 Escopo de Variável

Ao utilizamos JS é imprescindível que entendamos como aplicar escopo de variável:[3] de forma correta.

- Escopo global
  - É definida quando declaramos uma variável fora de qualquer função - ela torna acessível a qualquer parte da nossa aplicação ou site, podendo ser lida e alterada
- Escopo Local
  - É declarada dentro de uma função

Escopos criados por funções são chamados de **function scopes**, enquanto escopos criados por estruturas de controle são chamados de **block scopes**.

[2]Na W3schools usamos camelCase para nomes de identificadores (variáveis e funções).

[3]Escopo de variável é o local de nosso código onde uma determinada variável pode ser acessada:global ou local

3.0.4 Identificador var

O identificador **var** age sobre o **escopo da função**:Ao declaramos uma variável sem o uso da palavra reservada **var** estaremos criamos uma variável **global implicitamente**, e automaticamente ela se torna global independente de onde ela for definida

3.0.5 Identificadores let e const

A grande mudança trazida pelo **ES2015** foram a introdução de **let** e **const** como maneiras de definirmos variáveis.
Essas **keywords** permitem que trabalhemos não só com o escopo de funções, mas também com o escopo dos blocos

- let tem escopo de bloco
- let pode ser atualizado, mas não declarado novamente.
- Declarações com const têm escopo de bloco
- Variáveis declaradas com const mantêm valores constantes
- const não pode ser atualizado nem declarado novamente
- Cada declaração com const deve ser inicializada no momento da declaração.
- Declarações de const somente podem ser acessadas dentro do bloco onde foram declaradas.

Chapter 4

JavaScript - Introdução a Funções

Função é um bloco de construção, é um procedimento, ou seja, é um conjunto de instruções que executa uma tarefa. Para usar uma função, você deve defini-la em algum lugar no escopo do qual você quiser chamá-la.

4.0.1 Declarando Função

Para definir uma função (declaração de função) devemos utilizar a **palavra chave** function, seguida por:

- Nome da Função.
- Lista de argumentos para a função, entre parênteses e separados por vírgulas.
- Declarações JavaScript que definem a função, entre chaves { }.

Exemplo de código (Função)

```
function fatorial(n){
  if ((n == 0) || (n == 1))
    return 1;
  else
    return (n * fatorial(n - 1));
}
```

Figure 4.1: Exemplo de uma função JS

4.0.2 Chamando Funções

A definição de uma função não a executa. Chamar a função executa realmente as ações especificadas com os parâmetros indicados. A figura 4.2 mostra a chamada (5 vezes) da função fatorial()

```
var a, b, c, d, e;
a = fatorial(1); // a recebe o valor 1
b = fatorial(2); // b recebe o valor 2
c = fatorial(3); // c recebe o valor 6
d = fatorial(4); // d recebe o valor 24
e = fatorial(5); // e recebe o valor 120
```

Figure 4.2: Exemplo chamada de função em JS