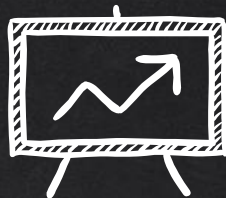


Instituto Federal de São Paulo

Técnico de Informática Integrado ao Ensino Médio

EDUCAÇÃO
PÚBLICA
100%
GRATUITA



INTRODUÇÃO À & JAVASCRIPT

PROFESSOR JOHNATA SANTICIOLI





SUMÁRIOS

- X Introdução
- X Sintaxe
- X Operadores
- X Variáveis
- X Tipos de Dados
- X Funções





INTRODUÇÃO



Estrutura



Estilo



Comportamento





INTRODUÇÃO

- X JavaScript é uma linguagem de programação *interpretada*. Foi originalmente implementada como parte dos navegadores web para que scripts pudessem ser executados do *lado do cliente* e interagissem com o usuário sem a necessidade deste script passar pelo servidor, controlando o navegador, realizando comunicação assíncrona e alterando o conteúdo do documento exibido.

Flanagan, David; Ferguson, Paula (2002). JavaScript: The Definitive Guide 4th ed. O'Reilly & Associates [S.l.]





INTRODUÇÃO

- X JavaScript:
- X Interpretada pelo navegador
- X Modelo de execução controlado por eventos
- X Multi-paradigma
- X Baseado em objetos
- X Tipagem dinâmica
- X Case-sensitive





INTRODUÇÃO

- X O que pode-se fazer com JavaScript:
- X Animações
- X Verificar formulários
- X Manipular elementos do documento HTML
- X Manipular arquivos XML e JSON
- X Server-side com node.js e deno.js





INTRODUÇÃO

X Pode-se integrar o JavaScript no HTML das seguintes formas:

@ Integração interna

- No corpo da página (`<body>...</body>`)
- No cabeçalho (`<head>...</head>`)
- Em uma tag HTML

@ Integração externa

- Em um arquivo(.js) separado



INTRODUÇÃO

X Pode-se integrar o JavaScript no HTML das seguintes formas:

```
<!DOCTYPE html>
<html lang="pt-br">
<head>
  <script> alert("1-Olá Mundo!") </script> //no cabeçalho
  <script type="text/javascript" src="script.js"></script> //arquivo externa
</head>
<body>
  <input type="button" onclick="alert('3-Olá Mundo!')"> //na tag html
  <script> alert("2-Olá Mundo!") </script> //no corpo
  <script type="text/javascript" src="jquery.js"></script> //arquivo externa
</body>
</html>
```



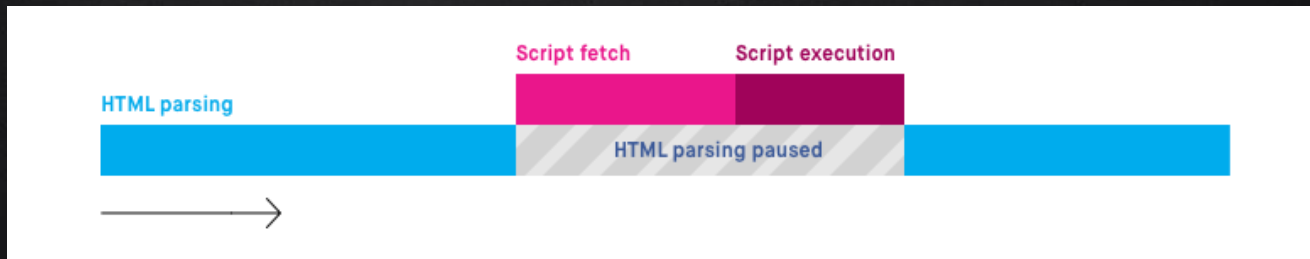


INTRODUÇÃO

X Execução normal

X Por que normalmente coloca-se os scripts no fim da página?

...
<script type="text/javascript" src="script.js"></script>
</body>

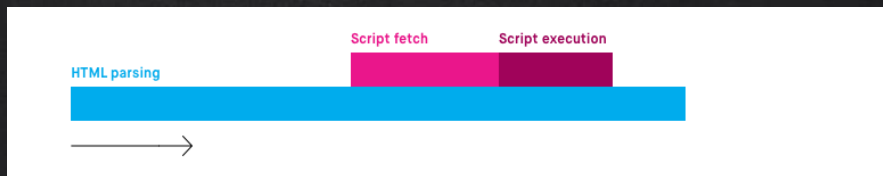




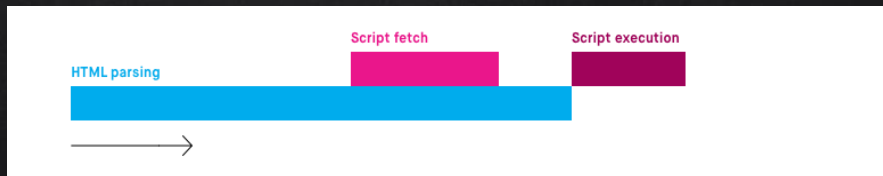
INTRODUÇÃO

X Atributos **async** e **defer**

`<script async src="script.js"></script>` *//só para arquivo externa*



`<script defer src="script.js"></script>`



Saiba mais em: <https://www.braziljs.org/p/diferencas-entre-async-e-defer>



COMENTÁRIOS

```
<script>  
    //Comentário de linha  
  
    /*  
        Comentário  
        de bloco  
    */  
</script>
```





TIPO DE DADOS

@ Tipos primitivos

- Boolean (*Lógico*)
- Null
- Undefined
- Number (*numérico*)
- BigInt (*número extensos e strings*)
- String (*caracteres*)
- Symbol (*símbolo único e imutável para chave de propriedade*)
- Function

@ Tipos Objetos

- Date, RegExp, Error, ... , objeto global e prototype
- Arrays





TIPO DE DADOS

Tipo	valor	Descrição	Exemplo
Boolean	true	Valor lógico verdadeiro	<code>var b = true;</code>
	false	Valor lógico falso	<code>var b = false;</code>
Number	NaN	Not a Number é o resultado de uma expressão com um operando que não pode ser convertido em valor numérico	<code>var a = NaN;</code> <code>var a = 10 / "x";</code> <code>// a assume valor NaN</code>
	Infinity	Representação de um valor infinito	<code>var a = Infinity;</code> <code>var a = 10 / 0;</code> <code>// a assume valor Infinity</code>
	undefined	conteúdo de variáveis não iniciadas	<code>var a;</code> <code>// a assume valor undefined</code>
	null	representa o não valor , ou seja a inexistência de valor associado a uma variável	<code>var a = null;</code> <code>// x tem um não valor, ou seja null</code>



VARIÁVEIS

- X JavaScript é uma linguagem de tipagem dinâmica e fraca:
- Não é necessário **declarar o tipo** de uma variável;
 - Todas as variáveis são **objetos** (referência);
 - A variável irá “alterar” o seu tipo de dado conforme os valores forem atribuídos:
 - Tipo de dado dinâmico:
 - `var x;` // x é indefinido
 - `x = 5;` // x é um número
 - `x = "Johnata";` // x é uma string
 - `x = true;` // x é um valor lógico
 - `x = null;` // x é nulo

```
//template string  
console.log(`Nome: ${x}`)
```





PALAVRAS RESERVADAS

abstract
boolean **break** byte
case catch char class const **continue**
debugger **default delete do** double
else enum export extends
false final **finally** float **for function**
goto
if implements import **in instanceof** int interface
long
native **new null**
package private protected public
return
short static super **switch** synchronized
this throw throws transient **true try typeof**
var volatile **void**
while with





TIPO DE DADOS

X Função **typeof**(operador) – retorna o tipo da variável ou constante

> a=9.5

> typeof(a)
'number'

Object	'object'	Atenção !
Array	'object'	
Function	'function'	Atenção !
String	'string'	
Number	'number'	
Boolean	'boolean'	
null	'object'	
undefined	'undefined'	



ESCOPO DE VARIÁVEIS E CONSTANTES

X Declarando variáveis e constantes:

- `var x = 5` – usada para declarar tanto variáveis locais em funções como variáveis globais
- `let x = 5` – usada para declarar uma variável local de escopo de bloco
- `const x = 5` – usada para declarar uma constante local de escopo de bloco





ESCOPO DE VARIÁVEIS E CONSTANTES

keyword	const	let	var
global scope	NO	NO	YES
function scope	YES	YES	YES
block scope	YES	YES	NO
can be reassigned	NO	YES	YES





OPERADORES

X Operadores no JavaScript:

- **aritméticos**: +, -, *, **, /, %
- **atribuição**: =, +=, -=, *=, **=, /=, %=, ++, --
- **relacionais**: ==, ===, !=, !==, <, <=, >, >=
- **lógicos**: &&, ||, !
- **ternário**: **condição** ? **caso verdadeiro** : **caso falso**

```
var salario = 1000
```

```
var bonus = salario * (salario > 1000 ? 0.10 : 0.15)
```





OPERADORES LÓGICOS

O operador `===` e `!==` avalia os operandos e então os compara, sem realizar conversão de tipo.

```
> 1 === 1
true
> 1 === '1'
false
> 1 !== 1
false
> 1 !== '1'
true
>
```

Os operadores `==` e `!=` são menos estritos.

Se os valores dos operandos não forem do mesmo tipo, ele tenta algumas conversões de tipo e realiza a comparação novamente.

```
> 1 == 1
true
> 1 == '1'
true
> 1 != '1'
false
> 1 != 1
false
>
```



OPERADORES E PRIORIDADES



Operador	Descrição
<code>· [] ()</code>	Acesso a propriedades, indexação, chamadas a funções e sub-expressões
<code>++ -- ~ ! new delete typeof</code>	Operadores unários e criação de objectos
<code>* / %</code>	Multiplicação, divisão, divisão módulo
<code>+ -</code>	Adição, subtração, concatenação de <i>strings</i>
<code><< >> >>></code>	Deslocação de Bit
<code>< <= > >= instanceof</code>	Menor, menor ou igual, maior, maior ou igual, <i>instanceof</i>
<code>== != === !==</code>	Igualdade, desigualdade, igualdade estrita, e desigualdade estrita
<code>&</code>	AND bit a bit
<code>^</code>	XOR bit a bit
<code> </code>	OR bit a bit
<code>&&</code>	AND lógico
<code> </code>	OR lógico
<code>? :</code>	Operador condicional (ternário)



REFERÊNCIAS

Você poderá estudar mais sobre html5 nas seguintes páginas:

<https://developer.mozilla.org/pt-BR/docs/Web/JavaScript>

https://developer.mozilla.org/pt-BR/docs/Web/JavaScript/Guide/Grammar_and_types

<https://www.ecma-international.org/publications-and-standards/standards/ecma-262/>

<https://www.braziljs.org/p/diferencas-entre-async-e-defer>

<https://www.alura.com.br/artigos/entenda-diferenca-entre-var-let-e-const-no-javascript>





THANKS!

Até a próxima aula!!!

www.ifsp.edu.br
johnata.santicioli@ifsp.edu.br

