Introdução ao JavaScript

Rafael Hoffmann
raelhoff@{gmail.com/edu.univali.br}

UNIVALI
Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

JavaScript

- Também chamada de JS, é a linguagem de criação de scripts para a web.
- É utilizado por bilhões de páginas para:
  - Adicionar funcionalidades;
  - Verificar formulários;
  - Comunicar com servidores;
  - e muito mais.

JavaScript

- Originalmente criada pela Netscape por Brendan Eich em 1995
- Seu primeiro nome Mocha, depois LiveScript na versão beta do Netscape Navigator 2.0
- Browser's World War
- O nome da linguagem mudou para JavaScript, a partir de um acordo feito com a Sun com o objetivo de destruir a Microsoft, sendo assim, foi registrado com uma marca pertencente a Sun e era de uso exclusivo da Netscape

JavaScript

- Na mesma época, a Microsoft desenvolveu uma linguagem JScript, semelhante a linguagem Javascript.
- Em 1997, a Netscape conseguiu padronizar a linguagem junto a ECMA (*European Computer Manufacturers Association*) international, renomeando-a ECMAScript.

Tipos de Dados

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Variáveis

JavaScript é uma linguagem de tipagem dinâmica e fraca:

- Não é necessário declarar o tipo de uma variável;
- Todas as variáveis são objetos (referência);
- Números são todos reais de 64 bits;
- 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 = "John" ;  // x é uma string
x = true;       // x é um valor lógico
x = null;       // x é indefinido
```

String

```
var nome = "Desenvolvendo";


nome.charAt(2); //"e"
nome.charCodeAt(0); //68
nome.concat("!"); //"Desenvolvendo!"
nome.indexOf('D'); //0
nome.replace('Desenvolvendo', ' Agora'); //"Agora"
nome.split('e');    //[ 'D', 's', 'nvolv', 'ndo!' ];
nome.length  // 13
nome.substring(1,5) //esen
```

Documentação

Boolean

```
var Verdade = true;
Verdade.toString(); //’true’
Verdade.valueOf(); //true
```

Valores falsos:

- false
- null
- undefined
- 0
- NaN

Documentação

Number

```
var nota = 10;

nota.toExponential(2);  //100

nota.toString(); // "10"

nota.valueOf(); // 10
```

Documentação

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Undefind & Null

O tipo undefined é retornado caso uma propriedade de um determinado objeto seja consultada e não exista

```
var carro = {}
carro.ano      // undefined
carro.modelo // undefined

carro.ano      = 2010
carro.modelo =  ‘gol’

carro // { ano: 2010, modelo: 'gol' }
```

Undefind & Null

O tipo null indica a ausência de valor em uma determinada propriedade já existente

carro.modelo = null

carro // { ano: 2010, modelo: null }

Object

Um objeto é uma coleção dinâmica de chaves e valores de qualquer tipo de dado.

Para criar um objeto carro:

- var carro = { ano: 2010, modelo: 'gol', 'cor do carro': 'vermelho' }

Para usar os atributos:

- carro.cor do carro = ‘vermelho’ // erro
- carro[‘cor do carro’] = ’vermelho’;

Array

Os Arrays são apenas objetos especiais que oferecem meios para acessar e manipular suas propriedades por meio de índices.

```
var carros= [];
console.log(carros);
carros[0] = "Ka";
carros[1] = "Corsa";
carros[2] = "Palio";
console.log(carros);
```

Documentação

Array

Arrays são construídos através de um construtor e possuem tamanho dinâmico:

```
var carros = new Array(20);

console.log(carros)

carros[21] = "Corsa";

console.log(carros);
```

Array

```
carros.unshift(“Gol”); // adiciona parâmetro no início do array
carros.shift(“Gol”); // remove parâmetro no início do array

carros.push(“Gol”); // adiciona parâmetro no final
carros.pop(); // remove a última posição

carros.forEach(function(elemento){
        console.log(elemento);
})
```

Date

```
var hoje = new Date();

hoje.getTime() ; // tempo em milissegundo

Date.parse("2017/05/11") //1494460800000

new Date(1494460800000) // Thu May 11 2017 00:00:00
```

Documentação

Operadores

Operador	Descrição
+	Efetuar soma de números ou Concatenação de strings
-	Efetuar subtração de números
*	Efetuar multiplicação de números
/	Efetuar divisão de números (Sempre divisão real)
%	Resto da divisão
++	Incremento
--	Decremento

Operadores

Operador	Descrição
==	Valor igual. (5 == “5”) retorna true
===	Valor e tipo iguais. (5 === “5”) retorna false
!=	Valor diferente. (5 != “5”) retorna false
!==	Valor e tipos diferentes. (5 !== “5”) returna true
>	Maior
<	Menor
>=	Maior ou Igual
<=	Menor ou Igual
&&	E (and)
||	OU (or)
!	NÃO (not)

Cuidado com == e != (coerção de tipos)

```
'' == '0'                  // false
0 == ''                    // true
0 == '0'                   // true
false == 'false'           //false
false = '0'                //true
false == null              //false
false == undefind          //true
```

A **coerção de tipos** (ou **conversão** de tipos) tenta realizar operações com tipos de dados **diferentes**.

Para evitar toda essa preocupação, você **deve** usar os operadores `===` e `!==`, que fazem a mesma verificação de igualdade, porém **NÃO** fazem coerção de tipos!

Function

Uma função é um objeto que contém um bloco de código executável.

Tipos de funções

- Declaration
- Expression

Documentação 1

Documentação 2

Function Declaration

A function declaration é carregada antes do código ser interpretado

```
function multiplicacao ( a, b){

    return a * b;

}
```

Function Expression

A function expression é carregada durante a interpretação do código

```
var multiplicacao = function (a, b){

    return a * b;

}
```

Quatro formas de chamar uma função

Universidade do Vale do itajaí (UNIVALI)
Centro de Ciência e Tecnologia da Terra e do Mar (CTTMar)
Curso de Ciência da Computação - Campus São José

Invocando uma função diretamente no escopo global

```
var carro = {ano:2010, modelo:gol, preco:15000}
var formulampostoA = function(preco){return preco*0.5;};
var formulampostoB = function(preco){return preco*0.2;};

var calcularPreco = function(produto, formulaImposto){
      return produto.preco + formulaImposto(produto.preco);
}

calcularPreco(carro, formuImpostoA); //22500
calcularPreco(carro, formuImpostoB); //18000
```

Retornando uma função

```
var comprar = function(){
        return function(){
                return “Boa Compra”;
        }
};
console.log(comprar()());
```

Invocando uma função por meio de objeto

```
var carro = {
        ano: 2010,
        modelo:’gol’,
        getAno:function(){
                return this.ano;
        }
}
carro.getAno(); // 2010
```

call e apply

Toda função possui os métodos call() e apply(). Eles são utilizados para indicar em qual escopo uma função deve ser executada

A diferença é basicamente a forma com é utilizado:

funcao.call(escopo, parametro1, parametro2)

funcao.apply(escopo,parametros)

call e apply

```
var getAno = function(){
	return this.ano;
}
var carro = {
	ano: 2010,
	modelo:’gol’,
	getAno:getAno
}

carro.getAno(); // 2010
getAno(); // undefined
getAno.call(carro); // 2010
```

call e apply

```
var getAno = function(extra){
        console.log(arguments);
        return this.ano + extra;
}
var carro = {
        ano: 2010,
        modelo:"gol",
        getAno:getAno
}

console.log(carro.getAno(2));
console.log(getAno.call(carro, 2)); // 2012
console.log(getAno.apply(carro,[ 2])); // 2012
```

Função por meio de atributos

```
var criarCarro = function (ano, modelo){
        return{
                ano:ano,
                modelo:modelo
        }
}
console.log(criarCarro(2012, “Fox”);
console.log(criarCarro(2015, “Fiesta”);
```

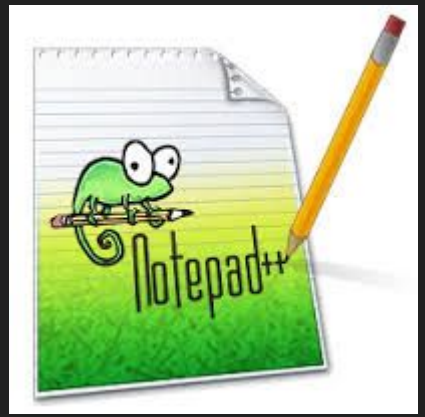
Invocando uma função por meio do operador new

```
var criarCarro = function (ano, modelo){
        this.ano      =  ano;
        this.modelo = modelo;
}
console.log(new criarCarro(2012, “Fox”);
console.log(new criarCarro(2015, “Fiesta”);
```

Ambiente de desenvolvimento

Editor de texto (sugestões):

- Sublime Text 2
- TextMate
- gedit
- Notepad++



Exercícios

1 - Escreva um código que calcule a soma 1 até 100. (obs: a resposta è 5050)

Exercícios

1 - Escreva um código que calcule a soma 1 até 100. (obs: a resposta è 5050)

```
var soma = 0;

for(var i =0; i <= 100; i++){

                soma = soma + i;
}

console.log(soma)
```

2 - Escreva uma função que crie uma lista de objetos carro (modelo, marca, ano e preço). Escreva uma função que exiba o carro mais caro e o mais barato da lista;