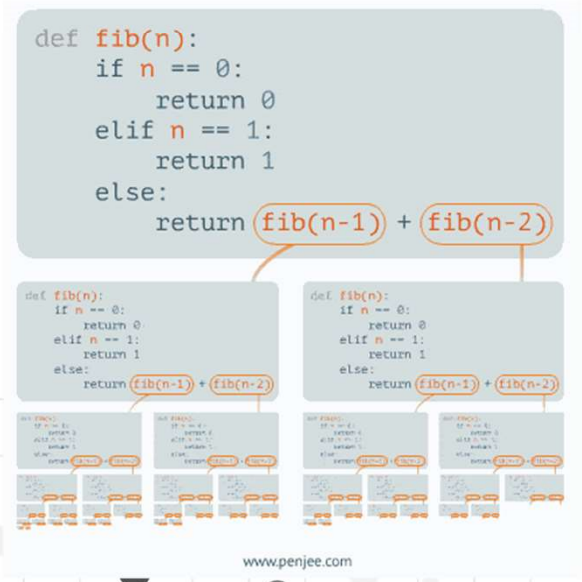
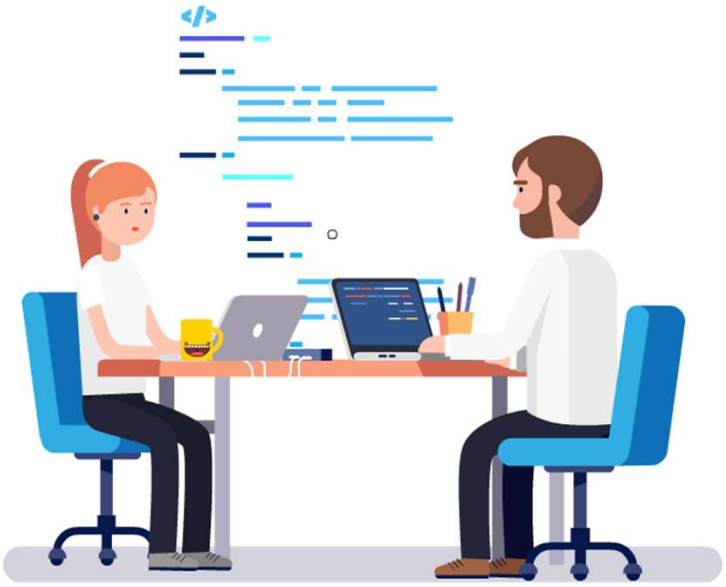
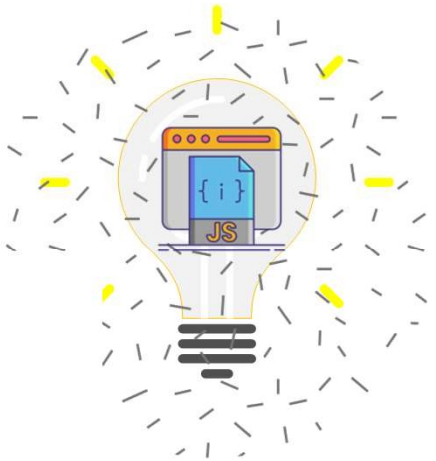
JAVASCRIPT – AULA 03







AGENDA 03/04

Parte I - JavaScript Básica

Parte II. Estrutura léxica

Parte III. Tipos, valores e variáveis

Parte IV. Expressões e operadores

Parte V. Instruções

Parte VI. Objetos

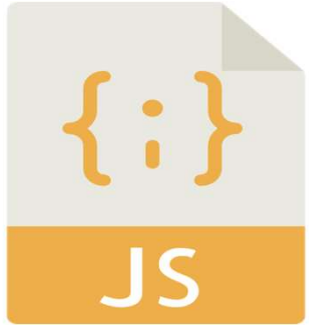
Parte VII. Arrays

Parte VIII. Funções

Parte IX. Classes e módulos

Parte X. JavaScript do lado do servidor

Parte XI. APIs de HTML5



INSTRUÇÕES

É uma maneira de “fazer algo acontecer” é alterar essa ordem de execução padrão, sendo que JavaScript tem várias instruções ou estruturas de controle que fazem justamente isso:

- As condicionais são instruções como if e switch que fazem o interpretador JavaScript executar ou pular outras instruções, dependendo do valor de uma expressão.
- Laços são instruções como while e for que executam outras instruções repetidas vezes.
- Saltos são instruções como break, return e throw que fazem o interpretador pular para outra parte do programa.

TIPOS DE INSTRUÇÕES EM JS

Instruções de expressões	Instruções compostas ou vazias	Instruções var / function (palavra-chave function é usada para definir funções)
imprimir= “Hello ” + name; i *= 3;	;	A instrução var declara uma (ou mais) variável. Aqui está a sintaxe: var nome_1 [= valor_1] [,..., nome_n [= valor_n]]

Os operadores de incremento e decremento, ++ e --	{
	x = Math.PI;
	cx = Math.cos(x);
	console.log("cos(π) = " + cx);
	}
		Por exemplo:
		var i; // Uma variável simples
		var j = 0; // Uma var, um valor var p, q; // Duas variáveis
		var imprimir = "hello" + name; // Um inicializador complexo
		var x = 2.34, y = Math.cos(0.75), r, theta; // Muitas variáveis
		var x = 2, // Diversas variáveis...
		f = function(x) { return x*x }, // cada uma em sua própria linha y = f(x);
		
		Aqui estão mais alguns exemplos de declarações de função:
		function hypotenuse(x, y)
		{ return Math.sqrt(x*x + y*y); // return está documentado na próxima seção } function factorial(n) { // Uma função recursiva if (n <= 1) return 1; return n * factorial(n – 1); }

counter++;	Considere o laço for a seguir	
As chamadas de função são outra categoria importante de instrução de expressão	// Inicializa um array a for(i = 0; i < a.length; a[i++] = 0) ;	

alert(imprimir);
window.close();
	if ((a == 0) || (b == 0)); // Opa! Esta linha não faz nada...
	o = null; // e esta linha é sempre executada.
	for(i = 0; i < a.length; a[i++] = 0) /* vazio */ ;	

OBJETOS EM JS

Em JavaScript, qualquer objeto pode ser usado como um template para criar novos objetos e usar as propriedades ou métodos definidos dentro dele.

Novos objetos também podem definir suas próprias propriedades ou métodos e podem ser associados como um protótipo para outro objeto.

Um objeto JavaScript é uma coleção de propriedades em que cada propriedade possui um nome e um valor, semelhante a Hash, Map ou Dictionary.

DEFININDO OBJETOS USANDO A NOTAÇÃO LITERAL

Um objeto pode ser criado usando a notação com colchetes {...} com uma lista opcional de propriedades, sendo que uma propriedade é um par "chave: valor", em que chave é uma string (também chamada de "nome da propriedade") e o valor pode ser qualquer coisa.

Aqui está a representação básica de um objeto pessoa na notação literal do objeto:

OBJETOS

Por exemplo: var pessoa = { id: "001", nome: “Sandra", Ativo: true, email:”...”}

1- primeira propriedade tem o nome id e o valor 001;

2- segunda tem o nome nome e o valorSandra;

3 - terceira tem o nome Ativo e o valor true

4- quarta tem o nome email e o valorsandra@gmail.com

```
1
2    var pessoa =
3    { id: "001", nome: "Sandra",
4     Ativo: true,
5     email: "sandra@gmail.com",
6      exibir : function()
7    {
8              return this.nome + " - " +
9               this.Ativo + " - " +this.email;
10    }
11   };
12   console.log(pessoa.exibir());
13
```

DEFININDO PROPRIEDADES - PATTERN CONSTRUCTOR

O padrão constructor permite definir parâmetros que restringem a passagem dos valores das propriedades pelos usuários ao instanciar os objetos.

Neste exemplo temos um objeto pessoa com as propriedades, id,nome, ativo e email onde as propriedades são atribuídas ao valor 'this' no corpo da função.

```
//Definindo propriedades usando o pattern constructor
function Pessoa (id, nome, ativo, email)
{
this.id =id
this.nome=nome
this.ativo = ativo
this.email = email
}

var p1 = new Pessoa (1,"Sandra","True","sandra@gmail")
console.log("Cógigo:",p1.id);
console.log("Nome:",p1.nome);
console.log("Ativo:",p1.ativo);
console.log("Email:",p1.email);
```

saída

```
Cógigo: 1
Nome: Sandra
Ativo: True
Email: sandra@gmail
```

ARRAY

Na linguagem JavaScript arrays são usados para armazenar mais de um valor em uma única variável. Para criar um array usamos o objeto Array.

```
1   var cor = [ "Azul", "Vermelho", "Preto", "Branco"]
2
3       cor.forEach
4       (function (cor, indice, array)
5        {
6            console.log("Cor:",cor, "-","Posição:",indice);
7        });
```

saída

```
Cor: Azul - Posição: 0
Cor: Vermelho - Posição: 1
Cor: Preto - Posição: 2
Cor: Branco - Posição: 3
```

Se você tem uma lista de itens

(uma lista de nomes de cores, por exemplo),

você poderia armazenar cada nome em uma variável individual assim:

Para acessar cada item no array você usa o índice atribuido a cada cor que começa com o valor zero(0).

Assim Cores[0] refere-se a primeira cor, cor[1] a segunda e cor[2] a terceira.

E para percorrer ou iterar sobre um array podemos usar o método forEach que lista cada elemento do array.

FUNÇÕES EM JAVASCRIPT

Uma função JavaScript nada mais é que um bloco de código que realiza uma tarefa, uma operação, sendo executada quando é chamada por alguém ou invocada.

Você define uma função usando a palavra-chave function seguida pelo nome da função e depois por parênteses () que podem incluir nomes de parâmetros separados por virgula: (param1,param2, etc...)

```
1   //Calculadora quatro operações
2   //Funções
3   function soma(v1,v2)
4   {return v1+v2}
5   function mult(v1,v2)
6   {return v1*v2}
7   function sub(v1,v2)
8   {return v1-v2}
9   function div(v1,v2)
10  {return v1+v2}
11  //Variável e resultado
12  var result = soma (5,6)
13  console.log ("Soma:",result)
14  var result = mult (5,6)
15  console.log ("Mult:",result)
16  var result = sub (5,6)
17  console.log ("Subt:",result)
18  var result = div (5,6)
19  console.log ("Divi:",result)
```

saída

```
Soma: 11
Mult: 30
Subt: -1
Divi: 11
```

FUNÇÕES EM JAVASCRIPT

Quando você chama a função você passa os valores usando argumentos

que dentro da função se comportam como variáveis locais.

Uma função pode ser executada:

Quando ocorrer um evento

Quando ela for invocada por outro código javascript

De forma automática (auto-invocação)

Uma função usa a instrução return para retornar um valor a quem a chamou.

Veja como funciona:

```
function ConvertFahrenheitCel(fahrenheit)
{
    return (5/9)* (fahrenheit-32)
}

let result1 = ConvertFahrenheitCel
console.log (result1)
console.log (typeof ConvertFahrenheitCel)
console.log ("-------------------------")
let result2 = ConvertFahrenheitCel (10)
console.log (result2)
console.log (typeof ConvertFahrenheitCel())
```

saída

```
> ƒ ConvertFahrenheitCel(fahrenheit)
  function
  -------------------------
  -12.222222222222223
  number
```