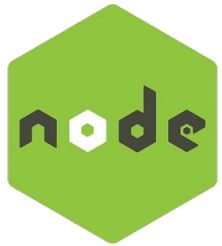
NodeJS



Prof. Celso Henrique Masotti

Necessidades Prévias

- Ter o node.JS instalado;
- Ter o gerenciador de pacotes (NPM) instalado;
- Ter um editor de códigos IDE(*), dê preferência ao “Visual Studio Code”
- Conhecer JavaScript básico.

(*) IDE, do inglês Integrated Development Environment ou Ambiente de Desenvolvimento Integrado, é um programa de computador que reúne características e ferramentas de apoio ao desenvolvimento de software com o objetivo de agilizar este processo, como um editor de texto para a digitação de códigos.

Você verá nesta aula:

Objetivo: conhecer e trabalhar com funções clássicas e arrow.

Funções
Funções Nativas
Funções Desenvolvidas
Arrow Functions
Função na Prática

Funções

Funções são blocos de construção fundamentais em JavaScript. Uma função é um procedimento de JavaScript - um conjunto de instruções que executa uma tarefa ou calcula um valor. Para usar uma função, você deve defini-la em algum lugar no escopo do qual você quiser chamá-la.

A declaração **function** define uma função com os especificados parâmetros.

name (nome): O nome da função;
param (parâmetros): O nome de um argumento a ser passado para a função. Uma função pode ter atè 255 argumentos;
statements (declarações): As instruções que compõem o corpo da função sempre ficam no interior das chaves { }.

```
function name([param,[, param,[..., param]]]) {
   [statements]
}
```

Funções

Toda função é um microprograma e também é um **objeto** *function*.

Funções não são como procedimentos (*procedure*). Uma função sempre retorna um valor, mas um procedimento pode ou não retornar um valor.

A linguagem JavaScript possui várias funções prontas em sua biblioteca. Estas são chamadas de funções nativas. Mas, também podemos criar funções específicas para usarmos quando desejarmos.

Uma função só entra em atividade quando a chamamos pelo nome. Também devemos nos atentar quanto ao fato desta pedir ou não valor para algum parâmetro para que funcione adequadamente.

Funções Nativas

Como foi dito, a linguagem javascript possui uma grande variedade de funções prontas em sua biblioteca e que pode ser chamada quando desejarmos.

Exsitem funções para tempo (data, hora, dia, mês, etc), para trabalhar com strings (trocar textos, alterar caixa, realizar contagens, localizar caracter, retirar espaços, etc.) e também para manipular variáveis.

Veja mais sobre o tema.

Funções Desenvolvidas

No exemplo ao lado chamamos a função “soma” enviando os valores para os parâmetros “v1” e “v2”. A função soma recebe os valores e, na área de declaração, “v1” e “v2” são somados e seu resultado é devolvido à linha que realizou a chamada. O resultado então é armazenado na variável “total” para ser impresso no console posteriormente.

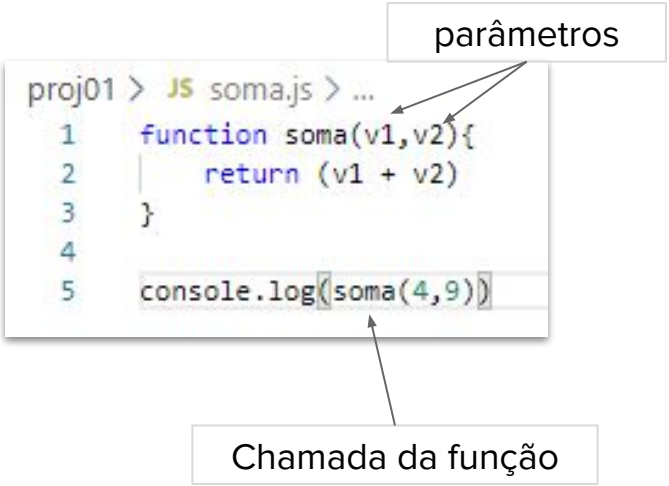
```
function soma(v1,v2){
    return (v1 + v2)
}

var total = soma(14,55)

console.log(total)
```

Funções Desenvolvidas

Exemplo de feitura: Crie um arquivo chamado “soma.js”. Desenvolva uma função que receba dois valores nos parâmetros e imprima o resultado da soma no console.




```
C:\NodeEstudos\proj01>node soma.js
13
```

Arrow Functions

Vimos até agora a estrutura clássica para funções. Mas, com o ES6 esta estrutura foi modificada para adquirir novos recursos de extrema importância.

Função Clássica

```
function soma(a,b){
    return a + b;
}
```

Função Arrow

```
const soma = (a, b) => {
    return a + b
}
```

```
const soma = function(a, b) => {
    return a + b
}
```

```
const soma = (a,b) => a + b
```

ou

```
const soma = (a,b) =>
    a + b
```

Função na Prática

1 - Crie um arquivo javascript contendo uma função que substitua palavras “Seu” e “Dona” por “Senhor” e “Senhora” como forma de tratamento.

2 - Crie um arquivo javascript que, recebendo um número qualquer menor que 10 carateres, preenche de zeros à sua esquerda. Exemplo: 198711 -> 0000198711.

3 - Crie um arquivo javascript que, recebendo dois valores sendo o primeiro numérico (1 ou 2) e o segundo “string” (nome da pessoa) devolva:
consolo.log: 1, nome (Sr nome); 2, nome (Sra, nome);
console.warn: 3, nome (Aviso de tratamento indefinido);
console.error: qualquer parâmetro cujo nome seja menor que 3 caracteres.

Obrigado

Celso Henrique Masotti