Universidade Cruzeiro do Sul

JS

Funções em

JAVA SCRIPT

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

O QUE SÃO FUNÇÕES?

Em JavaScript, funções são blocos de código reutilizáveis que realizam tarefas específicas. Elas são como pequenas máquinas que você pode acionar para executar uma ação.

Por que usar funções?

- Organização: Dividir o código em funções torna-o mais fácil de entender e manter.
- Reutilização: Uma vez criada, uma função pode ser chamada várias vezes em diferentes partes do seu código.
- Abstração: Funções permitem que você esconda a complexidade interna de um processo, expondo apenas a interface necessária.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

SINTAXE BÁSICA

```
function nomeDaFuncao(parametro1, parametro2, ...) {
  // Corpo da função: instruções a serem executadas
  return valor; // Opcional: retorna um valor
}
```

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

function: Palavra-chave que define uma função.

nomeDaFuncao: Nome que você escolhe para identificar a função.

parametro1, parametro2: Valores que você pode passar para a função (opcionais).

return: Palavra-chave usada para retornar um valor da função.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

TIPOS DE FUNÇÕES

Funções Void (sem retorno)

Funções void não retornam valor. Elas são utilizadas principalmente para executar ações, como imprimir mensagens no console ou modificar o DOM.

```
function imprimirMensagem(mensagem) {
  console.log(mensagem);
}
imprimirMensagem("Olá, mundo!"); // Imprime "Olá, mundo!" no console
```

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

function imprimirMensagem(mensagem): Define uma função chamada imprimirMensagem que recebe um parâmetro mensagem.
console.log(mensagem): Imprime o valor do parâmetro mensagem no console.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

Funções com Retorno

Funções com retorno podem devolver um valor para ser utilizado em outras partes do código.

```
function somar(a, b) {
  return a + b;
}
const resultado = somar(5, 3);
console.log(resultado); // Imprime 8 no console
```

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

function somar(a, b): Define uma função chamada somar que recebe dois parâmetros a e b.
return a + b;: Calcula a soma de a e b e retorna o resultado.
const resultado = somar(5, 3);: Chama a função somar com os argumentos 5 e 3 e armazena o resultado na variável resultado.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

Arrow Functions

Arrow functions são uma sintaxe mais concisa para definir funções.

```
const dobrar = numero => {
  return numero * 2;
};
const resultado = dobrar(5);
console.log(resultado); // Imprime 10 no console
```

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

const dobrar = numero => { ... };: Define uma arrow function chamada dobrar que recebe um parâmetro numero.
return numero * 2;: Multiplica o parâmetro numero por 2 e retorna o resultado.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

Exemplo sem as chaves (para uma única expressão):

```
const dobrar = numero => numero * 2;
```

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

FUNÇÃO ANINHADA

O que são funções dentro de funções?

Em JavaScript, é possível definir uma função dentro de outra. Essa técnica, conhecida como aninhamento de funções, cria um escopo local específico para a função interna. Isso significa que as variáveis declaradas dentro da função interna só são acessíveis dentro dela e da função externa que a contém.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

Por que usar funções aninhadas?

- Encapsulamento: Permite ocultar detalhes de implementação, tornando o código mais limpo e organizado.
- Closures: As funções internas formam closures, permitindo que elas acessem e usem variáveis da função externa, mesmo após a função externa ter terminado de executar.
- Modularidade: Ajuda a organizar o código em partes menores e mais gerenciáveis.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

```
function alertavermelho() {
        alert('fuja loco, o reator nuclear vai explodir');
    }
    function temperaturaDoReator(temperatura) {
        if (temperatura > 200) {
            alertavermelho();
        }
    }
    temperaturaDoReator(100);
    temperaturaDoReator(300);
```

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

O código define duas funções em JavaScript: alertavermelho() e temperaturaDoReator(temperatura). Cada uma delas tem uma função específica:

alertavermelho(): essa função é simples e tem apenas uma ação: exibir um alerta na tela com a mensagem "fuja loco, o reator nuclear vai explodir". Essa função serve para avisar o usuário de uma situação de perigo iminente.
temperaturaDoReator(temperatura): essa função é um pouco mais complexa. Ela recebe um parâmetro chamado temperatura que representa a temperatura atual de um reator nuclear. A função verifica se essa temperatura é superior a 200. Se for, ela chama a função alertavermelho(), ou seja, exibe o alerta de perigo.

Prof. Welington Soares - 2024

Universidade Cruzeiro do Sul

JS

As duas funções são definidas no início do código.

- **Chamada da função temperaturaDoReator()**: o código chama a função temperaturaDoReator() duas vezes, passando os valores 100 e 300 como temperatura do reator.
- **Primeira chamada**: quando a função é chamada com o valor 100, a condição if (temperatura > 200) não é verdadeira, pois 100 não é maior que 200. Portanto, a função alertavermelho() não é chamada e nada acontece.
- **Segunda chamada**: quando a função é chamada com o valor 300, a condição if (temperatura > 200) é verdadeira, pois 300 é maior que 200. Nesse caso, a função alertavermelho() é chamada, e o alerta "fuja loco, o reator nuclear vai explodir" é exibido na tela.

BORA PROGRAMAR?

Prof. Welington Soares - 2024