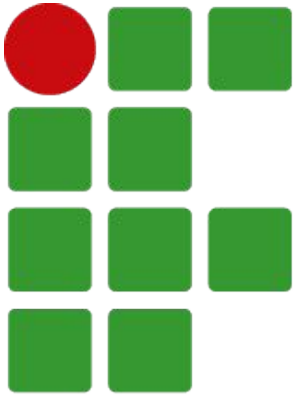


INSTITUTO FEDERAL
Paraná
Campus Paranaguá

JavaScript
Arrays, Funções e Objetos

Desenvolvimento Web

Prof. Diego Stiehl

INSTITUTO FEDERAL

Arrays

- Podemos definir arrays de duas formas

```
let nomes = ['Fritz', 'Franz', 'Berlin'];
let idades = new Array(5, 3, 1);
```

- Para acessar um elemento

```
const franz = nomes[1]; // 'Franz'
idades[2] = 2; // Idade mudada para 2
```

- Tamanho

```
const quantidade = nomes.length; // 3
```

INSTITUTO FEDERAL

Arrays no Console

- Podemos ver a estrutura dos Arrays no Log

INSTITUTO FEDERAL

Diferentes Tipos

- Os dados de um Array não precisam ser todos do mesmo tipo

```
const diego = [
  'Diego',
  'Stiehl',
  1988,
  3,
  false];
```

INSTITUTO FEDERAL

Adicionando e Removendo

```
let caes = ['Fritz', 'Franz', 'Berlin'];
caes.push('Frida');  // Adiciona no fim
caes.unshift('Rex'); // Adiciona no começo
caes.pop();          // Remove do último item
caes.shift();        // Remove do primeiro item
```

INSTITUTO FEDERAL

Percorrendo Array

- Existem várias formas de percorrer um Array
  - Por ora, utilizaremos o método “clássico”

```
let caes = ['Fritz', 'Franz', 'Berlin'];
for (let i = 0; i < caes.length; i++) {
  const cao = caes[i];
  console.log(`Cachorro ${i + 1}: ${cao}`);
}
```

INSTITUTO FEDERAL

Prática 4 - Arrays

- Peça para o usuário informar um número (N)
- Solicite que ele informe palavras N vezes
- Crie um Array e guarde cada uma delas
- Utilize um laço de repetição para imprimir as palavras do Array na ordem reversa

INSTITUTO FEDERAL

Funções

- Criamos funções quando queremos agrupar ou reaproveitar trechos de código
- Podemos passar parâmetros
  - Dinamicamente tipados
- Podemos retornar um valor
  - De qualquer tipo
- Podemos invocar (chamar) a função

INSTITUTO FEDERAL

Funções

- Sintaxe Básica

```
function nomeFuncao(param1, param2, param3) {
  // Código a ser executado
  // Se desejado, pode retornar um valor
  return 1;
}
```

INSTITUTO FEDERAL

Exemplo

```
// Declaração
function calcularIdade(anoNascimento) {
  return 2019 - anoNascimento;
}

// Invocação
var minhaIdade = calcularIdade(1988); // 31
```

INSTITUTO FEDERAL

Prática 5 - Aposentadoria

- Crie uma função que calcule e mostre no console o tempo restante para aposentadoria de uma pessoa
  - Ler nome e idade
  - Chamar `calcularIdade()` criada anteriormente
  - Considere idade para aposentadoria = 65 anos
  - Verificar se pessoa já está aposentada

INSTITUTO FEDERAL

Function Declaration

- A forma como declaramos nossa função é conhecida como **function declaration**
  - Definimos uma função, com seu nome fixo

```
function meDigaOla(meuNome) {
  return `Olá ${meuNome}`;
}
```

INSTITUTO FEDERAL

Function Expression

- Outra forma conhecida é chamada de **function expression**
  - Declaramos uma função anônima
  - Atribuímos essa função a uma variável

```
let meDigaOla = function(meuNome) {
  return `Olá ${meuNome}`;
}
```

- Por quê? Como?
  - Funções são como objetos do tipo function

INSTITUTO FEDERAL

Expressions X Statements

- Expressions
  - Tudo que, ao ser executado, gera um valor imediato
  - Pode ser atribuído a uma variável
- Declarations / Statements
  - Trechos JS que não produzem resultados imediatos
- **Guarde estas informações para uso futuro**

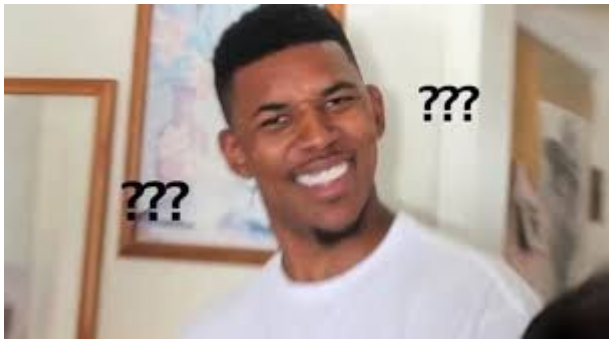
Prática 6 - Calcular Gorjeta

- Hugo e sua família foram a três restaurantes no último feriado. Nestes lugares, os garçons sempre cobram uma gorjeta.
- Hugo, que é muito hacker, decidiu criar uma calculadora para aplicar sua peculiar lógica de pagamento de gorjetas.
- Para calcular a gorjeta, Hugo considera justo uma taxa de 20% em refeições de valor inferior a R$ 50, 15% para refeições de valor entre R$ 50 e R$ 200 e 10% para valores acima de R$ 200.
- Peça para o usuário informar o nome dos três restaurantes.
- Peça para o usuário informar os três valores totais das contas.
- Crie uma função para calcular o valor da gorjeta.
- Armazene os resultados em três arrays: um com o nome de todos os restaurantes, outro com todas as gorjetas e outro com todos os valores originais.
- Crie uma função que receba os três arrays como parâmetro e mostre o detalhamento de cada uma das contas. A impressão de cada detalhamento deve ser feita por outra função, no seguinte formato: "Restaurante do Zé Birola - [Valor: R$ 60.00 | Gorjeta: R$ 9.00 | Total: R$ 69.00]"

INSTITUTO FEDERAL

Objetos

- Um objeto JavaScript é um conjunto de pares no formato chave: valor
- Serve para agrupar variáveis que pertencem ao mesmo escopo, sem uma ordem definida
- Um objeto NÃO tem uma classe
  - JavaScript não tem classes
- Mas tem “herança”



INSTITUTO FEDERAL

Object Literal

- Um objeto JavaScript pode ser **criado** utilizando um Object Literal
- Sintaxe: utilizar chaves → { }
- Definimos atributos e valores separados por dois pontos → `{ nome: 'Diego Stiehl' }`
- Separar os vários atributos com vírgula

```
{
  nome: 'Diego Stiehl',
  idade: 31
}
```

INSTITUTO FEDERAL

Definição de Objetos

- Assim como nos Arrays, temos uma forma alternativa para criar Objetos:

```
let pessoa = new Object();
```

- Mesma coisa que fazer:

```
let pessoa = {};
```

INSTITUTO FEDERAL

Variável

- Após criado, um objeto pode ser atribuído a uma variável

```
let diego = {
  nome: 'Diego',
  sobrenome: 'Stiehl',
  idade: 31
};
```

INSTITUTO FEDERAL

Acessando Atributos

- Para acessar os valores das propriedades usamos a notação com ponto (.) sobre a variável do objeto

```
const nomeCompleto = diego.nome + ' ' + diego.sobrenome;
console.log(nomeCompleto);

console.log(diego.idade);
const idadeDoDiegoDaquiA3Anos = diego.idade + 3;
console.log(idadeDoDiegoDaquiA3Anos);
```

INSTITUTO FEDERAL

Acessando Atributos

- Opcionalmente, também podemos usar a notação no formato de array (com chaves)
  - Melhor para explorar um objeto desconhecido

```
const nomeCompleto = diego['nome'] + ' ' + diego['sobrenome'];
console.log(nomeCompleto);

const atributoIdade = 'idade';
console.log(diego[atributoIdade]);
const idadeDoDiegoDaquiA3Anos = diego[atributoIdade] + 3;
console.log(idadeDoDiegoDaquiA3Anos);
```

INSTITUTO FEDERAL

Alterando Dados

- As mesmas notações podem ser usadas para alterar os dados do objeto

```
diego.sobrenome = 'Maradona';
diego['idade'] = 65;

const nomeCompleto = `${diego.nome} ${diego.sobrenome}`;

console.log(`Nome: ${nomeCompleto} - Idade: ${diego.idade}`);
```

INSTITUTO FEDERAL

Tipos de Dados

- Que tipos de dados eu posso colocar nos valores dos atributos?
  - R: Tudo que você possa colocar em variáveis
    - string, number, boolean, undefined, null
    - Outros Objetos
    - Array (que é um objeto na verdade)
    - Funções (se comportará como método)

INSTITUTO FEDERAL

Tipos de Dados

```
let diego = {
  nome: 'Diego',                          // string
  sobrenome: 'Stiehl',                    // string
  anoNascimento: 1988,                    // number
  admin: true,                            // boolean
  cachorros: ['Fritz', 'Franz', 'Berlin'], // Array
  profissao: {                            // object
    cargo: 'Professor',
    atribuicoes: 'Ensinar web pra piazada'
  },
  nomeCompleto: function() {              // function
    return `${this.nome} ${this.sobrenome}`
  }
};
```

INSTITUTO FEDERAL

Acessando Atributos

```
if (diego.admin) {
  console.log(diego.nomeCompleto());
  console.log("Cachorros:");
  for (let i = 0; i < diego.cachorros.length; i++) {
    console.log(diego.cachorros[i]);
  }
  console.log("Profissão:");
  console.log(diego.profissao.cargo);
  const profissao = diego.profissao;
  console.log(`Atribuições: ${profissao.atribuicoes}`);
}
```

INSTITUTO FEDERAL

Prática 7

- Reconstrua a Prática 1, mas agora usando objetos para armazenar os dados das duas pessoas.
- Os objetos da duas pessoas devem conter métodos que calculem e retornem seus IMCs
  - Os métodos também devem salvar o IMC calculado no próprio objeto ao serem invocados