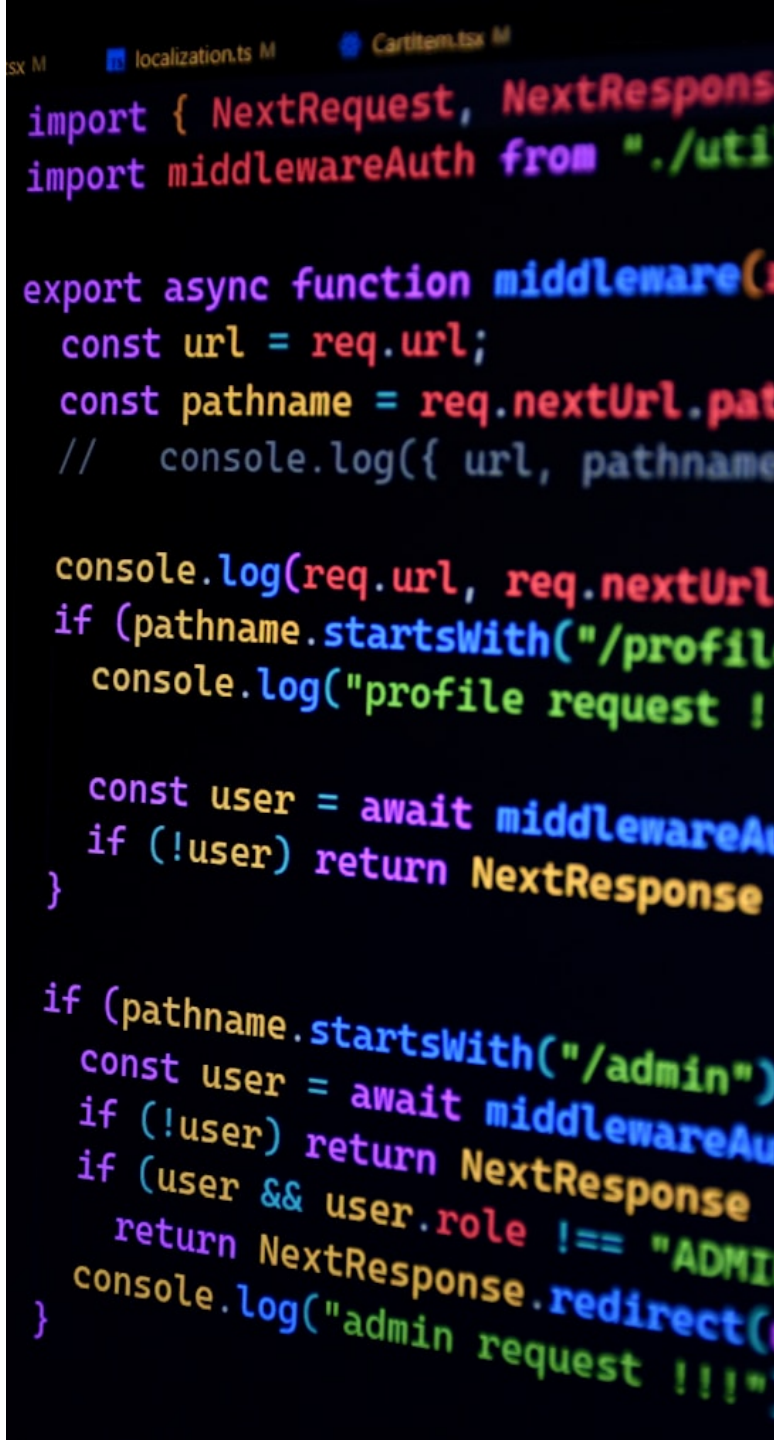
5954024 - Introdução ao Desenvolvimento Web

JavaScript - Teoria e Prática - Parte 1

Prof. Dr. Denis M. L. Martins
DCM | FFCLRP | USP

Objetivos de Aprendizagem

- Entender o papel do JavaScript no desenvolvimento Web
- Identificar características fundamentais (tipagem, execução, etc.)
- Escrever código básico em JavaScript
  - Declarar variáveis e utilizar tipos de dados
  - Utilizar estruturas condicionais e de repetição
  - Criar e utilizar funções simples



Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Ecossistema

HTML define a **estrutura**.

CSS define a **aparência**.

JavaScript define o **comportamento**.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

JavaScript

- Linguagem de programação que roda nativamente em navegadores Web.
  - Engine dedicada para executar código JavaScript. Exemplo: V8 desenvolvida pela Google para o Chrome.
  - Compilação JIT de bytecode em código de máquina.
  - Linguagem mais popular de 2025, segundo pesquisa da StackOverflow.
- Criada para o navegador Netscape.
- Na imagem: Brendan Eich, criador da JavaScript e co-fundador do Mozilla. Fonte da imagem: Wikipedia

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo prático

- Abra o navegador Web e selecione a opção para exibir as ferramentas de desenvolvedores. No Firefox/Chrome/Edge: **Ctrl + Shift + I** ou **F12**.
- Vá em *Console* e digite o comando abaixo:

```
console.log("Hello World!");
```

Case-sentive: **console.log()** é diferente de **Console.Log()**

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo prático (cont.)

O *Console* um ambiente REPL (Read-Eval-Print Loop). Escreva uma expressão JavaScript, pressione *Enter*, e o console vai mostrar o resultado imediatamente.

Tente as expressões abaixo, uma de cada vez:

```
3 + 3
```

```
"Hello, " + "World!"
```

```
let planet = "Earth";
console.log(planet);
```

```
window.location.href
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo prático (cont.)

Para escrever expressões multilinha ou em bloco, pressione *Shift + Enter*.

Tente o bloco abaixo:

```
function sayHello(userName) {
  return `Hello, ${userName}!`;
}
```

Para executar a função:

```
sayHello("Turing");
```

Note que a convenção para JavaScript é camelCase.

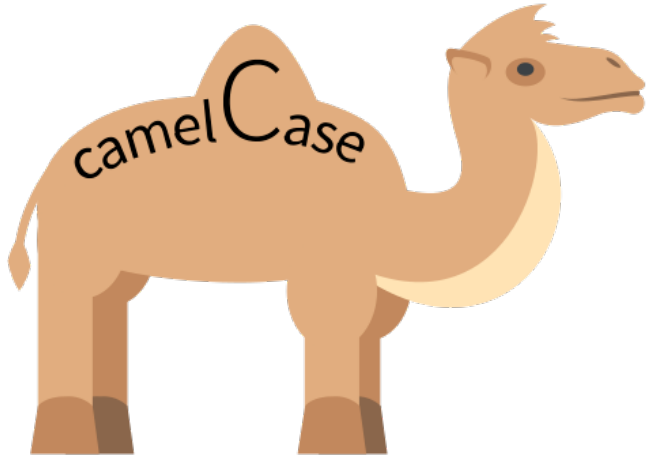


Fig. .1: A origem do Camel Case vem da corcova do Camelo.

Fonte da Imagem: Wikipedia.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Tipagem

Tipagem Dinâmica: O tipo é inferido na hora da execução.

```
let x = 10;       // Número
x = "texto";      // Agora é String
// Não há erro de compilação!
```

- `//` para comentários de uma linha.
- `/* ... */` para comentários multi-linha.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Sintaxe Básica

- Comentários: `// linha única` / `/* bloco */`
- Delimitadores: `{}` (blocos), `()` (parâmetros), `[]` (arrays)
- Operadores aritméticos, lógicos, de comparação
- **Flexível**: Ponto e vírgula (;) é opcional em muitas linhas.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Tipos de Dados Primitivos

Tipo	Exemplo	Observação
Number	`42`, `3.14`	IEEE-754 double
String	`"Olá"`	Imutável
Boolean	`true` / `false`	
Symbol	`Symbol('id')`	Único, não pode ser convertido automaticamente
BigInt	`9007199254740991n`	Para inteiros maiores que Number.MAX_SAFE_INTEGER
undefined	variável não inicializada	
null	valor intencionalmente vazio	

```
typeof "texto"; // "string"
typeof 123;     // "number"
typeof true;    // "boolean"
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Variáveis: Declaração e Escopo

- `var`: Funcionalidade antiga. Função de escopo (funcional). Pode ser redeclarada.
- `let`: Escopo de bloco (`{}`). Não permite redeclaração no mesmo escopo.
- `const`: Para constantes. Valor não pode ser reatribuído após declaração.

```
var idade = 25;
let nome = "Ana";
const PI = 3.14159;
nome = "Carlos"; // Reatribuição permitida em let/var
```

```
{
  let x = 1; // Bloco
}
console.log(x); // Erro: x não é definido
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Operadores e Conversão de Tipos

- **Operadores Aritméticos:** `+`, `-`, `*`, `/`, `%` (módulo).
- **Operadores Lógicos:** `&&` (E), `||` (Ou), `!` (Não).
- **Conversão Implícita:** JS tenta converter tipos automaticamente.
  - `"5" + 1 = "51"` (Concatenação de Strings).
  - `"5" - 1 = 4` (Subtração numérica).

Cuidado com `==` vs `===`:

```
let a = 5;
let b = "5";
console.log(a == b); // true (Igualdade de valor)
console.log(a === b); // false (Igualdade estrita de tipo e valor)
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Precedência de Operadores

```
let x = 100 + 25 * 2;
console.log(x);
```

```
let x = 100 / 25 * 2;
console.log(x);
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Estruturas Condicionais

```
const temperatura = 30;
if (temperatura > 25) {
  console.log("Está quente");
} else {
  console.log("Não está quente");
}
```

Operador ternário `?`

```
const status = temperatura >= 25 ? "Está quente" : "Não está quente";
console.log(status);
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Switch e Operador `in`

- **Switch**: Útil para múltiplas condições com o mesmo valor.
  - Mais legível que uma longa cadeia de else if.
- `case`: Cada caso termina com `break` para evitar "*fall-through*".
  - `default`: Executa se nenhum `case` for correspondido.

```
const dia = 2;
switch(dia) {
  case 1: console.log("Domingo"); break;
  case 2: console.log("Segunda"); break;
  default: console.log("Outro dia");
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Estrutura de Repetição

```
for (let i = 0; i < 5; i++) {
  console.log(`Número: ${i}`);
}
```

```
let contador = 0;
while (contador < 5) {
  console.log(contador);
  contador++;
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Arrays e Objetos

Arrays: Listas ordenadas de valores.

```
const lista = [1, 2, 3];
console.log(lista[0]);
```

```
const array = new Array(1, 2, 3);
console.log(array[0]);
```

Objetos: Coleções chave-valor.

```
const pessoa = { nome: "Ana", idade: 20 };
console.log(pessoa.nome);
console.log(pessoa["nome"]);
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Propriedades e Métodos de Array

```
const numeros = [1, 2, 3];
console.log(numeros.length);
```

```
numeros.length = 2; // configura o tamanho do array
console.log(numeros);
```

```
const cidades = ["Ribeirão Preto", "Campinas"];
cidades.push("Recife");
console.log(cidades);
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Propriedades e Métodos de Array (cont.)

map: Cria um novo array com base no atual.

```
const numerosSelecionados = [3, 5, 7];
const dobro = numerosSelecionados.map(n => n * 2);
console.log(dobro);
```

filter: Filtra elementos que atendem a uma condição.

```
function verificaMaioridade(value) {
  return value > 18;
}

let idades = [12, 5, 40, 17, 21];
const acimaDe18 = idades.filter(verificaMaioridade);
console.log(acimaDe18)
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Funções: Declaração e Expressão

Declaração: **function nome() {}** (Hoisting).

```
function saudacao(nome) {
  return "Olá, " + nome + "!";
}
console.log(saudacao("Mundo"));
```

Expressão: **const soma = () => ...** (Arrow Function).

```
const multiplicar = (a, b) => a * b;
console.log(multiplicar(5, 10));
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Objetos

Estrutura que agrupa dados relacionados e comportamentos.

- Delimitadores: Chaves { } definem o início e fim do objeto.
- Propriedades: Nomeadas por strings ou identificadores válidos.
- Formato: **nome: valor** dentro das chaves.

```
const pessoa = {
  nome: "Ada",
  idade: 25,
  ativo: true,
  cidade: "Ribeirão Preto"
};
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Acessando Propriedades

Dot Notation

```
const nome = pessoa.nome;
console.log(pessoa.idade);
```

Bracket Notation

```
// Com aspas simples
console.log(pessoa["nome"]);
```

```
// Com espaços no nome
const novaPessoa = { "nome completo": "Ana Silva" };
console.log(novaPessoa["nome completo"]);
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Métodos em Objetos

Palavra-chave **this** referencia o próprio objeto:

```
const pessoaAda = {
  nome: "Ada",
  saudar: function() {
    console.log(`Olá, sou ${this.nome}!`);
  }
};

pessoa.saudar();
```

Em métodos, **this** se refere ao objeto que contém o método.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Métodos em Objetos (cont.)

```
const produto = {
  nome: "Notebook",
  preco: 4500,
  aplicarDesconto: function(percentual) {
    const desconto = this.preco * (percentual / 100);
    return this.preco - desconto;
  }
};

console.log(produto.aplicarDesconto(10));
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Métodos em Arrow Functions

```
const usuario = {
  nome: "ada",
  email: "ada@email.com",
  // Método com arrow function
  enviarEmail: () => console.log(`Enviando para ${this.email}`),
  // Getter para propriedade calculada
  getNomeCompleto() { return this.nome; }
};
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Integrando JavaScript em HTML

Inline: Código diretamente em elemento HTML

```
<button onclick="alert('Cliquei!')">Clique Aqui</button>
```

Internal: Script embutido no arquivo **.html** (entre tags **<script>**)

```
<script>
  alert("Carregando página...");
</script>
```

External (recomendado): Arquivo separado **.js** vinculado via **src**

```
<head>
  <script src="scripts/app.js"></script>
</head>
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Exemplo

```html
<!DOCTYPE html>
<html>
<head>
  <title>Minha Página</title>
  <script src="scripts/app.js"></script>
</head>
<body>
  <h1>Bem-vindo!</h1>
  <button onclick="saudar()">Clique para dizer Olá</button>
</body>
</html>
```

No arquivo **app.js**:

```javascript
function saudar() {
  console.log("Olá, mundo!");
}
```


Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

TypeScript

- TypeScript é um superconjunto sintático estrito de JavaScript
- Adiciona tipagem estática
- Ajuda a verificar por erros antes de executar o código
- Na imagem: Anders Hejlsberg, core developer da TypeScript (e C# e Turbo Pascal e Delphi). Fonte da imagem: Wikipedia.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Conclusão

Conceitos-Chave:

- JavaScript é a principal linguagem para criar interatividade na web
- Permite implementar lógica, tomada de decisão e repetição
- É uma linguagem dinâmica e amplamente utilizada no mercado

Próxima aula: Manipulação DOM.

Material adicional YouTube: Understanding the V8 JavaScript Engine | JavaScript tutorial for beginners | Entrevista com Anders Hejlsberg

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Lab Prático

Objetivo: Criar um objeto "Biblioteca" com métodos de empréstimo.

1. **Passo 1:** Crie um arquivo **biblioteca.js**.
2. **Passo 2:** Declare um objeto **livro** com propriedades: título, autor, ano, emprestado (boolean).
3. **Passo 3:** Adicione métodos: **emprestar()**, **devolver()**, **informacoes()**.
4. **Desafio Extra:** Crie um array de livros e itere para mostrar todos os títulos disponíveis.

Resultado Esperado: Um objeto funcional que simula empréstimo de livros com console logs claros.

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br

Dúvidas e Discussão

Prof. Dr. Denis M. L. Martins | martins.denis@usp.br