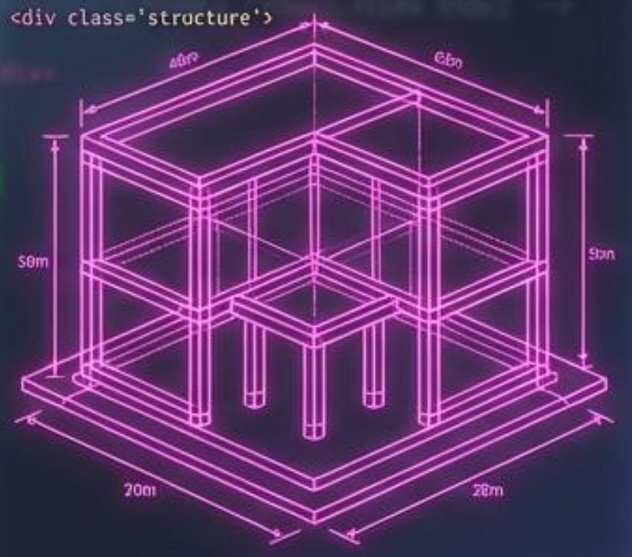
JavaScript: O Cérebro da Web

Do zero à interatividade completa.

A linguagem que dá vida e comportamento às páginas.

A Tríade da Web

HTML
Conteúdo
Estrutura



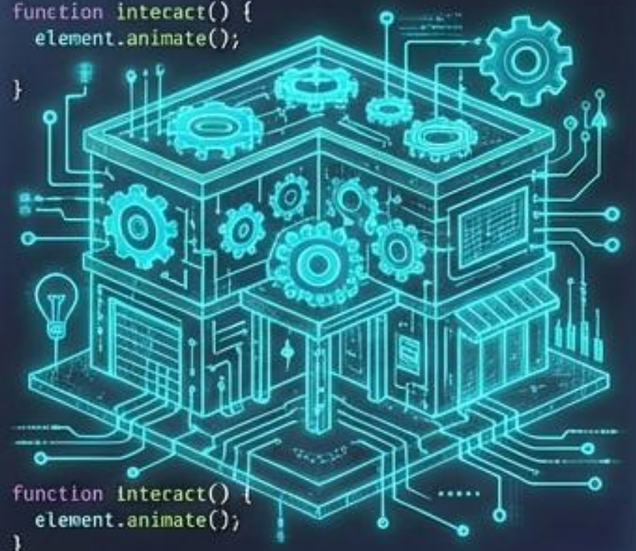
CSS
Layout
Estilo

```
.building {
  backgroond: #0DFF7F;
  border-radius: 10px;
}
```

```
.building {
  background: #0DFF7F;
  border-radius: 10px;
}
```

JavaScript
Comportamento
Interatividade

```
function intecact() {
  element.animate();
}
```



```
function intecact() {
  element.animate();
}
```

O HTML define o que é, o CSS define como se parece, e o JavaScript define o que ele faz.

Onde o Código Habita

Dentro do <head>

```
<!DOCTYPE html>
<html>
<head>
  <title>JavaScript</title>
  <meta charset="UTF-8">
  <script>
    alert("JavaScript no Head");
  </script>
</head>
<body>
</body>
</html>
```

Dentro do <body>

```
<!DOCTYPE html>
<html>
<head>
  <title>JavaScript</title>
  <meta charset="UTF-8">
</head>
<body>
  <script>
    alert("JavaScript no Body");
  </script>
</body>
</html>
```

Arquivo Externo

```
<!DOCTYPE html>
<html lang="en">
 <head>
   <meta charset="UTF-8">
   <title>JavaScript</title>
   <script src="meu_script.js"></script>
 </head>
 <body>
   <h1>Bem Vindo Alunos</h1>
 </body>
</html>
```

Boas Práticas: Mantenha seu JavaScript em arquivos externos (.js) para organização e performance.

Variáveis: As ‘Caixinhas’ da Memória

```
function init() {
  console.log('Variables loaded.');
}
```

```
var user = 'old';
```

```
let count = 0;
count = 1;
```

```
const PI = 3.14;
```

var
Legado / Global

let
Mutável - Pode trocar de conteúdo

const
Imutável - Uma vez cheia, não muda

Regras de Sintaxe

- Case sensitive (nome ≠ Nome)
- Não começar com números
- Sem caracteres especiais (exceto _ e $)

Boas Práticas: Use nomes descritivos e prefira `let` e `const` para código mais seguro.

Tipos de Dados Primitivos

String

"Texto entre aspas"

Sequência de caracteres.

Number

10
Int

10.75
Float

Boolean

ON true

OFF false

Null / Undefined

Null

Undefined

```
var teste = "Texto";
// JS decide o tipo automaticamente
```

Transformação e Manipulação

Casting (Conversão)


```
var num = parseInt("10"); // 10
```

Anatomia da String

C	L	A	U	D	I	O
0	1	2	3	4	5	6

- .length (size)
- .toUpperCase()
- .replace()

```
> console.log("Transformando dados..._");
```

Operadores Aritméticos

Symbol (Fira Code)	Name (Inter)	Example (Fira Code)
+	Adição	`10 + 5 = 15`
-	Subtração	`10 - 5 = 5`
*	Multiplicação	`10 * 5 = 50`
/	Divisão	`10 / 2 = 5`
%	Módulo (Resto da Divisão)	`23 % 2 = 1`
++	Incremento	`var x=5; ++x; // x=6`
--	Decremento	`var y=5; --y; // y=4`

⚠ Cuidado com a concatenação: **'10' + 10 = '1010'**, não 20.

O Grande 'Gotcha': Comparação

Comparação Insegura (==)

10 == "10"

TRUE

Compara apenas o valor (Ignora o tipo).

```html
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript</title>
    <meta charset="UTF-8">
</head>
    <body>
        <script>

        var num="10";

        if (num == 10) {
            console.log("true"); // true
        }
        </script>
    </body>
</html>
```

Comparação Segura (===)

10 === "10"

FALSE

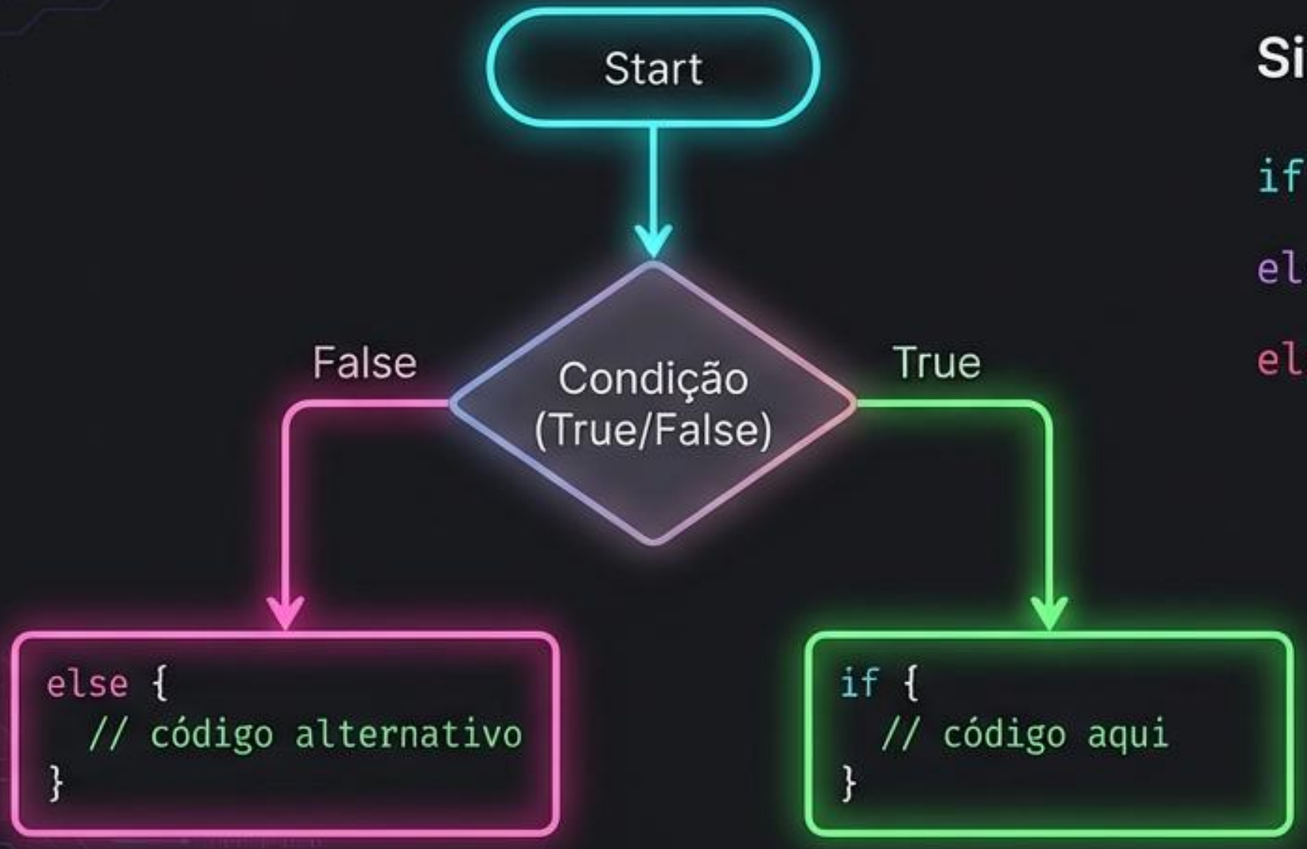
Compara valor E tipo (Seguro).

```html
<!DOCTYPE html>
<html>
<head>
    <title>JavaScript</title>
    <meta charset="UTF-8">
</head>
    <body>
        <script>
        var num="10";

        if (num === 16) {
            console.log("true");
        } else {
            console.log("false"); // false
        }
    </script>
</html>
```

Lógica Booleana: & & (E), || (OU), ! (NÃO)

Tomada de Decisão



Sintaxe

```
if (condição) { … }
else if (outra) { … }
else { … }
```

Alternativas

Switch

Operador Ternário

```
switch (opção) {
  case 1:
    …
    break;
  default:
    …
}
```

Automação com Loops

Não se repita. Deixe o código trabalhar.

For

```
for (let i = 0; i < 5; i++)
```

Quando sei quantas vezes

While

```
while (condição)
```

Enquanto for verdade

Do...While

```
do {
   ...
} while (condição)
```

Executa pelo menos uma vez

Arrays: Listas Ordenadas

```
var frutas = ["Banana", "Maçã", "Uva"];
```

Toolbox

- **push()** adicionar
- **pop()** remover
- **filter()** filtrar

O índice sempre começa do zero.

Estruturas e Ações

Funções (A Receita)

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

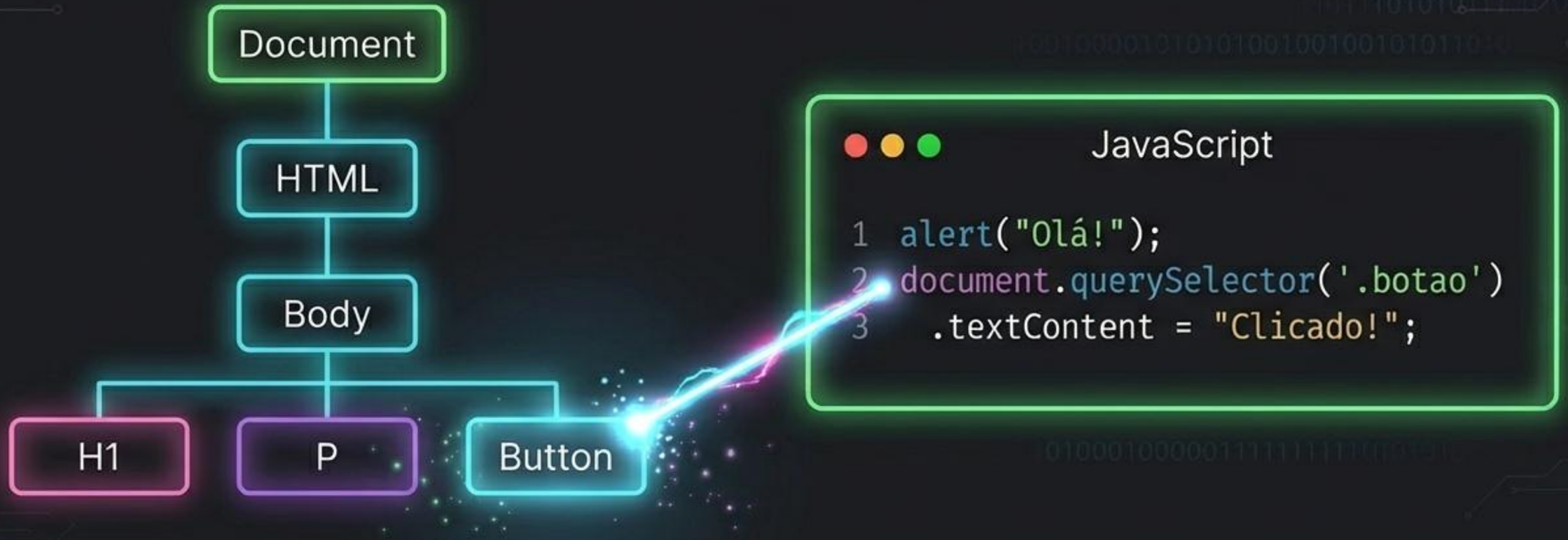
Arrow Function:
const somar = (a, b) => a + b;

Objetos (O Modelo)

```
{
  nome: "Claudio",
  idade: 30
}
```

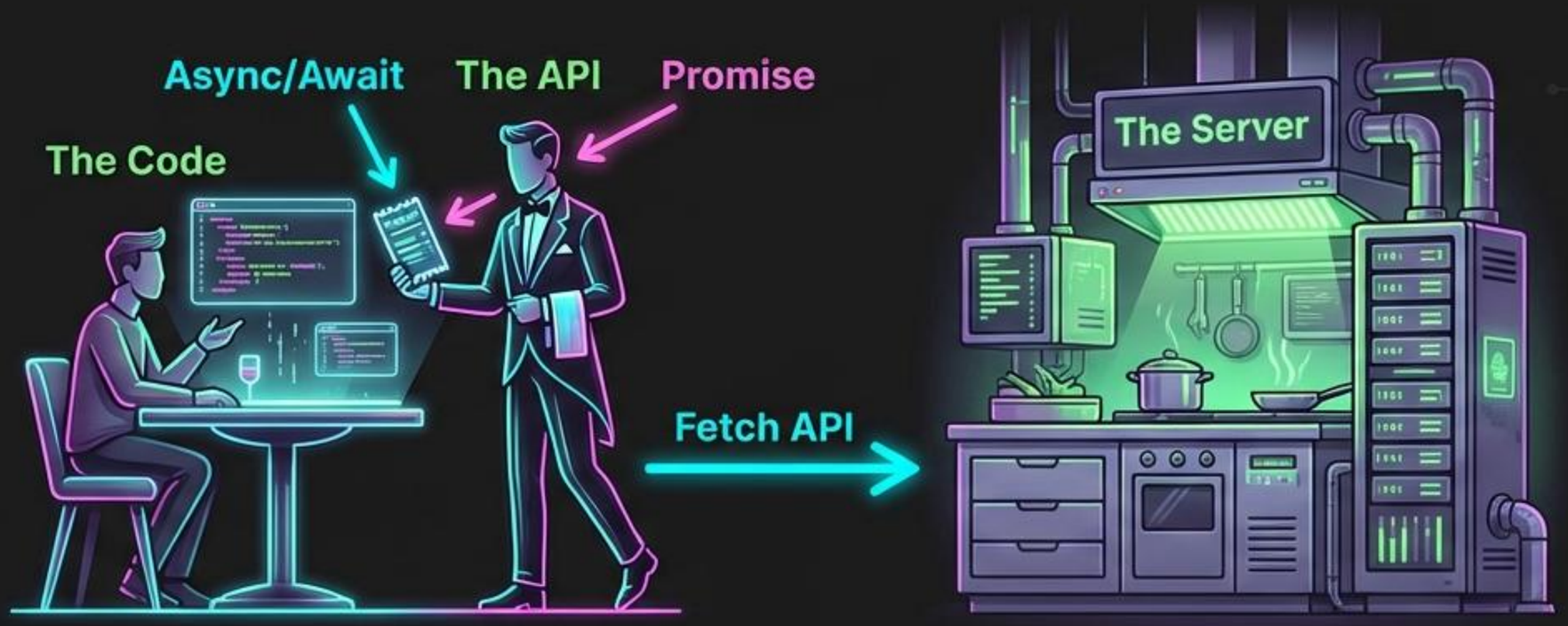
Agrupando propriedades (dados) e métodos (ações).

O DOM: Manipulando a Página



O JavaScript enxerga o HTML como uma árvore de elementos vivos.

Assincronismo e o Mundo Externo




```
async function buscarDados() {
  const resposta = await fetch(url);
}
```

Boas Práticas e Próximos Passos

- Use 'const' e 'let' (evite 'var')
- Código limpo e comentado
- Organize em arquivos separados
- Pratique todos os dias

“O código só funciona se você escrever.”