

Introdução a JavaScript

Programação para Internet I
IFB - Tecnologia em Sistemas para Internet

Marx Gomes van der Linden

JavaScript

- Linguagem de programação embutida em todos os navegadores modernos
- Programação para web primariamente no **cliente**
- Um dos 3 componentes básicos do desenvolvimento web em frontend



Atenção:

JavaScript \neq Java

JavaScript no navegador

- Controla a aparência e conteúdo do documento
- Verifica dados de formulários HTML
- Recebe dados do servidor depois que a página já está carregada (Ajax)
- Animações e jogos

Versões do JavaScript

- 1995: Primeira versão (Netscape)
 - » Mocha → LiveScript → JavaScript
- 1996: Implementação da Microsoft (JScript)
- 1997-1999: ECMAScript 1, 2 e 3
- 2009: ECMAScript 5
- 2011: ECMAScript 5.1
- 2015: ECMAScript 2015
- ECMAScript 2016, 2017, 2018...

node.js



- Ambiente de execução de JavaScript independente do navegador
- Baseado no engine V8 (Google Chrome)
- Usado na linha de comando ou como servidor web
- Inclui o gerenciamento de pacotes **npm**

Hello, World!

```
console.log('Olá, Mundo!');
```

- Executando no terminal:

```
> node arquivo.js
```

```
Olá, Mundo!
```

Executando no navegador (html)

```
<script>  
    console.log('Olá, Mundo!');  
</script>
```

- Tecle F12 para ver o resultado

Executando no navegador - alternativa

```
<script>  
    document.write('Olá, Mundo!');  
</script>
```

- O resultado aparece no próprio documento

Características da linguagem

- Interpretada (não é compilada)
- Família sintática: C, C++, C#, Java, PHP...
- Tipos de dados **implícitos** e **dinâmicos**
- Caractere ";" opcional no final da cada instrução
 - » Mas altamente recomendado
- Multiparadigma: Estruturada, funcional, orientada a objetos

Tipos de variáveis

- Número: **12**, **13.5**
- String: **"aspas duplas"**, **'aspas simples'**, **`aspas especiais`** (ES2015)
- Booleano: **true**, **false**
- **null**, **undefined**
- Objetos e símbolos

Variáveis

- Declaração opcional:
 - » `let` (preferencial - ES2015)
 - » `var`
- Tipos implícitos
- Conversão automática

Operadores

- Aritméticos: `-` `+` `*` `/` `%`
- Concatenação: `+`
- Incremento e decremento: `+=` `++` `-=` `--`
- Comparação: `<` `<=` `>` `>=` `==` `!=` `===` `!==`
- Operadores lógicos: `&&` `||` `!`
- Operador ternário: `?` `:`
- Detecção de tipo: `typeof`

Qual é o resultado?

```
let n = 1;  
  
let txt1 = "teste" + n + 1;  
let txt2 = "teste" + (n + 1);  
  
console.log(txt1);  
console.log(txt2);
```

```
teste11  
teste2
```

Comentários

```
/*  
    Comentário de múltiplas linhas  
    Código de demonstração do operador '+'  
*/  
let n = 1;  
  
// Comentário de 1 linha  
let txt1 = "teste" + n + 1;      // Concatenação  
let txt2 = "teste" + (n + 1);    // Soma  
  
console.log(txt1);  
console.log(txt2);
```

Interpolação de strings

- Novidade no ES2015
 - » Não suportado no Internet Explorer
- Utiliza o caractere: `string`
 - » Não confundir com: 'string'

```
let a = 5;  
let b = 15;  
console.log(`A soma de ${a} com ${b} é  
igual a ${a + b}`);
```

A soma de 5 com 15 é igual a 20

Parâmetros da linha de comando

– No Node.js, utilize: `process.argv[n]`

» Onde `n`:

› `0` → Executável do Node

› `1` → Arquivo do script

› `2` → Parâmetro 1

› `3` → Parâmetro 2

› ...

```
> node teste.js param1 param2 param3
```

0

1

2

3

4

Constantes (ES2015)

- Parecidas com variáveis, mas o valor não pode ser modificado

```
const NUMERO_MAGICO = 300;  
  
console.log(NUMERO_MAGICO);  
  
NUMERO_MAGICO = 350; ← Erro!
```

Operadores de incremento

```
let num = 1;
```

```
num += 5;  
console.log(num);
```

```
num *= 3;  
console.log(num);
```

Pré-incremento, pós-incremento

```
let num = 1;  
  
console.log('Pré-incremento:', ++num);  
console.log('Novo valor:', num);  
  
console.log();  
  
num = 1;  
console.log('Pós-incremento:', num++);  
console.log('Novo valor:', num);
```

Qual é o resultado?

```
let a = 2;  
let b = a++ + a-- + ++a + --a;  
console.log(b);  
console.log(a);
```

10

2

Hoisting

- Em JavaScript, declarações de variáveis com **var** são movidas para o topo do script!
 - » Mas não a inicialização
- Pode resultar em comportamento inesperado

Hoisting

```
console.log(x);
```

```
console.log(x);  
var x = 5;  
console.log(x);
```

```
console.log(x);  
let x = 5;  
console.log(x);
```

Instrução if-else

- Sintaxe similar a outras linguagens:

```
let x = 3;  
  
if(x == 3)  
    console.log("x é igual a três");  
else  
    console.log("x é diferente de três");
```


Instrução if-else

- Caracteres {}

```
let x = 3;
let foiigual;

if(x == 3) {
    foiigual = true;
    console.log("x é igual a três");
}
else{
    foiigual = false;
    console.log("x é diferente de três");
}
```

Operador ternário

- Sintaxe:
condição ? expressão-if : expressão-else
- A expressão completa retorna um valor
- Equivalente a uma instrução if-else

```
let preco = temDesconto? 15.00 : 20.00;
```

Instrução switch-case

- Equivalente a vários **if-else** aninhados
- Normalmente se usa **break** após cada caso

```
let acao = 'embalar';

switch (acao) {
  case 'preparar':
    console.log("Preparar");

  case 'embalar':
    console.log("Embalar");

  case 'enviar':
    console.log("Enviar");
    break;

  default:
    console.log("Erro!");
}
```

Estruturas de loop

- Permitem repetir código de maneira estruturada
 - » while
 - » do ... while

```
let i = 1;  
while (i < 5) {  
  console.log(i);  
  i++;  
}
```

```
while (not edge) {  
  run();  
}
```

```
do {  
  run();  
} while (not edge);
```



Loop for

```
let i = 1;  
while (i < 5) {  
  console.log(i);  
  i++;  
}
```



```
for(let i=1; i < 5; i++) {  
  console.log(i);  
}
```

Verdadeiro e falso em JavaScript

- São valores falsos:
 - » `false`
 - » `""` ou `' '` (string vazia)
 - » `0`
 - » `NaN`
 - » `undefined`
 - » `null`
- Todos os outros valores são verdadeiros