Curso Web

A Linguagem JavaScript (Parte I)

Softblue
cursos online

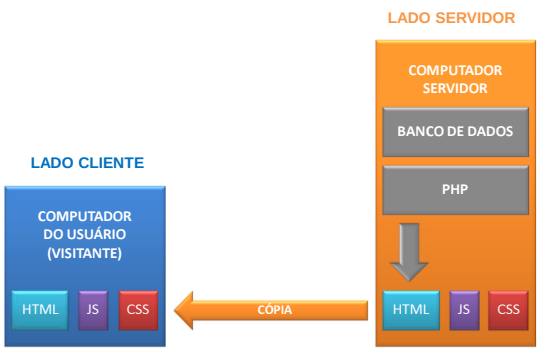
Hello JavaScript!

Softblue

- O que é?
  - Linguagem de programação
  - Criada em 1995 pela Netscape
  - Criar páginas dinâmicas
- Como funciona
  - Executa no lado cliente

Cliente vs. Servidor

Softblue



Continuando...

- O que é?
  - Linguagem de programação
  - Criada em 1995 pela Netscape
  - Criar páginas dinâmicas
- Como funciona
  - Executa no lado cliente
  - Códigos desprotegidos
- JavaScript vs. Java
  - Java: compilada, robusta, software
  - JavaScript: script, navegador web
- Pré-requisito
  - Lógica de Programação (gratuito)

Hello JavaScript!

- Marcação <script>

```
<script>

 /*
   Bloco de código JavaScript comentado...
 */

 // Linha comentada...

</script>
```

- /* */: Comentário de bloco
- //: Comentário de linha

Tipos de dados

- String: textos e caracteres

```
var alfa = "Curso Web";
```

- Número: números inteiros e decimais

```
var beta = 50.03;
```

- Booleano: verdadeiro ou falso

```
var gama = true;
```

- Objeto: qualquer tipo de objeto

```
var delta = new array("Maçã", "Laranja", "Uva");
```

Propriedades de textos

- length: tamanho da string
- indexOf: posição inicial de uma busca
- substring(início, fim): trecho de string

```
var texto = "Softblue!!";
document.write("O texto tem " + texto.length + " posições.<br>");          // 10

var posicao = texto.indexOf("f");
document.write("Resultado da busca 'f': " + posicao + "<br>");              // 2

posicao = texto.indexOf("w");
document.write("Resultado da busca 'w': " + posicao + "<br>"); // -1

var trecho = texto.substring(4, 7);
document.write("Trecho entre posições 4 e 7: " + trecho);                   // blu
```

S	o	f	t	b	l	u	e	!	!
0	1	2	3	4	5	6	7	8	9

Arrays

- Arrays
  - Coleções de objetos
  - Array é um objeto

```
var lista = new Array("Maçã", "Laranja", "Abacaxi");

document.write(lista[0]);          // Maçã
document.write(lista[1]);          // Laranja
document.write(lista[2]);          // Abacaxi
```

Maçã	Laranja	Abacaxi
0	1	2

Operadores e comandos

- Operadores matemáticos
- Operadores condicionais
- Operadores lógicos
- Comandos de controle
- Comandos de tomada de decisão
- Comandos de repetição

Operadores matemáticos

= Atribui o valor da sua direita à variável a sua esquerda

```
var x = 3;                    // Resultado: x armazena o valor 3
```

+ Concatena dois valores de formato texto (string)

```
var x = "Soft" + "blue";      // Resultado: x armazena o valor "Softblue"
```

+ Soma dois valores numéricos

```
var x = 3 + 5;                // Resultado: x armazena o valor 8
```

- Subtrai dois valores numéricos

```
var x = 3 - 5;                // Resultado: x armazena o valor -2
```

* Multiplica dois valores numéricos

```
var x = 2 * 6;                // Resultado: x armazena o valor 12
```

/ Divide dois valores numéricos

```
var x = 18 / 3;               // Resultado: x armazena o valor 6
```

% Obtém o resto da divisão entre dois valores numéricos

```
var x = 19 % 3;               // Resultado: x armazena o valor 1
```

Operadores matemáticos

++ Incrementa em 1 o valor da variável acoplada

```
var x = 3;
x++;                          // Resultado: x neste momento armazena o valor 4
```

-- Decrementa em 1 o valor da variável acoplada

```
var x = 3;
x--;                          // Resultado: $x neste momento armazena o valor 2
```

- É possível utilizar estes operadores em expressões e outros comandos

```
var x = 3;
var y = 2 + x++;              // Resultado: y = 5 e x = 4
```

- Diferença entre ++$x e $x++

```
var x = 3;
var y = 2 + ++x;              // Resultado: y = 6 e x = 4
```

Operadores matemáticos

Assumir var x = 5 para o início de cada exemplo apresentado neste slide

+= Soma à variável à sua esquerda o valor apresentado em sua direita

```
x += 3;       // Resultado: x armazena o valor 8
```

-= Subtrai da variável à sua esquerda o valor apresentado em sua direita

```
x -= 3;       // Resultado: x armazena o valor 2
```

*= Multiplica a variável à sua esquerda pelo valor apresentado em sua direita

```
x *= 3;       // Resultado: x armazena o valor 15
```

/= Divide à variável à sua esquerda o valor apresentado em sua direita

```
x /= 3;       // Resultado: x armazena o valor 1.6666666666667
```

%= Atribui à variável da esquerda o resto de sua divisão pelo valor apresentado em sua direita

```
x %= 3;       // Resultado: x armazena o valor 2
```

Operadores relacionais

- Utilizados para avaliar uma condição

Operador	Significado
==	Compara se dois valores tem o mesmo valor, mesmo que sejam de tipos de dados diferentes
!=	Compara se dois valores são diferentes, mesmo que sejam de tipos de dados diferentes
===	Compara se dois valores tem o mesmo valor, e se são do mesmo tipo de dado
!==	Compara se dois valores são diferentes, e se não são do mesmo tipo de dado
<	Compara se o valor da esquerda é menor que o da direita
>	Compara se o valor da esquerda é maior que o da direita
<=	Compara se o valor da esquerda é menor ou igual que o da direita
>=	Compara se o valor da esquerda é maior ou igual que o da direita

Operadores lógicos

- Utilizados para montar uma condição

Operador	Significado
!	Nega a condição informada (inverte seu resultado lógico)
&	Retorna verdadeiro se ambas as condições da esquerda e direita forem satisfeitas
|	Retorna verdadeiro se pelo menos uma das condições (da direita ou da esquerda) for satisfeita
&&	O mesmo que &, porém é otimizado: Se a condição do lado esquerdo for falsa, não verifica o lado direito da expressão. Resulta a operação toda em FALSA neste caso.
||	O mesmo que |, porém é otimizado: Se a condição do lado esquerdo for verdadeira, não verifica o lado direito da expressão. Resulta a operação toda em VERDADEIRA neste caso.

Comandos de decisão

- if(condição) { } [else if(condição) { }] [else { }]
  - Permite executar um bloco de código se determinada condição for verdadeira

```
if(3 > 5) {
	// Não entra no primeiro if.
}

if(1 < 10) {
	// Entra no segundo if.
}
else {
	// Não entra no else do segundo if.
}
```

```
var i = 5;
if(i == 3)
{
	// O valor de i é 3.
}
else if(i == 4)
{
	// O valor de i é 4.
}
else {
	// O valor de i não é 3 nem 4.
}
```

Comandos de decisão

- switch(valor) { cases }
  - Permite executar um bloco de código específico dependendo do valor avaliado

```
var i = 1;

switch(i)
{
	case 0:
		// O valor de i é 0
		break;
	case 1:
		// O valor de i é 1
		break;
	case 2:
		// O valor de i é 2
		break;
	default:
		// Nenhum
		break;
}
```

Comandos de repetição

- for(inicialização; condição; incremento)
  - Permite criar um laço de repetição, executando para cada valor o mesmo bloco de código

```
for(i = 0; i < 10; i++)
{
	document.write(i + ' ');
}

// Resultado: 0 1 2 3 4 5 6 7 8 9
```

Comandos de repetição

- for(elemento in array)
  - Versão alternativa para navegação em arrays

```
var lista = Array("Maçã", "Uva", "Laranja");

for(fruta in lista)
{
	document.write(fruta + ' ');
}
// Resultado: 0 1 2

for(fruta in lista)
{
	document.write(lista[fruta] + ' ');
}
// Resultado: Maçã Uva Laranja
```

Comandos de repetição

- while(condição)
  - Permite executar várias vezes um bloco de código enquanto sua condição for verdadeira

```
var i = 0;

while(i < 5)
{
	document.write(i + ' ');
	i++;
}

// Resultado: 0 1 2 3 4
```

Comandos de repetição

- do { } while (condição);
  - Similar ao while, com a diferença de que o do while tem o bloco de código executado obrigatoriamente uma vez antes de avaliar a condição

```
var i = 0;

do
{
	document.write(i + ' ');
	i++;
}
while(i < 5);

// Resultado: 0 1 2 3 4
```

Comandos de controle

- break
  - Permite interromper suspender o comando de repetição e ir para o próximo comando

```
for(i = 0; i < 10; i++)
{
	if(i == 4)
	{
		break;
	}

	document.write(i + ' ');
}

// Resultado: 0 1 2 3
```

Comandos de controle

- continue
  - Permite instruir ao comando de repetição que interrompa esta iteração e avançe para a próxima iteração da repetição

```
for(i = 0; i < 10; i++)
{
	if(i == 4)
	{
		continue;
	}

	document.write(i + ' ');
}

// Resultado: 0 1 2 3 5 6 7 8 9
```

Criando funções

- Permite centralizar blocos de código

Sintaxe

```
function nomeDaFunção([parâmetros]) { }
```

Exemplo

```
function calcularDobro(num)
{
	var dobro = num * 2;
	return dobro;
}

var i = 5;
var iDobro = calcularDobro(i);
document.write("O dobro de " + i + " é " + iDobro);

// Resultado: O dobro de 5 é 10
```

Aulas práticas e manuais on-line

Assista agora as aulas práticas, que apresentam o uso dos comandos abordados nesta aula teórica.

Clique aqui para visualizar as aulas práticas disponíveis