JavaScript
- Conceitos Básicos -

Projeto de
Construção de Sites

Introdução

- O *JavaScript* é uma linguagem de programação simples criada para dar mais interatividade e maior funcionalidade às páginas Web;
- Roda na máquina cliente, ou seja, é interpretado pelo navegador;
- *JavaScript* não é Java;
- Uma boa execução do *JavaScript* depende do suporte fornecido pelo navegador à versão da linguagem utilizada;

Meu Primeiro Script

```
<html>
    <head>
        <title>A Minha Página com JavaScript</title>
        <script type="text/javascript">
            alert("Seja bem vindo(a) à minha página!");
        </script>
    </head>
    <body>
        Aqui colocamos o conteúdo da página em HTML
    </body>
</html>
```

Mais Dinamismo

O JavaScript permite atribuir mais dinamismo às suas páginas utilizando, por exemplo, o tratamento de eventos.

Exemplo

```
<html>
    <body>
        <a href="http://www.artifice.web.pt/" target="_blank"
        onclick="alert('Obrigado por visitar o Artífice da Web!')">
        visite o Artífice da Web</a>
    </body>
</html>
```

Localização do Código JS

- **Solto no conteúdo HTML**: executados na seqüência em que aparecem na página;
- **Atrelado a algum evento**: executados apenas quando o evento ocorre.

Localização do Código JS

```
<html>
        <script type="text/javascript">
          alert("Seja bem vindo(a) à minha página!");
        </script>

    <body>
        <a href="http://www.artifice.web.pt/" target="_blank"
        onclick="alert('Obrigado por visitar o Artífice da
Web!')">    visite o Artífice da Web</a>
    </body>
</html>
```

Solto na página

Atrelado a evento

Comentários

- Os comentários permitem-nos descrever o código *JavaScript* que produzimos tornando-o mais legível e mais fácil de manter.
- Comentário de linha: //

```
// esta linha está comentada
```

- Comentário de bloco: /* */

```
/* este bloco está comentado */
```

Bloco de Código

- Os blocos de código são delimitados por { }.
- Exemplo:

```
If (a == b){          *** Início do bloco ***
   a = b + c;
   b++;
}                     *** Fim do bloco ***
```

Variáveis

Introdução

- As variáveis são posições de memória que servem para guardar informação;
- Elas nos permitem dar nomes a cada um dos fragmentos de informação com que temos de lidar;
- Para evitar erros e aumentar a produtividade é importante escolher nomes que descrevam o conteúdo que cada variável armazena.

Nomes de Variáveis

1. Todos os nomes têm de começar com uma letra ou com o caracter _
2. Os caracteres restantes podem ser números.

*Nunca se esqueça que para o JavaScript é **Case-Sensitive**, ou seja, faz distinção entre letras maiúsculas e minúsculas.*

Declaração de Variáveis

- Declaração é o ato de criar uma variável;
- Variáveis podem ser
  - **Globais:** podem ser utilizadas em qualquer parte do script;
  - **Locais:** podem ser utilizadas apenas dentro da função em que foi declarada.
    - Para declarar uma variável local deve-se utilizar a palavra reservada **var**.

Exemplos

```
dividendo = 12;          // declaração de variável global
var divisor = 3;         // declaração de variável local
sabor = "Doce";
var pi = 3.14159;
```

Tipos de Variáveis

- O JavaScript é capaz de reconhecer três tipos de dados:
  - Números, como por exemplo 12 ou 3.14159
  - Texto (variáveis de tipo String), como por exemplo: "Seja Bem Vindo(a)!“
  - Valores lógicos (true ou false)

null é uma palavra reservada que significa que a variável não guarda qualquer valor

Conversões de Tipos

- Em JavaScript não é necessário definir o tipo de uma variável;
- O próprio interpretador define o tipo da mesma ao longo da execução do código;
- Deve-se, no entanto, tomar o devido cuidado ao utilizar tal característica da linguagem para que o código não fique difícil de entender e, portanto, de ser atualizado.

Expressões Literais

Introdução

- As expressões literais representam valores fixos;
- São escritas diretamente pelo programador ao produzir o script;
- Exemplos:
  - 123
  - "Isto é uma expressão literal"

Exemplo

```
<html>
  <body>
  <script type="text/javascript">
  <!--
  var nome = "visitante";
  var hora = 11;
  if(hora < 12)
    document.write("Bom dia. Seja bem vindo senhor " + nome);
  else {
    if(hora >= 13)
      document.write("Boa tarde. Seja bem vindo senhor " + nome);
    else
      document.write("Seja bem vindo! Almoça conosco?");
   }
   -->
   </script>
   </body>
</html>
```

Números Inteiros

- Podem ser expressos:
  - Na forma decimal (base 10)
  - Na forma hexadecimal (base 16)
  - Na forma octal (base 8)

```
var i = 42;        // decimal
var j = 052;       // octal. Inicia com 0
var k = 0X2A;      // hexadecimal
```

Exemplo

```
<html>
<body>
    <script type="text/javascript">
    <!--
     var i = 42;          // decimal
     var j = 052;         // octal
     var k = 0X2A;        // hexadecimal

     document.write("i = " + i);
     document.write("<br/>");
     document.write("j = " + j);
     document.write("<br/>");
     document.write("k = " + k);
     -->
     </script>
</body>
</html>
```

Número com ponto flutuante

- Uma expressão literal com ponto flutuante representa um número que não é inteiro mas que contém uma parte inteira e uma parte fracionária.
- A representação que a máquina constrói para estes números baseia-se na notação científica
- O número -7645.4532 é igual a -7.64532 multiplicado por 10 elevado a 3 e escreve-se como -7.6454532E3, em que E3 representa 10 elevado a 3

Usando *Array*

Declaração e Indexação

```
var frutas_tropicais = new Array("Goiaba",
   "Manga", "Maracujá");
var frutas_nacionais = new Array(3);
frutas_nacionais[0] = "Maçã";
frutas_nacionais[1] = "Cereja";
frutas_nacionais[2] = "Laranja";
```

Exemplo

```
<html>
<body>
    <script type="text/javascript">
    <!--
    var sortido = new Array(8975, "Livro", false, -27.765, "Bolachas");
    document.write("sortido = " + sortido);

    sortido[0] = 0.0004763;
    sortido[2] = true;
    sortido[6] = "Caderno";

    document.write("<br/>");
    document.write("sortido = " + sortido);
    -->
    </script>
</body>
</html>
```

Usando Array

- Um *array* é um objeto, logo, possui atributos e métodos.
- Um dos atributos do *array* informa o comprimento. Tal atributo é o **length**;

```
var numeroElementos = sortido.length;
```

Operadores

Operadores de Atribuição

Operador	Exemplo	É O Mesmo Que
=	x=y	x=y
+=	x+=y	x=x+y
-=	x-=y	x=x-y
*=	x*=y	x=x*y
/=	x/=y	x=x/y
%=	x%=y	x=x%y

Exemplo

```
<html><body>
<script type="text/javascript">
<!--
var s = "Isto vai ser bonito... " // uma string
var z = s + "Ah pois vai!"
var p = q = 2                                    // dois números inteiros
document.write("s = \""+s+"\"")
document.write("<br/>")
document.write("z = \""+z+"\"")
document.write("<br/>")
document.write("<br/>")
document.write("p = "+p)
document.write("<br/>")
document.write("q = "+q)
// -->
</script>
</body></html>
```

Operadores de Comparação

Operador	Descrição	Exemplo
Igualdade (==)	Verifica se os dois operandos são iguais	x==y dá true se x igualar y
Desigualdade (!=)	Verifica se os operandos são desiguais	x!=y dá true se x não for igual a y
Maior do que (>)	Verifica se o operando da esquerda é maior do que o da direita	x>y dá true se x for maior do que y
Maior ou igual (>=)	Verifica se o operando da esquerda é maior ou igual ao da direita	x>=y dá true se x for maior ou igual a y
Menor do que (<)	Verifica se o operando da esquerda é menor do que o da direita	x<y dá true se x for menor do que y
Menor ou igual (<=)	verifica se o operando da esquerda é menor ou igual ao da direita	x<=y dá true se x for menor ou igual a y

Exemplo

```
<html><body>
<script type="text/javascript">
<!--
var b = (5 > 2)
document.write("b = " + b + " (porque 5 > 2 é uma afirmação verdadeira)")
document.write("<br/>")
b = (5 != 2)
document.write("b = " + b + " (porque 5 != 2 é uma afirmação verdadeira)")
document.write("<br/>")
b = (5 == 2)
document.write("b = " + b + " (porque 5 == 2 é uma afirmação falsa)")
document.write("<br/>")
b = (5 >= 2)
document.write("b = " + b + " (porque 5 >= 2 é uma afirmação verdadeira)")
document.write("<br/>")
// -->
</script>
</body></html>
```

Operadores Aritméticos

Operador	Descrição	Exemplo	Resultado
+	Adição	2+2	4
-	Subtracção	5-2	3
*	Multiplicação	4*5	20
/	Divisão	15/5 5/2	3 2.5
%	Modulus (resto da divisão)	5%2 10%8 10%2	1 2 0
++	Incrementar (aumentar uma unidade)	x=5 x++	 x=6
--	Decrementar (diminuir uma unidade)	x=5 x--	 x=4

Exemplo

```
<html><body>
<script type="text/javascript">
<!--
var p = 5,n = 2
var r = p % n
document.write("O resto da divisão de 5 por dois é " + r)
document.write("<br/>")
document.write("p = " + p)
p *= 3
document.write("<br/>Mas agora: p = " + p + “<br/>")
p /= 5
document.write("e agora: p = " + p + ", porque o seu valor foi dividido por 5<br/>")
p += 4
document.write("e agora: p = " + p + ", porque ao seu valor somámos 4
    unidades<br/>")
p -= 3
document.write("e agora: p = " + p + ", porque ao seu valor subtraímos 3 unidades")
// -->
</script> </body></html>
```

Operadores Lógicos

Operador	Descrição	Exemplo
&&	e (and)	x=6 y=3 (x < 10 && y > 1) dá true
||	ou (or)	x=6 y=3 (x==4 || y==4) dá false
!	negação (not)	x=6 y=3 !(x==y) dá true

Exemplo

```
<html><body>
<script type="text/javascript">
<!--
var b = (5 > 2)
var v = (2 > 5)
document.write("b = " + b + " (porque 5 > 2 é uma afirmação verdadeira)“ + “<br>”)
document.write("v = " + v + " (porque 2 > 5 é uma afirmação falsa)“ + “<br>”)
document.write("!b = " + !b + " (porque a negação de true é false)“ + “<br>”)
document.write("!v = " + !v + " (porque a negação de false é  true)“ + “<br>”)
var c = b && v
document.write("c = " + c + " (porque a operação: true && false dá false)“ + “<br>”)
c = b || v
document.write("c = " + c + " (porque a operação: true || false dá true)“ + “<br>”)
b = (3 < 10 && 2 > 1) || !c
document.write("b = " + b + " (porque a operação anterior dá true)")
// -->
</script>
</body></html>
```

Precedência de Operadores

1. () [] . Parêntesis, colchetes e o operador ponto que serve para os objetos
2. ! - ++ -- negação, negativo e incrementos
3. * / % Multiplicação, divisão e módulo
4. + - Soma e diferença
5. << >> >>> Mudanças a nível de bit
6. < <= > >= Operadores condicionais
7. == != Operadores condicionais de igualdade e desigualdade
8. & ^ | Lógicos em nível de bit
9. && || Lógicos booleanos
10. = += -= *= /= %= <<= >>= >>>= &= ^= != Atribuição

"Sempre avaliando a expressão da esquerda para a direita"

Exercícios de Operadores e Variáveis

Estruturas Condicionais

A instrução IF

- A instrução if é utilizada para testar uma condição e executar um bloco de código apenas quando ela é satisfeita;

```
if (condição) {
        código a executar se a
        condição for verdadeira
}
```

A instrução IF...ELSE

- A instrução **if ... else** é utilizada para testar uma condição. Se a condição for satisfeita será executado um bloco correspondente ao **if**, caso contrário, é executado o bloco correspondente ao **else**;

```
if (condição) {
        código a executar se a condição for
        verdadeira
} else {
        código a executar se
        a condição for falsa
}
```

Exemplo

```
<html><body>
<script type="text/javascript">
<!--
var d = new Date();
var hora = d.getHours();
if (hora < 12)
    document.write("<h1>Bom dia</h1>");
else{
    if(hora > 18)
        document.write("<h1>Boa noite</h1>");
    else
        document.write("<h1>Boa tarde</h1>");
}
-->
</script>
</body> </html>
```

A instrução SWITCH

- Esta instrução é utilizada para comparar o valor da sua expressão com vários valores. Para cada caso em que houver uma igualdade será executada uma determinada porção de código;

```
switch (expressão) {
        case label1:
                código a executar se expressão = label1
                break
        case label2:
                código a executar se expressão = label2
                break
        default: código a executar se a expressão não for igual
                 a nenhuma das alternativas apresentadas antes
}
```

Exemplo

```
<html><body>
<script type="text/javascript">
<!--
var d = new Date()
var dia = d.getDay()
switch (dia){
        case 5: document.write("Finalmente é sexta!")
                        break
        case 6: document.write("Hoje é sábado. Fixe!")
                        break
        case 0: document.write("Hoje é domingo")
                        break
        default: document.write("Que aflição, ainda falta tanto
    para o fim de semana.")
}
// -->
</script>
</body></html>
```

Estruturas de Repetição

A instrução FOR

- Instrução utilizada quando se sabe o número de vezes que o laço deve executar.

```
for (inicialização; condição; atualização){
      bloco de código a executar
}
```

Exemplo

```
<html>
<body>
<script type="text/javascript">
<!--
for (var i = 1; i <= 6; i++){
    document.write("<h" + i + ">Este é um cabeçalho de
    nível " + i)
    document.write("</h" + i + ">")
}
// -->
</script>
</body></html>
```

A instrução WHILE

- É utilizada quando o laço pode ser interrompido a qualquer momento, ou seja, não se conhecia, de ante-mão, o número de iterações.
- A instrução while repete a execução de um bloco de código enquanto uma condição for satisfeita;

```
while (condição){
        código a executar
}
```

Exemplo

```
<html>
<body>
<script type="text/javascript">
<!--
var i = 0
while (i != 5){
    document.write("O número é " + i)
    document.write("<br/>")
    i = window.prompt(“Digite o valor de i:”);
}
document.write("O ciclo terminou<br/>")
// -->
</script>
</body></html>
```

Exercícios de Estruturas Condicionais e Estruturas de Repetição

Exercícios

1. Crie um código em *JavaScript* que exiba na página HTML uma seqüência de números de 1 a 100. Os números pares devem ser formatados em negrito e os números ímpares em itálico;
2. Crie um código em *JavaScript* que dado um array de inteiros e um número qualquer, verifica se este número existe no array. Se existir exibe “O número foi encontrado”, caso contrário exibe “Não foi possível encontrar o número”.

Trabalhando com Funções

Construção de Função

- Função é um bloco de código a que damos um nome para que possa ser chamado várias vezes em locais diferentes;
- Tem-se, então, duas etapas:
  - **Declaração da função**
  - **Chamada da função**

Declaração de função

- É dito ao interpretador *JavaScript* que uma função está sendo criada;
- É dito quais os parâmetros serão passados na chamada da função;
- É dito qual o código a ser executado pela função, incluindo a operação *return,* caso necessário.

```
function multiplicar(p, q) {
        var m = p*q
        return m
}
```

Chamada de função

- É dito qual a função a ser chamada;
- São informados os valores dos parâmetros da função, os quais foram definidos na sua declaração;
- É tratado o valor de retorno da função.

```
resultado = multiplicar(521, 2)
```

Minha Primeira Função

Crie uma função que calcula e retorna a média entre dois números passados como parâmetro

Exercícios

1. Crie uma função que tem o objetivo de formatar um texto passado como parâmetro com a fonte Verdana e cor azul;
2. Crie uma função que retorna o maior entre três números;
3. Crie uma função que dada uma seqüência de números em um array retorna ***true*** se a seqüência estiver em ordem crescente e ***false***, caso contrário.

JavaScript
- Conceitos Básicos -

Projeto de
Construção de Sites

CEFETES
Centro Federal de Educação
Tecnológica do Espírito Santo