TICs

B

Linguagem do lado cliente

Unidade B
Linguagem de Programação Web

Sistema Universidade Aberta do Brasil - UAB | IF Sul-rio-grandense

UNIDADE B

LINGUAGEM DO LADO CLIENTE

Introdução

Esta unidade tem por objetivo estudar uma linguagem do lado cliente, que permite incorporar às páginas (X)HTML a possibilidade de interação com o usuário, ou seja, vai proporcionar o comportamento da página.

Na unidade anterior vimos que a (X)HTML possibilita, por meio de marcações, a estruturação do documento com seu conteúdo. Porém, não conseguimos introduzir lógica em um documento usando apenas (X)HTML, por exemplo:

- se acontecer isso, então faça aquilo...
- se o usuário interagir com a página desta forma, então faça aquilo...
- se o usuário está utilizando o navegador tal, então mostre tal objeto...

Por isso, precisamos conhecer a linguagem JavaScript para que possamos introduzir comportamento aos documentos que vamos construir. Assim, a seguir abordaremos a sintaxe básica da linguagem, como capturar eventos, criação de funções, interação com formulários (X)HTML, manipulação de janelas, entre outros. Então, mãos a obra...

Introdução a linguagem JavaScript

Histórico e características

JavaScript é uma linguagem criada especificamente para a Web, por isso, é leve e interpretada pelo navegador. Como já comentamos no nosso documento introdutório, ela é uma linguagem do lado cliente, e possibilita a interação com o usuário, uma vez que é possível controlar a página e seus elementos.

Atenção:

Não faça confusão da linguagem JavaScript com a linguagem JAVA. Elas podem até ter alguma semelhança de sintaxe em alguns comandos, mas não vai além disso. Vale lembrar que a linguagem JAVA permite a criação de aplicações para *desktop*, web e dispositivos, ou seja, é uma plataforma mais completa.

A linguagem foi desenvolvida inicialmente pela Netscape e foi chamada de LiveScript. Já no fim de 1995, a Sun (que também desenvolveu o Java) se incorporou ao projeto e a linguagem foi nomeada de JavaScript. A versão da JavaScript atual e disponível nos navegadores é a 1.5.

Vamos conhecer algumas características da linguagem JavaScript:

linguagem não-tipada:

Isto significa que as variáveis não precisam ser criadas com um tipo específico. Os tipos são associados

Fomento ao Uso das Tecnologias da Informação e Comunicação

com valores atribuídos, não com variáveis. Por exemplo, a variável x poderia ser associada a um número e mais tarde associada a uma string.

Case sensitive:

Significa dizer, que o interpretador diferencia minúsculas de maiúsculas, tanto para variáveis/funções definidas pelo programador, como para todos os comandos da linguagem. Isso faz com que os comandos tenham que ser escritos da mesma forma que são apresentados. Caso isso não ocorra o comando passa a ser interpretado como uma variável.

Baseada em objetos:

É uma linguagem baseada em objetos nativos ou criados pelo programador, ou seja, trata todos os elementos de uma página como objetos, que normalmente são agrupados por tipo ou finalidade.

Sintaxe

Para começar, a programação JavaScript é incluída dentro dos documentos (X)HTML, porém precisamos indicar que estamos programando em JavaScript. Para isso, identifica-se da seguinte forma:

```
<script>
```

comandos...

```
</script>
```

Tudo que for escrito entre o <script></script> será considerado programação JavaScript. Porém, recomenda-se utilizar o atributo type que define o tipo de conteúdo do script. Ficando assim:

```
<script type="text/javascript">
```

comandos...

```
</script>
```

Outro forma é gerar um documento com a programação JavaScript a parte e fazer referência a este arquivo. Para tanto se utiliza o atributo src. Isso é bastante comum e até recomendado. Veja o exemplo abaixo.

```
<script type="text/javascript" src="js/funcoes.js">

</script>
```

Neste caso só vai a referência ao arquivo e o elemento é fechado. Veja que o padrão para extensão de arquivos que contenham programação JavaScript é js. Também é recomendado que se crie uma pasta específica para estes arquivos.

Além disso, é importante observar que existe uma padronização quanto a inclusão da programação JavaScript no documento. A convenção especifica que o elemento script deva estar dentro do cabeçalho do documento. Veja na figura B1.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <title>Exemplo</title>
5
6  <script type="application/javascript">
7
8  </script>
9
10 </head>
11
12 <body>
13 </body>
14 </html>
```

Figura B.1 - Elemento script
Fonte: do autor

Comentários

A convenção JavaScript para comentários é apresentada abaixo:

Tudo o que está escrito entre o // e o fim da linha será ignorado.

```
// comentário de linha
```

Também é possível incluir comentários em diversas linhas como abaixo:

```
/* comentário em

diversas linhas */
```

Atenção:

Não confundir os comentários Javascript e os comentários (X)HTML (<!—comentário -->).

Dicas:

Faça uso dos comentários ao longo da sua programação, isso pode ajudar na manutenção posterior do código e ainda faz você compreender melhor os recursos da linguagem.

Variáveis

Trabalhar com variáveis na linguagem JavaScript vai parecer mais fácil que outras linguagens, pois não é necessário declarar o tipo da variável. Para fazer a declaração use a palavra var. O nome da variável pode ser de qualquer tamanho, conter números, dígitos e underlines ('_'). Exemplos:

```
<script type="text/javascript">

        var nome;

        var preco_unitario;

        var valor1;

        var email = 'aaaa@teste.com.br';
```

Fomento ao Uso das Tecnologias da Informação e Comunicação

```
</script>
```

A variável assumirá um tipo de acordo com o valor que lhe for atribuído. Cabe ressaltar que a palavra **var** é opcional, porém é altamente recomendado que faça uso na declaração de forma a evitar erros.

Atenção:

Como JavaScript é case sensitive, a variável de nome teste é diferente da variável de nome Teste.

Dicas:

Na linguagem JavaScript você pode usar tanto aspas simples como duplas para indicar que um conteúdo é String.

Na linguagem JavaScript o ponto e vírgula (;) no fim da linha de comando é opcional, porém seu uso é recomendado.

Primeiro Script

Bem pessoal, vamos fazer o nosso "Olá mundo!" em JavaScript. Vocês podem usar o bloco de notas para codificar (não se esqueçam de salvar o arquivo com a extensão html). Veja o código da figura B.2:

- Na linha 11 foi criada a variável chamada teste com valor "olá mundo!!".
- Na linha 12 estamos mostrando o valor da variável teste no navegador. Para isso, estamos usando o ***document.write()***. Isto significa que estamos escrevendo no documento.

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5  <title>Exemplo</title>
6  </head>
7  <body>
8
9  <script type="application/javascript">
10
11     var teste = 'olá mundo!!';
12     document.write(teste);
13
14 </script>
15 </body>
16 </html>
```

Figura B.2 - Primeiro Script
Fonte: do autor

Atenção:

Na figura B.2 o script está dentro do elemento body, isso porque estamos escrevendo no documento.

A figura B.3 mostra o resultado da página no navegador.

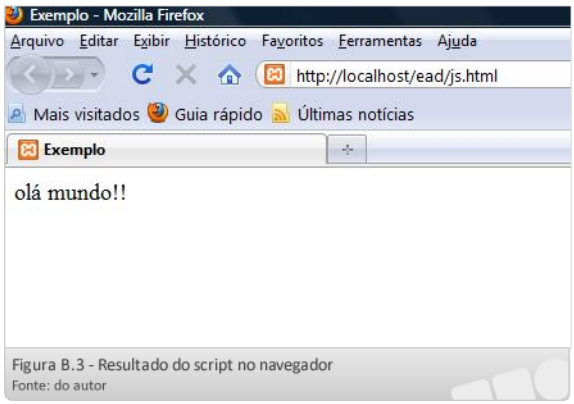


Figura B.3 - Resultado do script no navegador
Fonte: do autor

Operadores Lógicos

Operadores lógicos são operadores normalmente utilizados em comandos condicionais, como *if*, *for* e *while*

Operador lógico	Finalidade
==	Igual
!=	Diferente
>	Maior
<	Menor
>=	Maior ou igual
<=	Menor ou igual
&&	E
||	Ou (pipeline)

Operadores Lógicos

Operadores Matemáticos

Operador Matemético	Finalidade
+	Este operador serve para adição de valores e ao mesmo tempo, este operador pode ser usado na concatenação de strings. (concatenação = junção ou união) Exemplo: "programação" + "I" //retorna programaçãoI
-	Subtração de valores
*	Multiplicação de valores
/	Divisão de valores

%	Retorna o resto de uma divisão.
	Exemplo:
	`150 % 13 retornará 7`
	`7 % 3 retornará 1`

Operadores Matemáticos

Fomento ao Uso das Tecnologias da Informação e Comunicação

Expressões Simples com operadores

Operador Matemático	Finalidade
+=	Concatena /adiciona ao string/valor já existente. Exemplo: x += y é o mesmo que x = x + y, da mesma forma podem ser utilizados: -= , *= , /= ou %=

++	Acrescenta 1 no valor
	Exemplo:
	`X = 3;`
	`X++; // X valerá 4`

--	Decrementa 1 no valor
	Exemplo:
	`X = 3;`
	`X--; // X valerá 2`

Expressões Simples com operadores

Conversão de tipos

Em expressões que envolvam valores numéricos e strings com o operador +, os valores numéricos serão convertidos implicitamente para strings. Exemplo:

```
X = "A resposta é " + 35; // retorna "A resposta é 35"

Y = 35 + " é a resposta"      // retorna "35 é a resposta"

Y = "37" + 7 // retorna 377
```

Assim, as vezes torna-se necessário fazer a conversão de valores que estão como string para que possamos fazer cálculos. Para converter explicitamente strings em números existem as funções predefinidas *parseInt* e *parseFloat*.

parseInt: Converte uma string em um número inteiro.

Ex.:

```
num = "3A";

x = parseInt(num);          // x recebe 3
```

parseFloat: Converte uma string em um número real.

Ex.:

```
z = parseFloat("3.15");       // z recebe 3.15
```

Instruções

A seguir veremos as principais instruções condicionais e de repetição. Veremos que seguem uma sintaxe muito parecida de outras linguagens.

if

A instrução **if** é utilizada quando é necessário tomar decisões. Sua estrutura é:

```
if (<expressão>)
    <Instrução>
```

A expressão é avaliada, caso seja *true* (verdadeira) a instrução é executada, caso contrário a instrução não é executada;

Se o bloco de instrução possuir mais de uma instrução, deve-se usar início de bloco e fim de bloco, ou seja {}.

```
if (<expressão>){
      <Instrução>
}
```

Caso exista instrução a ser executada quando a expressão retornar false(falso), pode ser usar o else.

```
if (<expressão>){
      <Instrução>
}else{
      <instrução>
}
```

switch

O switch também pode ser utilizado para testes condicionais e é indicado para situações que exijam vários testes.

Sintaxe:

```
switch (variavel_de_controle) {
 case opção1 :
       comandos ;
       break;
 case opção2 :
       comandos ;
       break;
 default :
```

```
        comandos;
}
```

É importante observar que a instrução *break* é utilizada ao final de cada opção. O *break* faz com que a execução pule para o fim do *switch*. Isto é importante, uma vez que quando uma opção verdadeira for encontrada o switch segue executando todas as instruções até o fim se não possuir um *break*.

for

Para situações em que precisamos realizar um bloco de instrução diversas vezes, uma opção de solução é o comando de repetição for. Vamos supor que você tenha que escrever em seu código (X)HTML os números de 1 até 20. Para tanto, você pode utilizar o laço for.

Sintaxe:

```
for (<inicialização>;<condição>;<atualização variável>){
  <instruções>
}
```

Veja na figura B.4 como fica a construção do laço *for* para mostrar os números de 1 a 20. Na linha 5 foi concatenado o elemento "<br/>" para quebrar a linha cada vez que mostra o valor da variável **x**.

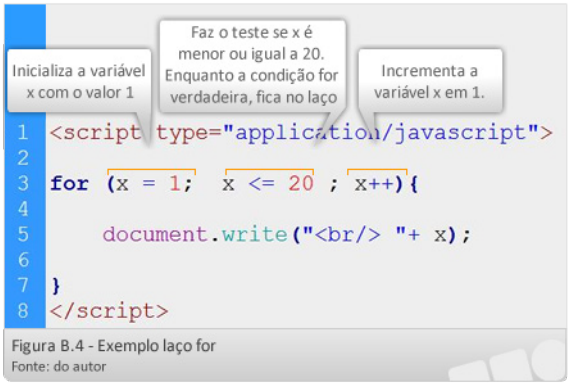


Figura B.4 - Exemplo laço for
Fonte: do autor

while

O while é semelhante ao comando for, porém às vezes o while é aplicado quando não sabemos determinar a quantidade de vezes que nosso laço vai executar as instruções. Neste laço, enquanto a condição do while for verdadeira as instruções serão executadas.

Sintaxe:

```
while(<condição>){
  <instruções>
{
```

Dicas:

Cuidado para que seu laço não seja infinito. A condição precisa ser falsa em algum momento para sair do

laço ou pode usar o comando *exit* para sair do laço.

A figura B.5 apresenta o exemplo que mostra os valores de 1 até 10. A variável **i** é que controla o laço. Observe que dentro do comando que mostra o valor da variável **i**, ela já é incrementada (i++).

```
<script type="application/javascript">

    var i=1;
    while(i <= 10){

        document.write(i++ +"<br>");
    }

</script>
```

Figura B.5 - Exemplo laço while
Fonte: do autor

Objetos JavaScript

Em JavaScript é utilizado o conceito de objetos para manipulação de todos os elementos. Tudo pode ser considerado objeto no mundo (uma caneta, um animal, uma pessoa, uma mesa). Mas vamos ver como a linguagem JavaScript trabalha baseada em objetos. Na figura B.6 podemos ver vários objetos. A janela do navegador é um objeto, o próprio documento é considerado um objeto, um formulário, uma caixa de texto, e assim por diante. Assim, existe uma hierarquia para manipulação dos objetos:

Janela(window) → documento(document) → formulário → elemento

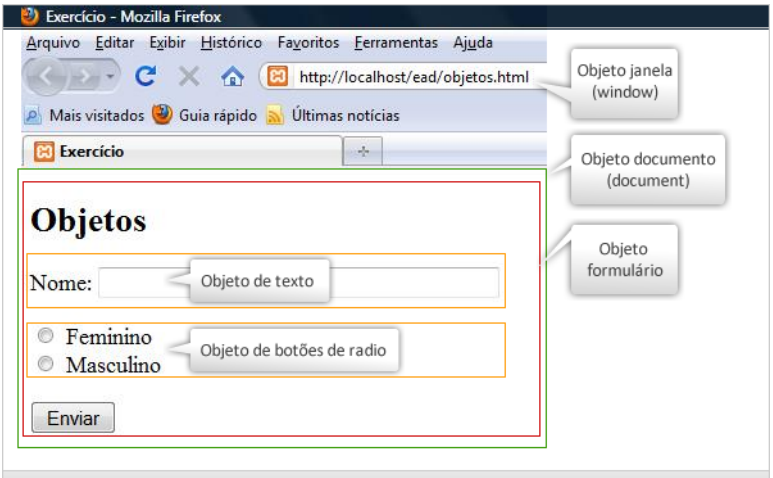


Figura B.6 - Exemplos de objetos JavaScript
Fonte: do autor

O objeto janela será acessado por meio do termo *window* e o objeto documento usando *document*. Para acessar o formulário e os demais elementos do formulário, é necessário saber a sua identificação. Vejamos o exemplo da figura B.7. Os objetos possuem um *name* e um *id*, que geralmente tem o mesmo valor (o radio é uma exceção). Nunca teremos dois objetos com o mesmo id, pois ele identifica de forma única o objeto. Já em relação a propriedade *name*, podemos ter mais de um objeto com o mesmo *name*. Observe nos balões o *id/name* de cada objeto.

Fomento ao Uso das Tecnologias da Informação e Comunicação

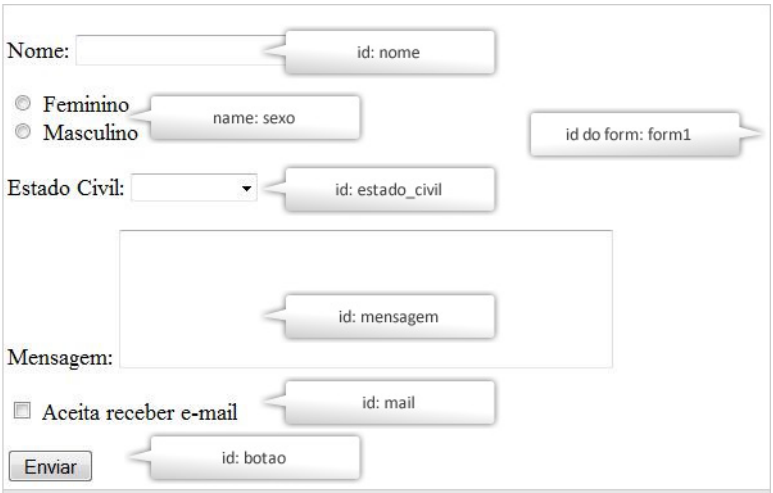


Figura B.7 - Formulário
Fonte: do autor

Só para relembrar vamos observar também o (X)HTML correspondente ao formulário acima. Veja na figura B.8. Observe que o id do formulário é *form1*.

```
<form id="form1" name="form1" method="post" action="">
    <label for="nome">Nome:</label>
    <input name="nome" type="text" id="nome" size="40" maxlength="40" /><br />
    <label> <input type="radio" name="sexo" value="F" id="sexo_0"
      Feminino</label><br />
    <label> <input type="radio" name="sexo" value="M" id="sexo_1" />
      Masculino</label>   <br />
    <label for="estado_civil">Estado Civil:</label>
    <select name="estado_civil" id="estado_civil">
      <option></option>
      <option value="S">Solteiro</option>
      <option value="C">Casado</option>
      <option value="D">Divorciado</option>
      <option value="V">Viúvo</option>
      <option value="O">Outro</option>
    </select>   <br />
    <label for="mensagem">Mensagem:</label>
    <textarea name="mensagem" id="mensagem" cols="45" rows="5"></textarea>  <br />
    <input type="checkbox" name="mail" id="mail" />
    <label for="mail">Aceita receber e-mail</label> <br />
    <input type="submit" name="botao" id="botao" value="Enviar" />
</form>
```

Figura B.8 - Código (X)HTML do formulário
Fonte: do autor

O *id* e o *name* são propriedades dos objetos. Uma **propriedade** é um atributo, uma característica, uma descrição do objeto. Uma propriedade tem um nome e um valor associado. Para acessar as propriedades, utiliza-se a sintaxe:

```
nome_do_objeto.nome_da_propriedade
```

Vejamos um exemplo de propriedade para o objeto nome do nosso formulário da figura B.7. Se desejarmos "pegar" o valor que o usuário informar na caixa de texto nome, ou "atribuir" um valor a ele, usamos a propriedade value. Veja com o fica:

```
x = document.form1.nome.value; //pegando valor

document.form1.nome.value = 'teste';//atribuindo valor
```

Vamos ver como fica para os demais objetos do nosso formulário na tabela abaixo:

Objeto	Id	JavaScript para pegar valor
Nome:	nome	`document.form1.nome.value`
Estado Civil:	estado_civil	`document.form1.estado_civil.value` obs.: retorna o valor do item selecionado
Mensagem:	mensagem	`document.form1.mensagem.value`

Aceita receber e-mail	mail	`document.form1.mail.value`
		Obs.: se estiver marcado retorna o valor senão retorna *false*.
		• Para verificar se o campo está marcado usa-se a propriedade *checked* que retorna *true* se estiver marcado e *false* se estiver desmarcado:
		`document.form1.mail.checked`

Feminino Masculino	sexo	Obs.: os botões de radio funcionam um pouco diferente. Vejamos: • Os botões de radio são utilizados para escolher apenas uma opção, e só uma, entre um conjunto. Por isso, todos os botões tem o mesmo nome, justamente para que seja possível desmarcar os outros botões de mesmo nome quando um for marcado. Assim, cada botão está relacionado a um índice do conjunto, começando com o índice zero(0). No nosso exemplo o primeiro botão "Feminino" é o índice zero e o botão "Masculino" o índice 1. • É necessário verificar se o botão está marcado, para então pegar o valor dele. `if (document.form1.sexo[0].checked){ x = document.form1.sexo[0].value };`

Atenção:

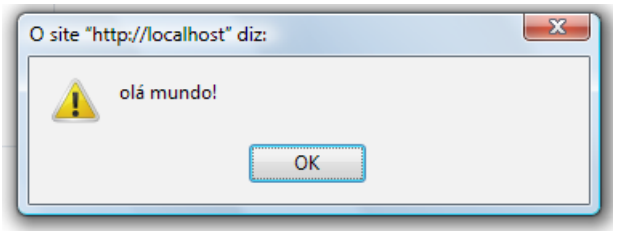
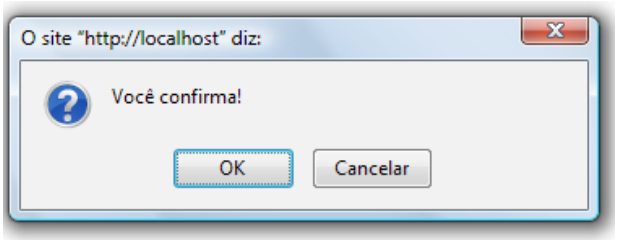
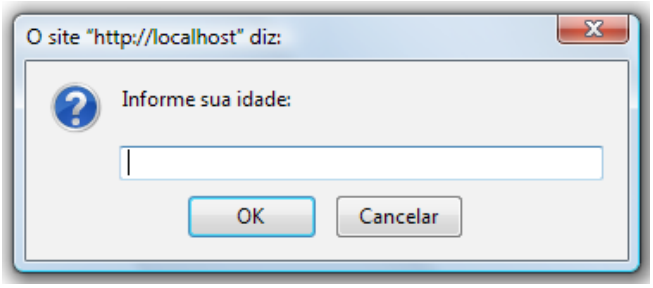
Lembre-se que JavaScript é *case sensitive*, ou seja diferencia maiúsculas e minúsculas.

Outro conceito importante relacionado a objetos é referente a **métodos**. Vimos que as propriedades são informações/características dos objetos, já os métodos, são ações realizadas sobre os objetos (gatos miam, carros correm, canetas escrevem, etc.). Vamos ver um exemplo já usado anteriormente. Sabemos que o nosso documento é considerado um objeto e vimos que podemos escrever no documento:

Fomento ao Uso das Tecnologias da Informação e Comunicação

```
document.write('escrevendo no documento');
```

No exemplo acima, usamos o método *write()* do objeto *document* que mostra no documento o que foi passado como parâmetro, no nosso caso a *string* 'escrevendo no documento'. Vejamos outros exemplos de métodos:

Objeto/método	Finalidade
`window.alert()`	Mostra uma mensagem de alerta. `window.alert('olá mundo!');` 
`window.confirm()`	Mostra uma mensagem de confirmação com botões de OK e Cancelar. Se o botão OK for acionado retorna *true*, se acionar o botão Cancelar retorna *false*. `window.confirm('Você confirma!');` 
`window.prompt()`	Mostra uma caixa de diálogo para entrada de dados. O segundo parâmetro é referente ao valor padrão que aparece na caixa de texto. Ao clicar no botão OK o valor informado será retornado. `window.prompt('Informe sua idade:','');` 

Métodos

Agora que conhecemos um pouco sobre métodos, podemos ver outra forma de manipular os objetos. Vamos ver o método *getElementById()* do objeto *document*. Este método retorna o objeto de qualquer elemento do documento que tenha o *id* passado por parâmetro. Por exemplo, para pegar o valor do objeto *nome* do nosso formulário da figura B.7, usamos:

```
var x = document.getElementById('nome').value;
```

Ou

```
var obj = document.getElementById('nome');
var x   = obj.value;
```

Quando 'pegamos' o objeto podemos manipular seus atributos e métodos como mostrado no exemplo acima, com a variável *obj* que recebeu o objeto *nome*.

Eventos JavaScript

Eventos são disparados a partir de algumas ações que ocorrem com os objetos, por exemplo, quando se clica em algo, quando passa o mouse sobre um objeto, quando algum objeto recebe o foco. Nestes casos existem situações que precisamos fazer algo, por isso existem os eventos. Vejamos alguns eventos importantes:

Eventos	
click	Quando clica sobre um botão, um *link* ou outros elementos.
load	Quando a página é carregada pelo navegador (aplicado ao elemento *body*).
unload	Quando saia da página (aplicado ao elemento *body*).
mouseover	Quando coloca o ponteiro do mouse sobre um *link* ou outro elemento.
mouseout	Quando o ponteiro do *mouse* sai de um *link* ou outro elemento.
focus	Quando um elemento de formulário tem o foco, isto é, está ativo.
blur	Quando um elemento de formulário perde o foco, isto é, quando o deixa de estar ativo.
change	Quando o valor de um campo de formulário é modificado.
select	Quando seleciona um campo dentro de elemento *select* de um formulário.
submit	Quando clica sobre o botão *Submit* para enviar um formulário.

Vamos ver um exemplo na figura B.9. O elemento botão, cujo id é *botao*, está com um código no evento *onclick*. Veja que está se programando em JavaScript dentro das aspas duplas. Isto pode ser feito dentro de um evento que o navegador vai interpretar corretamente.

```
<form id="form1" name="form1" method="post" action="">
  <label for="nome">Nome:</label>
    <input name="nome" type="text" id="nome" size="40" maxlength="40" />
    <input type="submit" name="botao" id="botao" value="OK"
    onclick="alert('clicou no botão');" />
</form>
```

Figura B.9 - Código (X)HTML do formulário
Fonte: do autor

O resultado será uma mensagem de alerta quando o botão for clicado. Veja na figura B.10.

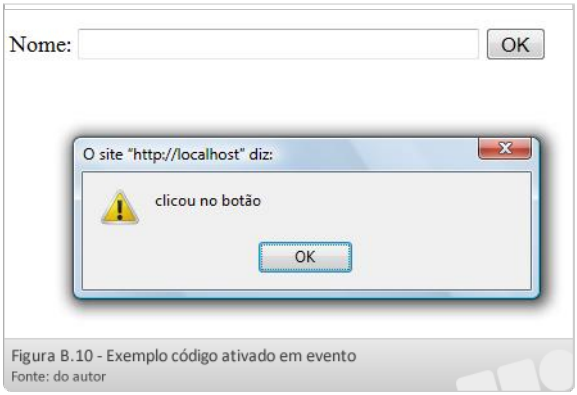


Figura B.10 - Exemplo código ativado em evento
Fonte: do autor

Fomento ao Uso das Tecnologias da Informação e Comunicação

Eventos são bastante utilizados em JavaScript, principalmente para chamada de funções predefinidas pelo programador que veremos a seguir.

Funções

Uma função é um conjunto de instruções destinado a uma tarefa bem específica e que podemos utilizar várias vezes. A utilização de funções melhora bastante a leitura do script. Em Javascript existem dois tipos de funções: as funções próprias do Javascript, que chamamos de "métodos", como já visto, e as funções escritas pelo programador. É este último tipo que nos interessa agora.

Para declarar ou definir uma função, utiliza-se a palavra function:

```
function nome_da_funcao(parâmetros) {


        // instruções

}
```

Atenção:

Lembre-se que JavaScript é case sensitive e que o nome de uma função deve ser único, ou seja não podemos ter duas funções com o mesmo nome

Os parâmetros são opcionais, mas os parênteses devem sempre aparecer. Uma função só é executada quando chamamos a função, ou seja, apenas a sua definição não executa as instruções pertencentes a ela. A chamada ou invocação de uma função se faz pelo nome da função com parênteses, por exemplo:

```
nome_da_funcao();
```

Atenção:

A função deve estar definida antes de ser chamada, caso contrário um erro será gerado

Uma função pode retornar um valor, neste caso usa-se a palavra reservada *return*. Vamos a um exemplo. A figura B.11 apresenta o código JavaScript que cria uma função que recebe dois valores por parâmetro. A função soma os dois valores e retorna o resultado. A definição da função começa na linha 9 e termina na linha 14. A seguir na linha 16 é feita a chamada da função passando os dois valores para cálculo. Como a função retorna o valor total, será mostrado no navegador o resultado conforme figura B.12.

Dicas:

É convenção inserir todas as declarações de funções no cabeçalho da página, isto é entre os elementos *<head>...</head>*

```
1  <!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN"
   "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">
2  <html xmlns="http://www.w3.org/1999/xhtml">
3  <head>
4  <meta http-equiv="Content-Type" content="text/html; charset=utf-8" />
5  <title>Exercício</title>
6
7  <script type="text/javascript">
8
9  function somar(v1, v2){
10
11     var total = v1 + v2;
12     return total;
13
14 }
15
16 document.write("A soma dos valores é igual a " + somar(3,5));
17
18 </script>
19 </head>
```

Dois valores recebidos por parâmetro.

Retornando valor calculado

Retornando valor calculado

Figura B.11 - Exemplo função
Fonte: do autor

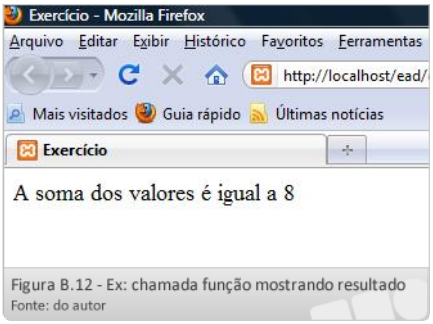


Figura B.12 - Ex: chamada função mostrando resultado
Fonte: do autor

Com as funções, é importante compreender o uso de **variáveis locais** e **globais**.

Uma variável declarada dentro uma função pela palavra chave ***var*** será válida apenas dentro da própria função. Não se pode assim usá-la fora da função. Chamamos assim variável local. Se a variável é declarada contextualmente (sem utilizar a palavra var), a sua invocação será global. As variáveis declaradas logo no início do script, fora e antes de todas as funções, serão sempre globais, quer ela seja declarada com var ou de maneira contextual.

Atividades – Parte 1

1. Faça um código JavaScript que mostra uma caixa de diálogo (prompt) para o usuário informar sua idade. Se o usuário informar uma idade menor que 18 mostrar em uma caixa de alerta a mensagem “Menor de idade”. Se o usuário informar um valor maior ou igual a 18, mostrar mensagem “Maior de idade”.

2. Crie um formulário de contato e torne os campos: nome, e-mail e assunto obrigatórios, validando-os através de uma função JavaScript. Para cada campo não preenchido mostre uma mensagem de alerta.

3. No processo de seleção para cargos de uma empresa, o candidato faz 3 provas, cujos os pesos são:

- Prova 1 = peso 3
- Prova 2 = peso 3
- Prova 3 = peso 4

Para ser considerado aprovado o candidato deve ter média acima de 7 para o cargo de gerente e acima de 8 para o cargo de diretor. Faça uma função JavaXcript para calcular a nota final do candidato e mostra em uma caixa de alerta sua situação (APROVADO/REPROVADO) considerando o formulário abaixo.

```
<form name="form1" method="post" action="cargo.php">
```

Fomento ao Uso das Tecnologias da Informação e Comunicação

```
Nota prova 1 : <input name="p1" type="text" id="p1">  <br>
Nota prova 2 : <input name="p2" type="text" id="p2">  <br>
Nota prova 3 : <input name="p3" type="text" id="p3">  <br>
Cargo:
<select value="cargo">
   <option></option>
   <option value="7">Gerente</option>
   <option value="8">Diretor</option>
</select>
<input name="verificar" type="submit" id="Verificar" value="Verificar">
</form>
```

4. Crie a função *calculadora()* que considera os objetos do formulário (dois valores e a operação que pode ser +, -, * ou /). A função deve fazer o cálculo correspondente a operação e mostrar o resultado em uma caixa de alerta. Considere um formulário como da figura abaixo. Faça a chamada da função no evento *onclick* do botão

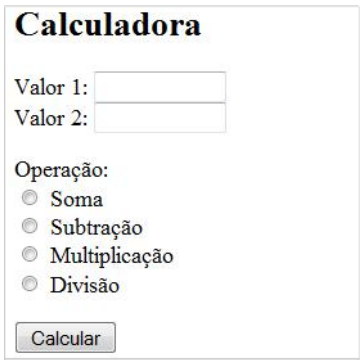


Figura B.13 - Exercício 4
Fonte: do autor

5. Dado o enunciado abaixo marque a opção correta para a estrutura de função:

Faça uma função que recebe por parâmetro um nome de objeto e dois valores. A função deve fazer a média dos dois valores recebidos e mostrar o resultado no objeto cujo nome foi recebido por parâmetro.

a)
```
function fteste(x, a, b){
   m = (a + b) /2;
   obj = document.
getElementById('x');
   x.value = m;            if *
}
```

b)
```
function fteste(x, a, b){
   m = (a + b) /2;
   obj = document.
getElementById(x);
   obj.value = m;          if *
}
```

c)
```
function fteste(x, a, b){
   m = (a + b) /2;
   obj = document.
getElementById(x);
   x.value = m;            if *
}
```

d)
```
function fteste(x, a, b){
   m = (a + b) /2;
   x.value = m;            if *
}
```

6. Dado o enunciado abaixo marque a opção correta para a estrutura de função:

Faça uma função que recebe por parâmetro dois objetos. A função deve concatenar os dois valores dos objetos e mostrar na div cujo id é 'prova'.

a)

```
function fteste(a, b){
  v1 = document.getElementById(a).value;
  v2 = document.getElementById(b).value;
  document.getElementById('prova').value = v1+v2;
}
```

b)

```
function fteste(a, b){
   p = a + b;
  document.getElementById('prova').innerHTML =   p;
}
```

c)

```
function fteste(a, b){
   p = a.value + b.value;
  document.getElementById('prova').innerHTML =   p;
}
```

d)

```
function fteste(a, b){
   p = a.value + b.value;
  document.getElementById('prova').value =   p;
}
```

7. Pesquise e apresente atributos não vistos neste documento para os objetos: *window*, *document* e objeto de formulário select.

8. Pesquise e apresente métodos não vistos neste documento para os objetos: *window*, *document* e objeto de formulário *input*.

9. Apresente um exemplo de uso de um evento em elementos de formulário (diferente de *onclick*).

10. Como uma função JavaScript pode ser chamada em um *link*? Apresente um exemplo.

Fomento ao Uso das Tecnologias da Informação e Comunicação

Introdução a linguagem JavaScript – Parte 2

Continuando a unidade sobre linguagem do lado cliente, nesta parte 2 do material abordaremos a manipulação de arrays, strings e datas em JavaScript. Ao final desta parte são apresentadas a atividades para realização.

Arrays

Array, ou vetor, também é tratado como um objeto em JavaScript. O objeto Array é uma lista de elementos indexados nos quais pode-se guardar (escrever) dados ou as invocar (ler). O primeiro elemento do Array está no índice zero(0). Para criar um Array precisamos inicialmente definir a sua estrutura:

```
var nome_do_array = new Array (num);
```

Onde *num* é o número de elementos do array, sendo o máximo 255.

Para atribuir valores ao Array:

```
nome_do_array [i] = “valor”;
```

Onde i é um número compreendido entre 0 e *num* menos 1.

Exemplo: uma lista de quatro frutas

```
var frutas = new Array(4);

frutas[0]=”maça”;

frutas[1]=”banana”;

frutas[2]=”abacaxi”;

frutas[3]=”lima”;
```

ou podemos definir o vetor da forma abaixo:

```
var frutas = new Array(‘maça’,’banana’,’abacaxi’,‘lima’);
```

Para mostrar os dados de um array é muito comum usar o uma estrutura de repetição. Vejamos um exemplo utilizando o laço *for*.

```
for(i=0; i <= frutas.length -1; i++){


        document.write (‘<br/>’ + frutas[i]);

}
```

Veja que utilizamos a propriedade *length* do vetor para recuperar o número de elementos do vetor. Assim, se temos um vetor de tamanho 4 (quatro elementos), nosso laço vai variar de *i* valendo 0 até *i* igual a 3.

Alguns métodos do objeto Array:

join()	Junta todos os elementos do array em uma única cadeia. Os elementos são separados por um caractere separador especificado no argumento. Por padrão, este separador é uma vírgula.
	`var x = frutas.join();`
	`document.write(x);`

sort()	Ordena os elementos de forma crescente. `frutas.sort();` `for(i=0; i <= frutas.length -1; i++){` `document.write (' ' + frutas[i]);` `}`
reverse()	Inverte a ordem dos elementos no array. `frutas.reverse();` `for(i=0; i <= frutas.length -1; i++){` `document.write (' ' + frutas[i]);` `}`

Strings

String é uma seqüência de letras, dígitos, caracteres de pontuação e outros, que são representados pela linguagem como texto. Strings literais podem ser usadas delimitando por pares de aspas simples('...') ou aspas duplas ("... "). Em JavaScript, strings também podem ser manipuladas como objetos. A seguir veremos as principais propriedades e métodos associados a strings. Para criar explicitamente um objeto String a sintaxe é:

```
var texto = new String();
```

ou ainda:

```
var texto = new String('olá mundo!');
```

Considerando o objeto texto criado acima, vejamos a propriedade length que é bastante utilizada:

```
var tamanho = texto.length;
```

A propriedade *length* etorna o tamanho da string, ou seja, quantos caracteres possui.

Agora vamos ver na tabela abaixo os principais métodos do objeto String:

charAt()	Extrai o caractere em uma dada posição da string
	`var caracter = texto.charAt(4);`
	//retorna **m**

`indexOf()`	Pesquisa um caractere ou substring em uma string.
	Retorna a posição da primeira ocorrência da substring que aparece depois da posição inicial. Retorna -1 se não encontrar.
	`var retorno= texto indexOf('!',0);`
	// neste exemplo está procurando o caractere ! a partir da posição 0(zero)
	// retorna 9

`split()`	Divide um string em um array de strings, quebrando em um string delimitador.
	`var texto = '10/10/2012';`
	`var vetor = texto.split('/');`
	// vetor conterá 3 elementos que foram separados pelo delimitador /
	// retorna vetor['10','10','2012']

`substring()`	Extrai um substring de um string.
	`var texto = 'Olá Mundo!';`
	`var partetexto = texto.substring(4,9);`
	//observe que os parâmetros são posição inicial e posição final; porém o caractere da posição final não é incluído no retorno.
	// partetexto recebe 'Mundo'

`toLowerCase()`	Retorna uma cópia da string, com todos os caracteres convertidos em letras minúsculas.
	`var texto = 'Olá Mundo';`
	`var novotexto = texto.toUpperCase();`
	// novotexto recebe 'olá mundo'

`toUpperCase()`	Retorna uma cópia da string, com todos os caracteres convertidos em letras maiúsculas.
	`var texto = 'Olá Mundo';`
	`var novotexto = texto.toUpperCase();`
	// novotexto recebe 'OLÁ MUNDO'

Data e Hora

Trabalhar com datas e horas também é muitas vezes necessário para incorporar algum recurso nas páginas (X)HTML. Por exemplo, mostrar a data atual, mostrar uma saudação de acordo com o horário. Em JavaScript, data e hora são manipulados como objeto, portanto precisamos criar um objeto para conseguir manipulá-los:

```
var datahora = new Date();
```

Atenção:

A "data e hora" retornadas no exemplo acima são do computador do utilizador. Lembre-se JavaScript é interpretado pelo navegador

Se mostrar a variável datahora será mostrado no navegador algo semelhante a:

```
Tue Jun 14 2011 20:41:13 GMT-0300 (Hora oficial do Brasil)
```

Quando não passamos parâmetros na criação do objeto será retornada a data hora atual. No entanto, podemos criar um objeto data passando alguns valores. Exemplo:

```
var data = new Date(2000, 11, 30, 22, 50, 0);
```

Os valores passados por parâmetro são na sequência: ano, mês, dia, hora, minutos e segundos. Os valores para hora são opcionais, ou seja, podemos criar um objeto passando apenas os valores da data, como no exemplo abaixo:

```
var data = new Date(2000, 11, 30);
```

Agora que sabemos criar o objeto vamos ver os métodos que podemos usar na tabela abaixo:

getDate()	Retorna um inteiro correspondente ao dia do mês entre 1 e 31. `var dia = datahora.getDate();`
getDay()	Retorna um inteiro correspondente ao dia da semana compreendido entre 0 e 6 (0 para Domingo, 1 para Segunda-feira, ...). `var dia_semana = datahora.getDay();`
getMonth()	Retorna um inteiro correspondente ao mês entre 0 e 11 (0 para janeiro, 1 para fevereiro, 2 para março, ...). `var mes = datahora.getMonth();`
getYear()	Retorna um inteiro correspondente ao ano. `var dia = datahora.getYear();`
getHours()	Retorna um inteiro correspondente a hora entre 0 e 23. `var dia = datahora.getHours();`
getMinutes()	Retorna um inteiro correspondente aos minutos entre 0 e 59. `var dia = datahora.getMinutes();`
getSeconds()	Retorna um inteiro correspondente aos segundos entre 0 e 59. `var dia = datahora.getSeconds();`

Atividades – Parte 2

1. Considerando o formulário abaixo, faça uma função JavaScript que valida os campos para os seguintes critérios:

- todos os campos são obrigatórios
- o nome do usuário deve ter no mínimo 8 caracteres
- a senha deve conter no mínimo 6 caracteres
- os campos senha e senhaconf devem possuir o mesmo valor

Mostrar alertas para o usuário para cada inconsistência encontrada.

Fomento ao Uso das Tecnologias da Informação e Comunicação

```
<form name="form1" method="post" action="">

 Nome:<input name="nome" type="text" id="nome" size="20"> <br>

 Senha:<input name="senha" type="text" id="senha" size="20"><br>

 Confirma senha: <input name="senhaconf" type="text" id="senhaconf" size="20">
<br>

<input name="Enviar" type="button" id="Enviar" value="Enviar">

</form>
```

2. Dado um formulário com um campo para informar o e-mail, faça uma função JavaScript que verifica o e-mail para os seguintes critérios:

- campo é de preenchimento obrigatório
- e-mail deve possuir no mínimo 6 caracteres
- e-mail deve possuir o caractere @
- e-mail deve possuir o caractere . (ponto)

Crie um formulário para usar a função.

3. Faça uma função JavaScript que recebe por parâmetro uma data e retorna o dia da semana por extenso (Domingo, Segunda-feira, Terça-feira, ...). Mostre um exemplo de uso da função.

4. Faça uma função JavaScript que recebe por parâmetro uma data e retorna o mês por extenso. Mostre um exemplo de uso da função.

5. Faça uma função JavaScript que retorna a data atual e uma saudação no seguinte formato:

14/06/**20**11 - Boa noite!

6. Dado o formulário abaixo, faça uma função que calcule o gasto calórico considerando o peso, a atividade desenvolvida e o tempo que a atividade foi realizada. Mostre o gasto calórico na *div* mensagem. Considere a seguinte fórmula para o cálculo:

```
peso * tempo * (valor da atividade física) * 0.0175
```

Dicas:

Para verificar qual a atividade que foi selecionada, você pode utilizar um laço. Lembre-se que o botão de rádio considera cada opção como um índice (atividades[0], atividades[1], atividades[2],...).

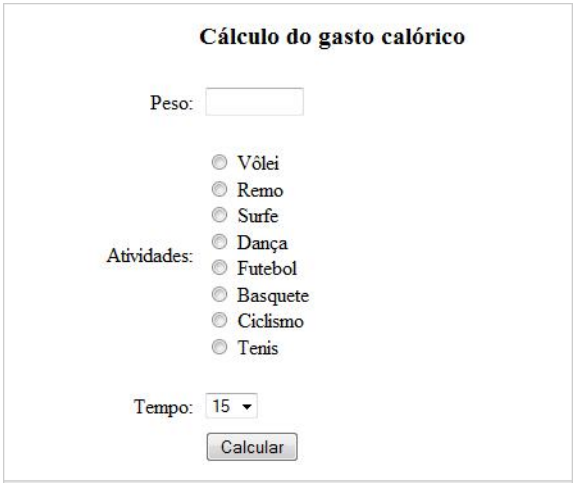


Figura B.14 - Formulário
Fonte: do autor

Segue o código (X)HTML do formulário acima:

```
<!DOCTYPE html PUBLIC “-//W3C//DTD XHTML 1.0 Transitional//EN” “http://www.
w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd”>
<html xmlns=”http://www.w3.org/1999/xhtml”>
<head>
<meta http-equiv=”Content-Type” content=”text/html; charset=utf-8” />
<title>Gasto Calórico</title>
<script src=”funcao_calorias.js” type=”text/javascript”>
</script>
</head>
<body>
<form id=”form1” name=”form1” method=”post” action=””>
 <table width=”600” border=”0” align=”center” cellpadding=”4” cellspacing=”2”>
    <tr> <td colspan=”2” align=”center”><h3><strong>Cálculo do gasto calórico
</strong></h3></td>
    </tr>
    <tr>
      <td width=”185” align=”right”><label for=”peso”>Peso:</label></td>
      <td width=”393”><input name=”peso” type=”text” id=”peso” size=”8” /></
td>
    </tr>
    <tr>
      <td align=”right”>Atividades:</td>
      <td><p>    <label>
          <input type=”radio” name=”atividades” value=”3” id=”atividades_0” />
          Vôlei</label>       <br />
        <label><input type=”radio” name=”atividades” value=”4”
id=”atividades_1” />
          Remo</label>        <br />
        <label><input type=”radio” name=”atividades” value=”4.5”
id=”atividades_2” />
          Surfe</label>       <br />
        <label><input type=”radio” name=”atividades” value=”5.5”
id=”atividades_3” />
          Dança</label>        <br />
```

```
        <label><input type="radio" name="atividades" value="6"
id="atividades_4" />
          Futebol</label>       <br />
        <label><input type="radio" name="atividades" value="8"
id="atividades_5" />
          Basquete</label>      <br />
        <label><input type="radio" name="atividades" value="7"
id="atividades_6" />
          Ciclismo</label>      <br />
        <label><input type="radio" name="atividades" value="9"
id="atividades_7" />
          Tenis</label>        <br />
      </p></td>
    </tr>
    <tr>
      <td align="right"><label for="tempo">Tempo:</label></td>
      <td><select name="tempo" id="tempo">
        <option value="15">15</option>
        <option value="30">30</option>
        <option value="45">45</option>
        <option value="60">60</option>
        <option value="75">75</option>
        <option value="90">90</option>
      </select></td>
    </tr>
    <tr>
      <td align="right"> </td>
      <td><input type="button" name="botao" id="botao" value="Calcular"
      onclick="fcalorias()" /></td>
    </tr>
    <tr>
      <td align="right"> </td>
      <td><div id="mensagem"></div></td>
    </tr>
  </table>
</form>
</body>
</html>
```

Introdução a linguagem JavaScript – Parte 3

Nesta última parte do material da linguagem JavaScript serão apresentados propriedades e métodos dos principais objetos JavaScript e um exemplo de resolução de problema desenvolvido passo a passo. Ao final desta parte, serão apresentadas as atividades para realização.

Objeto window

O objeto window representa a janela do navegador a qual o script está em execução. Este objeto possibilita o acesso a propriedades e execução de métodos. Assim, a janela do navegador pode ser manipulada de diferentes formas, por exemplo, modificando o seu tamanho, aparência ou posição, podemos abrir e fechar janelas, transferir informações entre janelas e criar janelas de diálogo.

A tabela abaixo apresenta as principais propriedades do objeto **window**:

defaultStatus	Determina o conteúdo padrão da barra de status do navegador, quando nada de importante estiver acontecendo. `Ex: widow.defaultStatus= 'texto'`
status	Define uma mensagem que aparecerá na barra de status do navegador, em substituição, por exemplo, a URL de um link, quando o mouse estiver sobre o link. `Ex: window.status="texto"`
name	Contém o nome da janela.

A tabela abaixo apresenta os principais métodos associados ao objeto **window**:

open("URL") ou **open("URL","nome") ou** **open("URL","nome",** **"características")**	Abre uma nova janela contendo o documento indicado pela URL. Opcionalmente, a janela pode ter um nome que pode ser usado em HTML, ou especificar características como tamanho, layout, etc. O método retorna uma referência do tipo window para a janela criada: `filha = window.open("filha.htm");`
close()	Fecha uma janela.
focus()	Torna uma janela ativa (traz para frente das outras, se for uma janela independente)

As janelas abertas podem ter suas características alteradas no momento em que são abertas. Estas características deverão vir em uma string com uma lista de opções separadas por vírgulas, como o terceiro argumento opcional do método open(). As características estão na tabela abaixo:

height=h	h é a altura da janela em pixels: `height=150`
width=w	w é a largura da janela em pixels: `width=300`
resizable	Se estiver presente permite redimensionar a janela
toolbar	Se estiver presente, mostra a barra de ferramentas do browser
scrollbars	Se estiver presente, mostra as barras de rolagem do browser
menubar	Se estiver presente, mostra a barra de menus do browser

Fomento ao Uso das Tecnologias da Informação e Comunicação

location	Se estiver presente, mostra o campo para entrada de URLs
status	Se estiver presente, mostra a barra de status

Cada característica pode ou não ter um valor. Não deverá haver espaços em qualquer lugar da lista. Por exemplo:

```
window.open("teste.html", "j2", "height=200,width=400,status");
```

O código acima abre uma janela de 200 pixels de altura por 400 de largura sem barra de ferramentas, sem barra de diretórios, sem campo de entrada de URLs, sem barra de menus, não-redimensionável e com barra de status.

Atenção:

Vale lembrar os métodos do objeto ***window*** que já foram abordados na parte 1 desta unidade, tais sejam: *alert() , prompt() e confirm()*.

Objeto history

O objeto **history** está associado ao objeto **window** e armazena as informações sobre os URLs que foram visitados antes e depois do atual e inclui métodos para ir para as localizações anteriores ou próximas:

go(±n) ou **go("string")**	Avança ou volta n páginas no histórico. A segunda forma procura no histórico até encontrar a primeira página que tenha a string especificada no título do documento ou nas palavras da sua URL. Ex.: `window.history.go(+1)`
back()	Volta uma página no histórico (simula o botão "Back" ou "Voltar" do browser). Ex.: `window.history.back()`
forward()	Avança uma página no histórico (simula o botão "Forward" ou "Avançar" do browser). Ex.: `window.history.forward()`

Objeto Navigator

O objeto **navigator** (associado ao objeto **window**) representa as propriedades do navegador. Usando suas propriedades e métodos é possível identificar características do navegador e desenvolver páginas personalizadas com conteúdo específico para aproveitar ao máximo os recursos existentes. Veja abaixo uma lista de propriedades vinculadas ao navigator:

userAgent	Uma string com a informação contida no cabeçalho HTTP User-Agent. Esta propriedade é a combinação das propriedades appCodeName e appVersion.
	Exemplos:
	`Mozilla/4.0 (compatible; MSIE 4.0; Windows 95)`
	`Mozilla/4.5 [en] (Win95; I)`

appCodeName	Contém o nome interno do navegador. Exemplo: Mozilla

appVersion	Contém informações sobre a versão.
	Exemplos:
	`4.0 (compatible; MSIE 4.0; Windows 95)`
	`4.5 [en] (Win95; I)`

appName	Contém o nome oficial do browser.
	Exemplos:
	`Microsoft Internet Explorer`
	`Netscap`

Objeto document

O objeto document representa o documento HTML atual. document é uma propriedade de window e, portanto, pode ser usado sem fazer referência a window:

```
window.document // ou simplesmente document
```

As principais propriedades do tipo **document** estão listadas na tabela abaixo:

title	Recupera o título do documento
location	Contém String com URL do documento. Pode ser alterado. Ex.: `document.location = "http://www.ifsul.edu.br";`
URL	Mesma coisa que location.
lastModified	Contém a data da última modificação do arquivo. Está no formato de data do sistema. Pode ser convertida usando *Date.parse()* e transformada em objeto ou automaticamente em String.

Atenção:

Com relação aos principais métodos de **document**, já vimos na parte 1 desta unidade. Vale relembrar, por exemplo, do *document.write* utilizado para escrever algo no documento.

Fomento ao Uso das Tecnologias da Informação e Comunicação

Solução passo a passo

O objetivo de solução de um problema passo a passo é mostrar de forma integrada a utilização dos recursos estudados e que auxiliarão no desenvolvimento das atividades propostas.

Então, mãos à obra. O enunciando do nosso problema é:

O formulário abaixo solicita que o usuário informe o peso, escolha uma atividade física e o tempo em que a atividade foi realizada. Dado este formulário, calcule, de acordo com a atividade física, o peso e o tempo de exercício, quantas calorias você queima praticando esporte. O cálculo usa a seguinte fórmula:

GastoCalorico = peso * tempo * (valor da atividade física) * 0.0175

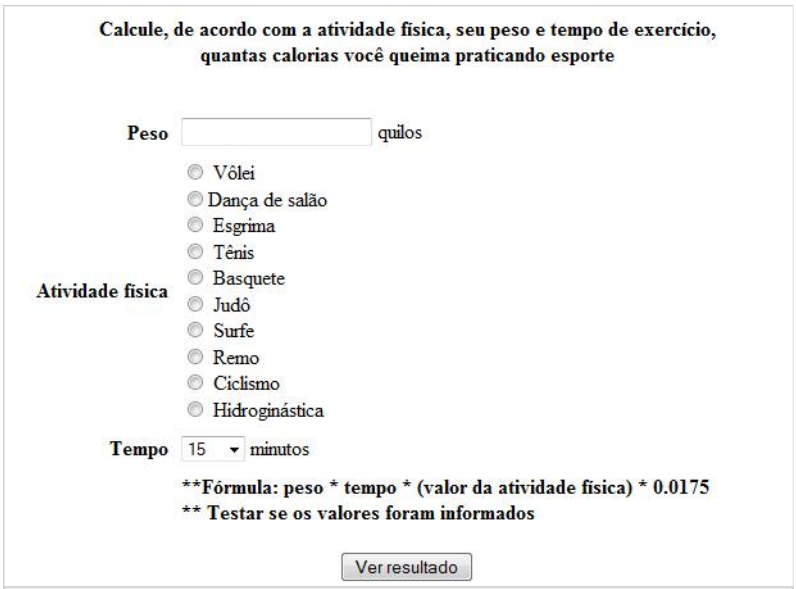


Figura B.15 - Formulário
Fonte: do autor

Agora vamos à solução. Primeiro vamos construir o formulário:

a) Criar um arquivo HTML nomeado exercicio.html
b) Incluir um título para o documento (Exercício passo a passo)
c) Incluir um elemento form
d) Incluir um elemento table com a seguinte formatação:
   - 8 linhas
   - 2 colunas
   - width=”50%”
   - border=”0”
   - align=”center”
   - cellpadding=”0”
   - cellspacing=”10”

Dica:

Vamos utilizar uma tabela com duas colunas para organizar os elementos do formulário de forma que fiquem alinhados.

Agora vamos incluir os elementos de formulário:

a) uma caixa de texto para peso (*id= peso*)

```
<input type="text" id="peso" name="peso" />
```

b) botões de rádio para as atividades (*name=atividade*)

```
<input type="radio" name="atividade" value="3" /> V&ocirc;lei <br />
<input type="radio" name="atividade" value="5.5" />Dan&ccedil;a de sal&atilde;o<br />
<input type="radio" name="atividade" value="6" />  Esgrima<br />
<input type="radio" name="atividade" value="6" />  T&ecirc;nis<br />
<input type="radio" name="atividade" value="8" />  Basquete<br />
<input type="radio" name="atividade" value="10" /> Jud&ocirc;<br />
<input type="radio" name="atividade" value="3" />  Surfe<br />
<input type="radio" name="atividade" value="3.5" /> Remo<br />
<input type="radio" name="atividade" value="4" />   Ciclismo<br />
<input type="radio" name="atividade" value="4" />   Hidrogin&aacute;stica </td>
```

Figura B.16 - botões de rádio para as atividades
Fonte: do autor

c) um menu de lista para o tempo (*id=tempo*)

```
<select id="tempo" name="tempo">
  <option value="15">15</option>
  <option value="30">30</option>
  <option value="45">45</option>
  <option value="60">60</option>
  <option value="75">75</option>
  <option value="90">90</option>
  <option value="105">105</option>
  <option value="120">120</option>
  <option value="135">135</option>
  <option value="150">150</option>
  <option value="165">165</option>
  <option value="180">180</option>
  <option value="195">195</option>
  <option value="210">210</option>
  <option value="225">225</option>
  <option value="240">240</option>
  <option value="255">255</option>
  <option value="270">270</option>
  <option value="285">285</option>
  <option value="300">300</option>
</select>  minutos
```

Figura B.17 - Menu de lista para o tempo
Fonte: do autor

d) uma div para o resultado (id=resultado)

e) um botão (id=btnCalcular)

A parte de programação JavaScript será feita em um arquivo separado (js.js). Esta é a forma recomendada de incluirmos nossa funções JavaScript no HTML, uma vez que geralmente temos várias funções que são utilizadas nas nossas páginas. O nome da nossa função será calcula() e será chamada no evento onClick do botão Calcular.

Antes de partirmos para a programação JavaScript vamos analisar todo o código HTML que é mostrado abaixo em duas figuras.

A figura B.18 mostra o código do arquivo *exercicio.html* da linha 1 até a linha 30.

```
1  <html>
2  <head>
3     <meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
4     <title>Exerc&iacute;cio passo a passo</title>
5     <script language="javascript" src="js.js"></script>
6  </head>
7  <body>
8     <form id="form1" name="form1" method="POST" action="">
9     <table width="50%" border="0" align="center" cellpadding="0" cellspacing="10">
10       <tr>
11         <td colspan="2" align="center"><strong>Calcule, de acordo com a atividade f&iacute;sica, seu peso e
   tempo de exerc&iacute;cio,<br>quantas calorias voc&ecirc; queima praticando esporte</strong></td>
12       </tr>
13       <tr>
14         <td width="120" align="right"><b>Peso</b></td>
15         <td><input type="text" id="peso" name="peso" /> q
16       </tr>
17       <tr>
18         <td width="120" align="right"><b>Atividade f&iacu
19         <td>
20              <input type="radio" name="atividade" value="3" /> V&ocirc;lei <br />
21              <input type="radio" name="atividade" value="5.5" />Dan&ccedil;a de sal&atilde;o<br />
22              <input type="radio" name="atividade" value="6" />  Esgrima<br />
23              <input type="radio" name="atividade" value="6" />  T&ecirc;nis<br />
24              <input type="radio" name="atividade" value="8" />  Basquete<br />
25              <input type="radio" name="atividade" value="10" /> Jud&ocirc;<br />
26              <input type="radio" name="atividade" value="3" />  Surfe<br />
27              <input type="radio" name="atividade" value="3.5" /> Remo<br />
28              <input type="radio" name="atividade" value="4" />   Ciclismo<br />
29              <input type="radio" name="atividade" value="4" />   Hidrogin&aacute;stica </td>
30       </tr>
```

Observe a referência ao arquivo js.js

Observe que cada atividade possui um valor correspondente

Figura B.18 - Código do arquivo HTML - parte 1
Fonte: do autor

Fomento ao Uso das Tecnologias da Informação e Comunicação

A figura B.19 apresenta a código do arquivo HTML das linhas 31 à 60.

```
31          <tr>
32           <td width="120" align="right"><b>Tempo</b></td>
33           <td>
34              <select id="tempo" name="tempo">
35                <option value="15">15</option>
36                <option value="45">45</option>
37                <option value="75">75</option>
38                <option value="105">105</option>
39                <option value="135">135</option>
40                <option value="165">165</option>
41                <option value="195">195</option>
42                <option value="225">225</option>
43                <option value="255">255</option>
44                <option value="285">285</option>
45              </select>  minutos
46              </td>
47          </tr>
48          <tr>
49            <td align="right"> </td>
50            <td><strong>**F&oacute;rmula: peso * tempo * (valor da atividade f&iacute;sica) * 0.0175<br />
    Testar se os valores foram informados <br /> </strong></td>
51          </tr>
52          <tr>   <td colspan="2" align="right"><div id="resultado"></div></td>    </tr>
53          <tr>
54            <td colspan="2" align="center"><input name="btnCacular" type="button" id="btnCacular"
55              onclick="calcula()" value="Ver resultado"/></td>
56          </tr>
57        </table>
58      </form>
59  </body>
```


Figura B.19 - Código do arquivo HTML - parte 2
Fonte: do autor

Tendo nosso arquivo *exercicio.html* completo, vamos à codificação JavaScript. A figura B.20 apresenta a função *calcula()* do arquivo js.js.

```
2   function calcula(){
3
4       peso  = document.getElementById('peso').value;
5       tempo = document.getElementById('tempo').value;
6       atividade   = document.getElementsByName('atividade');
7       valor_atividade = 0;
8
9       mensagem = "";
10      if (peso == "")
11          mensagem = "Informe o peso";
12
13      if (tempo == "")
14          mensagem += "<br /> Informe o tempo";
15
16      flag = false;
17      for(j=0;j<atividade.length;j++){
18          if (atividade[j].checked){
19              flag = true;
20              valor_atividade = parseFloat(atividade[j].value);
21          }
22      }
23
24      if (flag == false)
25          mensagem += "<br /> Selecione a atividade";
26
27      if (mensagem == ""){
28          calorias = parseFloat(peso) * parseInt(tempo) * valor_atividade * 0.0175;
29          mensagem = calorias.toFixed(2) + " calorias gastas ";
30      }
31
32      document.getElementById('resultado').innerHTML= mensagem;
33  }
```

Figura B.20 - função calcula() do arquivo js.js
Fonte: do autor

Agora vamos analisar todo o código da função:

Linha 2: Definição da função (não recebe parâmetros).

Linhas 4 e 5: Pega os valores dos elementos cujo *id* são respectivamente peso e tempo.

Atenção:

O método *getElementById* já foi visto na parte 1 desta unidade sobre JavaScript. Este método retorna o objeto de qualquer elemento do documento que tenha o *id* passado por parâmetro.

Linha 6: Veja que para pegar a atividade foi utilizado um método diferente (*document.getElementsByName*), isso porque existem vários elementos do tipo rádio com o mesmo name (neste caso atividade - quando um item é selecionado pelo usuário os demais itens com o mesmo nome são desmarcados). O método

document.getElementsByName retorna um *array*, contendo todos os elementos que possuem o nome passado por parâmetro. Nesse caso, vamos precisar de um laço para percorrer todo o *array*.

Linha 7: Inicia variável *valor_atividade* com zero. Essa variável será utilizada para pegar o valor correspondente à atividade selecionada.

Linhas 9 a 14: Inicialmente é criada uma variável *mensagem* sem valor. Nessas linhas, é verificado se foram informados valores para o peso e tempo. Caso não tenha sido informado, é atribuída mensagem para a variável *mensagem*. Observe que na linha 14 é feita uma concatenação (+=). Isso significa que a variável mensagem mantém o seu valor e concatena com o novo valor passado.

Linhas 16 a 22: Este conjunto de código tem por objetivo verificar se alguma atividade foi selecionada e neste caso pegar o valor correspondente.

- Linha 16: cria variável *flag* com valor *false*.
- Linha 17: laço *for* que percorre todas as opções de atividades (*atividade.length* retorna o número de opções, neste caso 10).
- Linha 18: verifica a propriedade *checked* da atividade. Se for *true* significa que está marcada.
- Linha 19: atribui *true* para variável *flag*. Significa que tem atividade selecionada.
- Linha 20: pega o valor correspondente à atividade. A função *parseFloat* converte o valor que é string para o tipo *float* necessário para fazer o cálculo.

Linhas 24 e 25: Verifica o valor de *flag*. Se continuar com *false* atribui uma mensagem.

Linha 27: Verifica se mensagem possui valor. Caso não possua valor procede ao cálculo.

Linha 28: Realiza o cálculo. Utiliza as funções *parseFloat* e parseInt para converter os valores respectivamente para *Float* e *Int* (necessário para realizar o cálculo).

Linha 29: Atribui uma mensagem com o resultado do cálculo. A função *toFixed(2) (calorias.toFixed(2)*) faz com que o valor apresente apenas 2 casas decimais.

Linha 32: Atribui o valor da variável *mensagem* para a *div resultado.*

Nossa solução está completa. Lembre-se de que um problema pode ter diferentes formas de solução. Aqui apresentamos uma solução que busca mostrar de forma didática cada passo realizado para a solução do problema.

Atividades – Parte 3

1. Crie duas páginas (X)HTML como mostrado na figura abaixo. A primeira página deve ter um botão "Abre Janela" que, quando clicado, deve abrir uma nova janela nas dimensões de 360 de largura x 280 de altura (pixels). Depois de aberta, a nova janela deverá estar na frente da antiga (use *focus*()). A segunda página deve conter um botão fechar que, ao ser ativado, fecha a janela.

Fomento ao Uso das Tecnologias da Informação e Comunicação

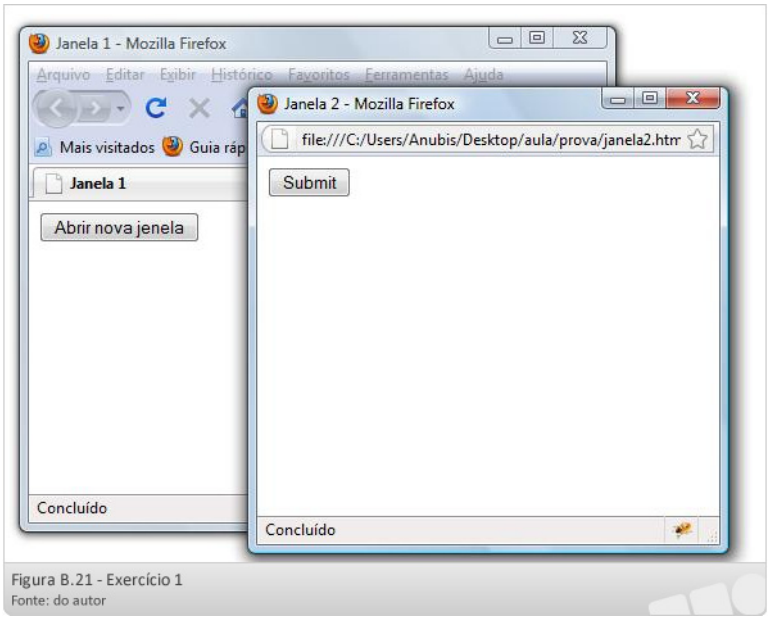


Figura B.21 - Exercício 1
Fonte: do autor

2. Considerando o código (X)HTML do formulário cujo código é disponibilizado abaixo, faça:

a) Validar se todas as perguntas foram respondidas. Caso haja perguntas não respondidas mostrar na div "resultado";

b) Calcular a pontuação de acordo com o valor de cada opção e mostrar na caixa "*pontos*".

c) Mostrar na div "*resultado*" o diagnóstico de risco conforme pontuação calculada (conforme abaixo).

Risco baixo (inferior a 6)
Com pouca probabilidade de sofrer doenças do coração na próxima década, se for preciso, faça algumas mudanças em seu estilo de vida para manter o seu risco cardíaco baixo. Se você fuma, abandone o cigarro. Se é sedentário, engaje-se numa rotina de exercícios físicos. Tenha uma dieta equilibrada e mantenha-se no peso ideal

Risco moderado (entre 6 e 8)
Ainda que você não sinta nada, é grande a probabilidade de que suas artérias já acumulem placas moles de gordura. O risco de você vir a ser acometido por males cardíacos nos próximos dez anos pode ser minimizado com alterações em seus hábitos de vida. Talvez o seu médico recomende que você tome um comprimido de aspirina diariamente. Se o nível de LDL for igual ou superior a 130, é o caso de recorrer às estatinas, os medicamentos contra o colesterol alto

Risco alto (superior a 8)
É certo que o médico adotará medidas preventivas mais agressivas. Além da aspirina, ele poderá indicar o uso de estatinas, os remédios contra o colesterol alto, e medicamentos para baixar a pressão arterial. A adoção de hábitos saudáveis de vida é urgente

Figura B.22 - Diagnóstico de Risco
Fonte: do autor

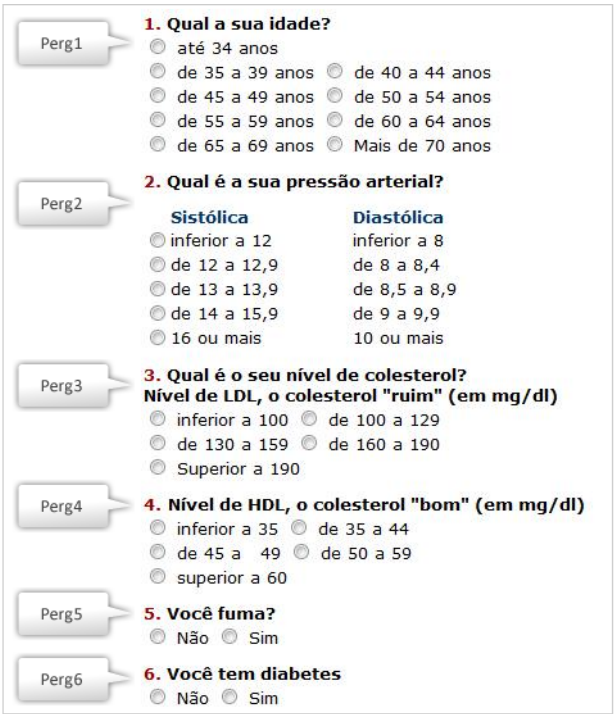


Figura B.23 - Visualização do formulário
Fonte: do autor

Segue o código (X)HTML do formulário acima:

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1" />
<title>PESQUISA</title>
<script language="javascript" src="js.js"></script>
</head>
<body>
<form id="form1" name="form1" method="POST" action="">
  <table width="50%" border="0" align="center" cellpadding="0" cellspacing="3"
bgcolor="#FFFFFF">
    <tr>
      <td align="center"><strong>QUESTION&Aacute;RIO</strong></td>
    </tr>
    <tr>
      <td><p><strong>1.Qual a sua idade?</strong><br />
        <input name="perg1" type="radio" value="-1" />        at&eacute; 34
anos<br />
        <input name="perg1" type="radio" value="0" />        de 35 a 39 anos
        <input name="perg1" type="radio" value="1" />        de 40 a 44
anos<br />
        <input name="perg1" type="radio" value="2" />        de 45 a 49 anos
        <input name="perg1" type="radio" value="3" />        de 50 a 54
anos<br />
        <input name="perg1" type="radio" value="4" />        de 55 a 59 anos
        <input name="perg1" type="radio" value="5" />        de 60 a 64
anos<br />
        <input name="perg1" type="radio" value="6" />        de 65 a 69 anos
        <input name="perg1" type="radio" value="7" />        Mais de 70 anos
</p>

        <p><strong>2.Qual &eacute; a sua press&atilde;o arterial
(sist&oacute;lica / diast&oacute;lica )?
        </strong></p>
```

Fomento ao Uso das Tecnologias da Informação e Comunicação

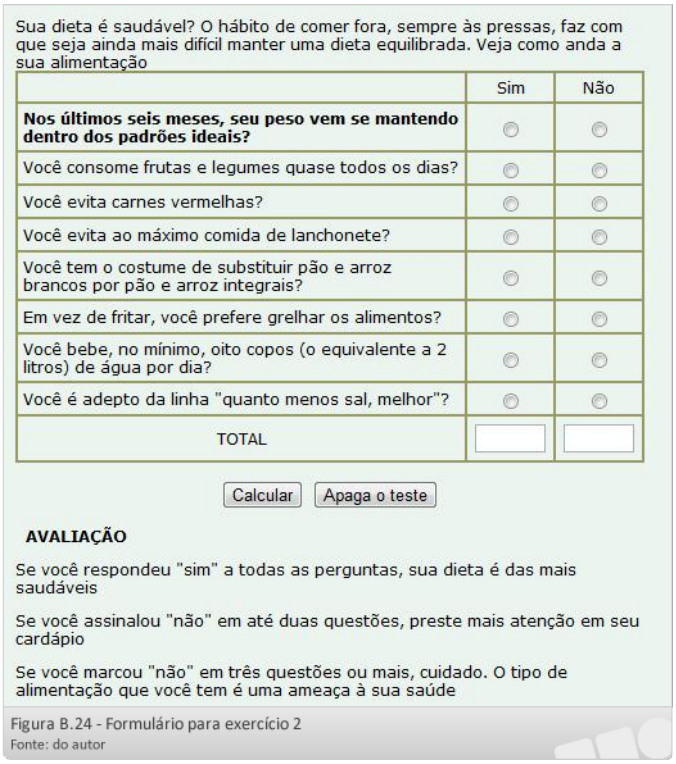
```
<input name=”perg2” type=”radio” value=”0” /> inferior  a 12 - inferior a
8<br>
<input name=”perg2” type=”radio” value=”0” /> de 12 a 12,9 - de 8 a 8,4<br>
<input name=”perg2” type=”radio” value=”1” /> de 13 a 13,9 - de  8,5 a 8,9<br>
<input name=”perg2” type=”radio” value=”2” />de 14 a 15,9 - de  9 a 9,9<br>
<input name=”perg2” type=”radio” value=”3” />  16 ou mais- 10 ou mais


<p> <strong>3. Qual &eacute; o seu n&iacute;vel de colesterol? </strong><br />
  <strong>N&iacute;vel de LDL, o colesterol &quot;ruim&quot; (em mg/dl)</
strong><br />

          <input name=”perg3” type=”radio” value=”-3” />          inferior a
100
          <input name=”perg3” type=”radio” value=”0” />          de 100 a
129<br />
          <input name=”perg3” type=”radio” value=”0” />          de 130 a 159
          <input name=”perg3” type=”radio” value=”1” />          de 160 a
190<br />
          <input name=”perg3” type=”radio” value=”2” />          Superior a
190</p>

<p><strong>4.N&iacute;vel de HDL, o colesterol &quot;bom&quot; (em mg/dl)</
strong><br />
          <input name=”perg4” type=”radio” value=”2” />          inferior a 35
          <input name=”perg4” type=”radio” value=”1” />          de 35 a 44<br
/>
          <input name=”perg4” type=”radio” value=”0” />          de 45 a
  49
          <input name=”perg4” type=”radio” value=”0” />          de 50 a 59<br
/>
          <input name=”perg4” type=”radio” value=”-1”/>          superior a
60</p>

<p><strong>5. Voc&ecirc; fuma?      <br>   </strong>
        <input name=”perg5” type=”radio” value=”0”/>  N&atilde;o
        <input name=”perg5” type=”radio” value=”2”/>    Sim </p>

<p><strong>6. Voc&ecirc; tem diabetes</strong><br />
        <input name=”perg6” type=”radio” value=”0” />   N&atilde;o
        <input name=”perg6” type=”radio” value=”2” />     Sim</p></td>
    </tr>
    <tr>
      <td align=”center”> </td>
    </tr>
    <tr>
      <td align=”center”><label for=”pontos”>Pontos</label>
        <input name=”pontos” type=”text” id=”pontos” size=”4” />  <br /> <br
/>
        <input name=”button” type=”button” id=”button” value=”Ver
resultado”/></td>
  </table>
</form>
</body>
</html>
```

3. Considerando o formulário abaixo, faça:

a) Validar se todas as perguntas foram respondidas. Caso haja perguntas não respondidas mostrar mensagem em uma .

b) Mostrar o total de respostas Sim.

c) Mostrar o total de respostas Não.


Sua dieta é saudável? O hábito de comer fora, sempre às pressas, faz com que seja ainda mais difícil manter uma dieta equilibrada. Veja como anda a sua alimentação

	Sim	Não
Nos últimos seis meses, seu peso vem se mantendo dentro dos padrões ideais?	○	○
Você consome frutas e legumes quase todos os dias?	○	○
Você evita carnes vermelhas?	○	○
Você evita ao máximo comida de lanchonete?	○	○
Você tem o costume de substituir pão e arroz brancos por pão e arroz integrais?	○	○
Em vez de fritar, você prefere grelhar os alimentos?	○	○
Você bebe, no mínimo, oito copos (o equivalente a 2 litros) de água por dia?	○	○
Você é adepto da linha "quanto menos sal, melhor"?	○	○
TOTAL		

Calcular | Apaga o teste

AVALIAÇÃO

Se você respondeu "sim" a todas as perguntas, sua dieta é das mais saudáveis

Se você assinalou "não" em até duas questões, preste mais atenção em seu cardápio

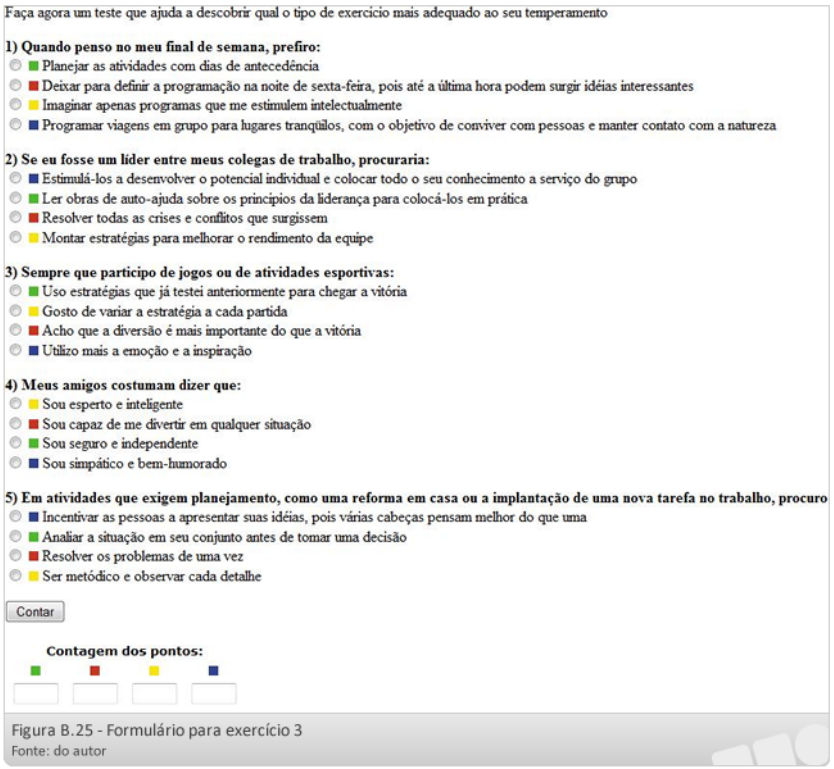
Se você marcou "não" em três questões ou mais, cuidado. O tipo de alimentação que você tem é uma ameaça à sua saúde

Figura B.24 - Formulário para exercício 2
Fonte: do autor

4. Considerando o formulário abaixo, faça a codificação JavaScript para:

a) Validar se todas as perguntas foram respondidas. Caso haja perguntas não respondidas mostrar mensagem em uma *div*.

b) Mostrar na caixa de texto correspondente o número de respostas para cada cor.


Faça agora um teste que ajuda a descobrir qual o tipo de exercício mais adequado ao seu temperamento

1) Quando penso no meu final de semana, prefiro:
- ○ Planejar as atividades com dias de antecedência
- ○ Deixar para definir a programação na noite de sexta-feira, pois até a última hora podem surgir idéias interessantes
- ○ Imaginar apenas programas que me estimulem intelectualmente
- ○ Programar viagens em grupo para lugares tranqüilos, com o objetivo de conviver com pessoas e manter contato com a natureza

2) Se eu fosse um líder entre meus colegas de trabalho, procuraria:
- ○ Estimulá-los a desenvolver o potencial individual e colocar todo o seu conhecimento a serviço do grupo
- ○ Ler obras de auto-ajuda sobre os princípios da liderança para colocá-los em prática
- ○ Resolver todas as crises e conflitos que surgissem
- ○ Montar estratégias para melhorar o rendimento da equipe

3) Sempre que participo de jogos ou de atividades esportivas:
- ○ Uso estratégias que já testei anteriormente para chegar a vitória
- ○ Gosto de variar a estratégia a cada partida
- ○ Acho que a diversão é mais importante do que a vitória
- ○ Utilizo mais a emoção e a inspiração

4) Meus amigos costumam dizer que:
- ○ Sou esperto e inteligente
- ○ Sou capaz de me divertir em qualquer situação
- ○ Sou seguro e independente
- ○ Sou simpático e bem-humorado

5) Em atividades que exigem planejamento, como uma reforma em casa ou a implantação de uma nova tarefa no trabalho, procuro
- ○ Incentivar as pessoas a apresentar suas idéias, pois várias cabeças pensam melhor do que uma
- ○ Analisar a situação em seu conjunto antes de tomar uma decisão
- ○ Resolver os problemas de uma vez
- ○ Ser metódico e observar cada detalhe

Contar

Contagem dos pontos:

Figura B.25 - Formulário para exercício 3
Fonte: do autor