Manual de Desenvolvimento – Codificação de Interface

Informações sobre o documento	
Versão	1.0
Data da Versão	22/05/2012
Autor	João Guilherme Esteves Barbosa (joaoguilherme@bradigital.com.br) Gustavo Thiesen (gustavot@bradigital.com.br)
Revisor	Marcelo Bock (marcelo@bradigital.com.br)

Histórico de atualizações do documento			
Versão	Motivo da atualização	Data	Autor
1.0	*Descrição da alteração*	*Data qual foi feita a mudança*	*Responsavél pela alteração*

Sumário

1. Introdução

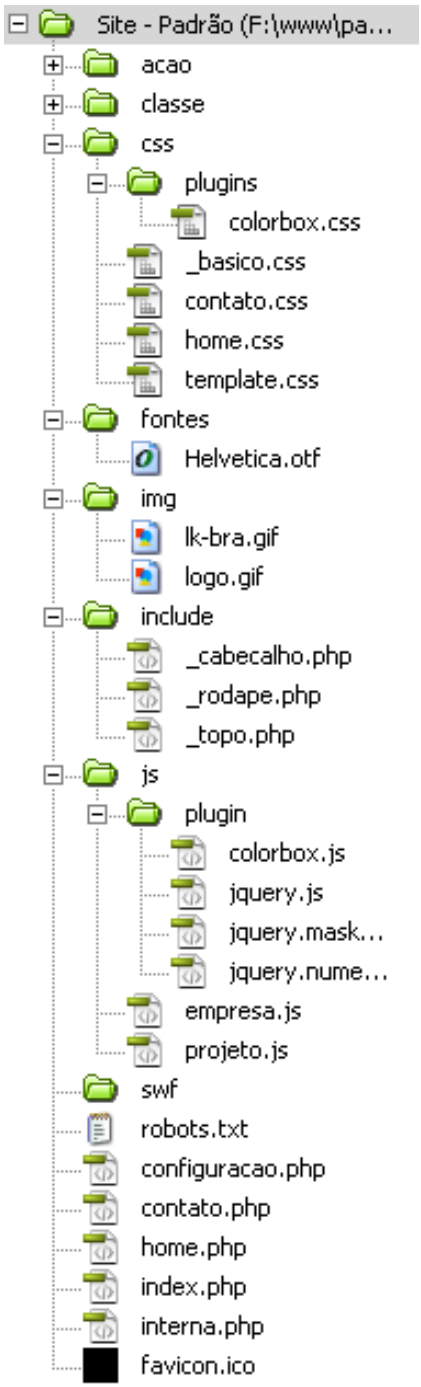
Para garantir um desenvolvimento e manutenção similar nos mais variados projetos – onde pode haver trabalho de mais de um desenvolvedor - foi criado esse documento. Ele possui regras, nomenclaturas e métodos que devem ser seguidos para que os projetos desenvolvidos pela BRA estejam sempre no mesmo padrão.

Todas as matérias descritas nesse documento estão abertas a discussões, para que se tenha uma evolução e melhoria constante dos padrões de desenvolvimento.

2. Estrutura de diretórios e nomenclatura de arquivos

Estrutura de diretórios

Estrutura Básica:



Descrição:

acao: Pasta de camadas de programação

classe: Pasta de classes de programação

css: Arquivos CSS que são utilizados no projeto. Ex.: default.css, template.css, empresa.css, etc.

O arquivo *template.css* é o arquivo onde deve ter todos os estilos usado na estrutura do site e estilos que são abrangente em mais de uma seção. Nesse arquivo também deve conter um "reset" do css. Haverá uma biblioteca de estilos para facilitar o desenvolvimento.

Arquivos que tiverem nome de alguma seção devem conter apenas estilos daquela seção. Estes arquivos devem ser chamados apenas nessas especificas páginas, para evitar requisições desnecessárias.

Dentro da pasta *plugins*, ficam arquivos CSS que são usados em plugins adicionados aos sites. Ex.: colorbox.css

fontes: Arquivos de fontes embedadas no html

img: Imagens que são utilizadas no projeto. Ex.: bg-site.jpg, lk-empresa.gif

include: Pasta na qual deve conter todos os arquivos com uma estrutura comum mais de uma página. Ex.: menu-topo.php (arquivo que contém o menu que é comum em todo o site)

js: Arquivos de javascript que são utilizados no projeto. Ex.: default.js, jquery.js, empresa.js, etc.

O arquivo *projeto.js* é o arquivo onde deve ter todas as funções usadas para o template do site e funções que são abrangente em mais de uma seção. Nesse arquivo haverá uma biblioteca de funções para facilitar o desenvolvimento.

Arquivos que tiverem nome de alguma sessão devem conter apenas estilos daquela seção. Estes arquivos devem ser chamados apenas nessas especificas páginas, para evitar requisições desnecessárias.

Dentro da pasta *plugins*, ficam arquivos JS dos plugins adicionados ao projeto. Ex.: colobox.js

swf: Arquivos SWF que são utilizados no projeto.

OBS1: Nenhuma das pastas é obrigada a existir, ela só irá existir se houver necessidade, ou seja, se existir algum arquivo que vá ali dentro, caso contrario a mesma deverá ser excluída. Ex.: se existir arquivos .pdf no site criar pasta pdf na raiz da pasta site.

Nomenclatura de arquivos

Arquivos HTML, ASP, PHP, JS e etc.

- Os arquivos devem sempre ser escritos com significado em português e nunca em outro idioma. Ex: menu-topo.php, menu-base.php, botoes.php
- Não utilizar nomenclatura com significado em outro idioma. Ex: menu-top.php, button-centro.png,img-bottom-centro.jpg;
- Não utilizar nomenclatura com números. Ex: 1.1.1.html;
- Não utilizar nomenclatura com acentuação ou espaços. Ex: ação.html;
- Todos os nomes dos arquivos devem ser em minúsculo;
- Para nomes compostos eles deverão ser separados por hífen e não por underline. Ex: empresa-contato.php
- Não utilizar nomes com acentos, e caracteres especiais.
- Os arquivos das seções deverão ter um nome significativo, pois este nome também é utilizado para indexação de conteúdos em buscadores. Ex: (missao.php, valores.php, calendario.php, eventos.php, etc). Se possível deve seguir o mapa do site.

SWF, PDF e DOC´s

- Deverão ter um nome significativo seguindo os padrões acima.

Imagens

A nomenclatura de imagens deverá seguir o padrão abaixo:

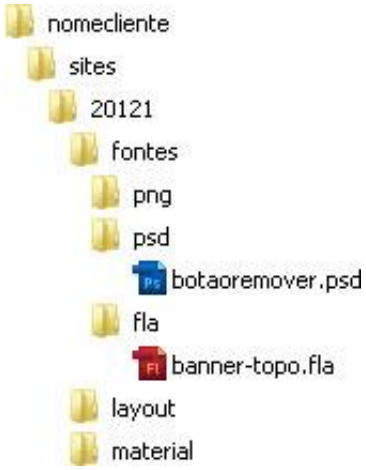
- bt-nome.gif - botão
- ic-nome.gif - ícones
- lk-nome.gif - links, lembrando que sempre devemos utilizar image-replacement.
- tt-nome.gif – usado para títulos, lembrando que sempre devemos utilizar image-replacement.
- tx-nome.gif - imagens que são apenas textos
- bg-nome.jpg - imagens de fundo
- ln-nome.gif - linhas ou fios

Arquivos fonte de layout

Para facilitar posteriores alterações nas imagens e/ou mídias do site, os arquivos utilizados para produzir as imagens/mídias do site devem ser salvos sempre em uma pasta chamada fontes (de arquivos fonte) dentro da pasta do projeto em questão. Ex: Arquivos png com texto editável, arquivo psd com texto editável, arquivos flash de banners, etc conforme exemplo:

Os arquivos utilizados para gerar imagens (jpg, gif ou png) que serão utilizadas no Html deverão ser salvos na pasta "fontes" dentro do projeto do cliente na unidade F:

(dados2) conforme a estrutura básica abaixo:



3. Padrão de desenvolvimento

Doctype

O doctype a ser utilizado para projetos da BRA é o doctype para XHTML Strict conforme abaixo:

```
<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Strict//EN"
"http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd">
<html xmlns="http://www.w3.org/1999/xhtml">
```

Indentação do código

O recurso de Indentação de códigos não é um recurso de linguagem, mas sim uma convenção de escrita de códigos que auxilia na escrita de códigos. A Indentação é o espaçamento antes de começar a escrever o código na linha e também pode ser chamado de tabulação.

O padrão é utilizar um tab e não utilizar 4 espaços em branco, como alguns editores utilizam.

Nomenclatura de classes e ids de objetos

Será utilizada a denominação CamelCase (denominação em inglês para a prática de escrever palavras compostas ou frases).
Nomes de classes e ids de css de objetos serão sempre escritos em lowerCamelCase (lowerCamelCase são CamelCase iniciados por letras minúsculas).

Não se deve economizar letras e palavras na hora de dar nome as classes ou ids. Use sempre o necessário para ser entendido por outros desenvolvedores

Ex.:

```
<div id="menuTopo">
    <ul class="classeEspecial">
        <li class="itemEspecial">Home</li>
    </ul>
</div>
```

OBS: Os nomes devem ter significado em português. Ex: menuTopo, menuBase, rodapeInternas.

Imagens

- Quando utilizar
    - Gif

Gifs serão utilizados quando a imagem não for uma foto. Quando as cores da imagem forem chapadas e/ou com transparências sem opacidade. Utilizar o formato exact na exportação. Caso alguma mensagem seja exibida avisando que não serão exportadas todas as cores, utilizar jpeg para exportar.

- Jpeg

  Jpegs serão utilizados quando a imagem for uma foto (salvo exceções de alguns gifs). Serão exportados sempre com compressão 80% no formato de cor RGB e com 76dpi de resolução.

- Png

  Pngs serão utilizados quando a imagem possuir transparências com opacidade. utilizar png 32 ou 24.

CSS

- Evitar usar CSS inline em elementos ou usar a tag <style> nos arquivos de interface.
- Técnicas
  - Image-replacement

    É uma técnica de substituir um texto por imagens com o objetivo de manter o texto "visível" ao browser, mas escondido do usuário. Os buscadores indexam os textos do image-replacement.

  - CSS Sprite

    O CSS Sprite é uma técnica que usa um "Sprite" para fazer uma “troca de imagem", ou seja, o deslocamento de posição do mapa de imagem que irá fazer, no caso, a "animação". Essa técnica permite que lizemos várias imagens em uma só, assim economizamos número de requisições no servidor e quantidade de imagens.

- Formatação
  - As propriedades do CSS devem ser escritas uma abaixo da outra.
  - Em casos onde houver somente uma regra de CSS, não é necessário endentar Ex.:

```
#principal{
    width:500px;
    height:500px;
    float:left;
}
lk-destaque{ color:#00000; }
```

- Comentário

  Sempre que possível, utilizar os comentários para denominar os blocos principais de

código e especificar classes.
Deve ser escrito em letras minúsculas e com espaços para separar palavras compostas.
Utilizar a hierarquia de comentário mostrada no exemplo abaixo

```
/*----------------------
    BOTOES
----------------------*/

/*--- BOTOES ---*/

/*-- botoes --*/

/*- botoes -*/

/* botoes */
```

HTML

- Cometário
  Sempre que possível, utilizar os comentários para denominar os blocos principais de código e especificar classes.
  Deve ser escrito em letras minúsculas e com espaços para separar palavras compostas.
  Deve haver um comentário antes de abrir o bloco e outro após o fechamento do mesmo.
  O comentário de fechamento deve conter uma barra como prefixo, assim seguindo a mesma lógica do fechamento de tag.
  Os comentários devem ser no mesmo nível das tags de abertura e fechamento do bloco.

```
<!-- imagem destaque -->
<div>…</div>
<!-- /imagem destaque -->
```

Embed de fontes

- Utilizar o gerador de *font-face h*ttp://www.fontsquirrel.com/

Javascript

- Evitar usar JS inline em elementos ou usar a tag <script> nos arquivos de interface.
- Framework padrão de javascript: JQuery (http://jquery.com/). O framework padrão de desenvolvimento deve conter sempre a última versão aprovada para uso.

- Plugins:

Janela Modal: ColorBox
Select customizado: jQuery custom selectboxes
Scroll customizado: jscrollpane
Banner customizado: Cycle
Mascaras de input: Masked Input Plugin
Carrosel: jCarousel

Formulários

Os nomes devem ser escritos com prefixo em letras minúsculas e uma ou mais palavras iniciadas com letra maiúscula escritas no singular.
Os atributos name e id são obrigatórios em todos os campos de formulário;
O atributo type é obrigatório em todos os campos que são input.

Segue lista dos elementos do formulário, com seus prefixos e exemplos de aplicações:

- Form (frm)

```
<form id="frmContato">
</form>
```

- Fieldset (fld)

```
<fieldset id="fldDadoPessoal">
</fieldset >
```

- Label (lbl)

```
<label for="txtNome">Nome:</label>
<input type="text" id="txtNome" name="txtNome" />
```

No atributo for deve ser colocado o mesmo valor do id do campo correspondente. Conforme exemplo.

- Password (pwd)

```
<input type="password" id="pwdSenha" name="pwdSenha" />
```

- Textarea (txa)

```
<textarea id="txaMensagem" name="txaMensagem"></textarea>
```

- Text (txt)

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

- Radio Button (rad)

```
<input type="radio" id="radMasculino" name="radSexo" />
<input type="radio" id="radFeminino" name="radSexo" />
```

O valor do atributo name deve ser o mesmo para os campos que pertencem ao mesmo grupo. O que os difere é o id. Conforme o exemplo.

- File (upl)

```
<input type="file" id="uplFoto" name="uplFoto" />
```

- select (sel)

```
<select name="selEstado" id="selEstado">
    <option>RS</option>
</select>
```

- checkbox (chk)

```
<input type="checkbox" id="chkEsporte" name="chkAssuntoInteresse[]" />
<input type="checkbox" id="chkTecnologia" name="chkAssuntoInteresse[]" />
```

O valor do atributo name deve ser o mesmo para os campos que pertencem ao mesmo grupo. O que os difere é o id. Conforme o exemplo.
No final do name devem ir colchetes, para facilitar posteriormente na programação.

- button, submit (btn)

```
<input type="submit" id="btnEnviar" name="btnEnviar" value="Enviar" />
```

Não esquecer de colocar o atributo value, que é o texto que aparece no botão

Extensões do Firefox mínimas para o desenvolvimento:

- Firebug - https://addons.mozilla.org/pt-BR/firefox/addon/1843
- Webdeveloper - https://addons.mozilla.org/pt-BR/firefox/addon/60
- IE Developer Toolbar para IE7 - http://www.microsoft.com/en-us/download/details.aspx?id=18359

4. Navegadores padrões

- Internet Explorer 7 e superiores;
- Firefox (Última versão);
- Chrome (Última versão);

Para conseguir testar com maior precisão, máquinas virtuais são utilizadas com navegadores ie7 e ie8 com sistema operacional windows xp. As maquinas virtuais estão localizadas na rede em F:\APPS\virtualpc e em F:\VMware Workstation. Os discos com a instalação do Windows xp já esta em cada pasta. Basta apenas copiar para a máquina e executar depois de instalado o software. Utilizar preferencialmente o vmware player.

5. Semântica

O HTML semântico tenta transmitir significado através das palavras e as tags em uma página. Tente pensar desta maneira: o conteúdo da página são as palavras que você fala. A marcação HTML fornece a estrutura, a entonação, as pausas e até mesmo a aparência do seu rosto. Basicamente, as marcações são a metade da sua mensagem.

- *<h1>,<h2>* - A tag de H1 a H6 definem quais são os títulos e sua relevância em um texto, sendo o H1 o mais relevante e o H6 o menos relevante.
- ** - A tag EM serve para dar ênfase em itálico, em alguma palavra ou trecho de um texto. Ao contrário da tag U que é somente visual.
- *<li>* - A tag LI, serve para definir uma lista de elementos. Sendo a tag OL para criar uma lista com elementos organizados por alguma ordem. E a UL organizando os elementos sem necessariamente possuírem uma ordem.
- *<p>* - Define o que é um parágrafo.
- ** - Define um destaque assim como a tag EM, mas no caso em negrito. Ao contrário da tag B que é somente visual.
- * ,* - S utilizados para estruturar os elementos, sendo o DIV para estruturar um bloco através do CSS e o SPAN uma linha.

Elementos de formulário já foram descritos acima.

6. Framework padrão de HTML

Atualmente a BRA disponibiliza um template padrão com a finalidade de facilitar o desenvolvedor. Nele já está com a estrutura de pasta padrões, assim como exemplo de todos outros padrões descrito nesse documento para facilitar o desenvolvimento de sites.

Nesse mesmo projeto vai haver uma biblioteca de estilos padrões no arquivo "*template.css*".

No arquivo "*projeto.js*" vai ter também uma biblioteca de funções Java script padrões.

Este projeto está no seguinte caminho:

F:\www\padrao.com.br\public_html\site

7. Checklist padrão

- Iniciar projetos utilizando o framework padrão de HTML;
- Código fonte deve conter comentários para entendimento de qualquer outro desenvolvedor;
- Utilizar include em trechos de códigos repetidos;
- Testar em todos navegadores padrões;
- Usar a nomenclatura correta para nome de classes e arquivos descrito nesse documento;
- Seguir a estrutura de pasta descrita nesse documento;
- Escrever ALT em todas as imagens, indicando a descrição da mesma em forma de texto;
- Validar o código pela w3.org (http://validator.w3.org/);
- Codificar sempre prevendo a escalabilidade e integração das páginas com ferramentas, possibilitando o acréscimo e retirada de conteúdo sem interferir no layout ou processo de renderização;
- Garantir a utilização do google analytics quando publicado o site;
- Testar o site com netlimiter para simular navegações com internet mais demoradas;
- Verificar se o projeto está com os devidos titles, metetags e description;
- Validar o site com a resolução de 1024px em todos navegadores padrões.