3 Além do HTML

3.1. Imagens

No meio eletrônico há basicamente dois tipos de imagens: as que são formadas por matriz de pixels (*bitmap*) e as *vetoriais*. As imagens do tipo bitmap (ou raster) são criadas a partir de uma matriz de pontos. A sua resolução é fixa, portanto, se uma imagem é criada em uma matriz de 100x100 pixels e depois ampliada para 1000x1000, cada ponto será dez vezes maior na versão ampliada. Imagens vetoriais são definidas através de equações e dados que permitem a sua reprodução. Uma imagem vetorial, quando ampliada, preserva a aparência da forma que representa. Os pontos na tela, que não fazem parte da informação armazenada com a imagem, não são ampliados como ocorre com as bitmaps, resultando em uma imagem mais nítida.

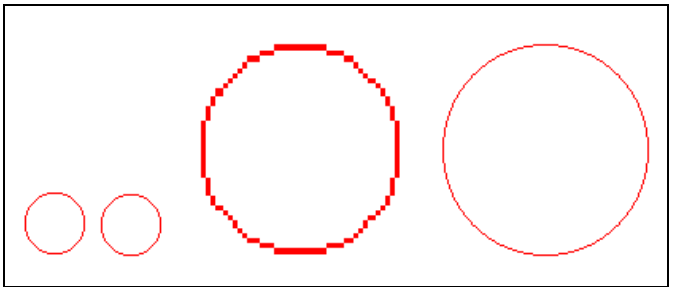


Figura 1 – O primeiro círculo é uma imagem bitmap. O segundo é uma imagem vetorial. O primeiro amplia o mapa de bits. O segundo amplia o raio da equação do círculo.

Existem softwares que trabalham exclusivamente com imagens vetoriais, chamados Line Art. Os mais populares são o *Corel Draw* (PC), *CorelXara*, *AutoCAD*, *Illustrator* (PC e Macintosh), *FreeHand* (Macintosh) e *Flash* (PC e Macintosh). O principal formato aberto utilizado é o *PostScript*. Imagens vetoriais são em geral *muito menores* que imagens bitmap, mas atualmente não são suportadas na World Wide Web a não ser através de extensões de browser como as extensões *Flash* e *Shockwave*..

O programa de criação e tratamento de imagens de bitmap mais popular é o *Adobe PhotoShop* (PC, Macintosh, Unix), que é praticamente o padrão de mercado. Seus concorrentes são *Corel PhotoPaint* (PC), o *Fractal Design Painter* (Macintosh), sharewares como o *LViewPro*, *PaintShopPro*, etc. Recentemente têm surgido pacotes criados especificamente para a Web como o *Adobe ImageReady*, o *Macromedia Fireworks* e o *Microsoft ImageComposer*. Esses pacotes são capazes de trabalhar com imagens vetoriais e gerar bitmaps otimizados para a Web.

Por serem muito grandes, *bitmaps sem compressão* raramente são usados na Web. Os formatos de compressão utilizados mais populares são *GIF* e *JPEG*. Na média, JPEG proporciona uma compressão maior que GIF. Mas em imagens pequenas, com poucas cores ou poucas variações, as imagens GIF tendem a ser menores.

3.1.1. Cores

A dimensão e o número de cores de uma imagem determina o seu tamanho. Imagens em preto-e-branco só precisam de um bit. Seus dois estados representam as cores desejadas: ligado

(branco) e desligado (preto). Com dois bits é possível representar quatro cores (como fazia o padrão de vídeo CGA) como preto (00), magenta (01), azul ciano (10) e branco (11). Cores adicionais podem ser produzidas através de dispersão, alternando pixels de cores diferentes e conseqüentemente, reduzindo a resolução. É possível construir cinza alternando bits pretos e brancos, mas a resolução será duas vezes menor que usando apenas bits cinza.

Quanto maior o tamanho de uma imagem em altura e largura, maior espaço será necessário para armazená-la na memória de vídeo e no disco. Uma imagem de um bit de profundidade com 100 pixels de largura por 100 pixels de altura ocupará um espaço de 100 x 100 = 10000 bits de informação. O armazenamento da imagem no computador poderá ocupar um espaço maior ainda, dependendo do formato utilizado.

Imagens coloridas tem mais bits por pixel. Se a imagem do exemplo anterior tivesse 4 cores (2 bits), o espaço necessário para armazená-la seria pelo menos duas vezes maior.

O número de cores que o seu sistema mostra na tela depende da quantidade de memória de vídeo disponível. Embora muitas máquinas mais sofisticadas têm capacidade de exibir imagens de 24 bits (16,777 milhões de cores), grande parte das máquinas populares só consegue exibir imagens de 8 bits, o que significa que seus usuários só podem ver 256 (2^8) cores na tela. Imagens de 24 bits são chamadas de *true color*, pois possuem 8 bits para representar cada cor, e conseguem exibir fotos de forma realística. Uma imagem *true color* com 100 pixels de largura por 100 pixels de altura ocupará um espaço de 100 x 100 x 24 = 240000 bits ou 29,4 kB, sem contar o *overhead* do formato de armazemanento (TIFF, BMP, etc.). A imagem terá esse tamanho mesmo que não use todas as cores.

Na Web, os tamanhos das imagens não aumentam desta forma quando se aumenta o número de cores, pois geralmente armazenadas em formatos de compressão como GIF e JPEG que reduz seu tamanho em um fator que varia 4 a 100 vezes.

3.1.2. Paletas de cores

Se uma imagem que foi criada com 16 milhões de cores (24 bits) for exibida em um ambiente gráfico de 8 bits, será necessário que o sistema realize conversões para que seja possível exibir a imagem. Sistemas de 256 cores são freqüentemente indexados (atribuem um número de 0 a 255 a cada cor que exibem). Esse conjunto de cores que são exibidas ao mesmo tempo é a paleta de cores. Ela pode ser escolhida de forma a permitir a melhor visualização possível. Por exemplo, se uma imagem é totalmente composta de tons de vermelho, pode-se usar uma paleta de 256 cores com predominância de tons avermelhados.

Mas as cores não servem apenas para a imagem. Elas têm que ser usadas também para colorir o ambiente gráfico do sistema operacional. Se uma imagem rouba todas as cores, o sistema operacional não será capaz de exibir as suas cores corretamente. Para evitar que uma aplicação roube todas as cores disponíveis, o número de cores que a aplicação pode usar poderá ser limitado.

Os browsers têm uma paleta própria chamada de *cubo de cores* para a exibição de imagens em monitores de 256 cores. O cubo consiste de 216 cores. As outras 40 são reservadas para o sistema. Os cantos do cubo são todas as oito possíveis combinações de 255 e 0:

- preto: 0,0,0;
- azul 0,0,255;
- verde 0,255,0;
- azul ciano 0,255,255;
- vermelho 255,0,0;
- magenta 255,0,255;
- amarelo 255,255,0 e
- branco 255,255,255

As cores internas do cubo são 4 cores intermediárias, uniformemente espaçadas, formando um total de 216. Quando uma imagem é carregada no browser, ela tem seus pixels adaptados à paleta do sistema operacional. Se a cor não existir, o sistema tentará criá-la por dispersão, alternando pixels. O browser automaticamente reduz as cores do browser (cores de fundo, textos, etc.) para cores da paleta.

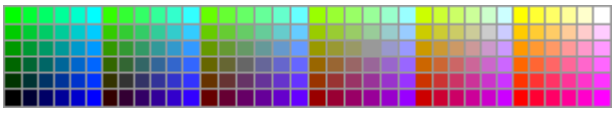


Figura 2 - Netscape Color Cube

Se você cria uma imagem em uma aplicação gráfica, sobre um fundo, digamos, azul, e esse azul não faz parte do cubo de cores, ele não terá uma cor correspondente no browser. Se você quiser que o fundo da imagem mescle de forma imperceptível com o fundo azul da página, você terá que escolher um azul que exista no cubo de cores, caso contrário, perceberá a diferença quando a imagem for carregada em um sistema de 256 cores. Utilizando-se as cores do "Color Cube", evita-se a dispersão incorreta das cores. A maior parte das aplicações gráficas possuem a paleta da Web.

3.1.3. Anti-aliasing

Anti-alising (ou anti-serrilha) é o processo que adiciona cores intermediárias para otimizar as áreas serrilhadas da imagem. O processo aumenta o número de cores da imagem, aumentando o tamanho da paleta (e conseqüentemente o tamanho da imagem). Se você abre o *PhotoShop* e faz um grande círculo de uma cor só, em um fundo uniforme, são grandes as chances de se ter mais de 200 cores! Além de aumentar o tamanho, *anti-aliasing* também reduz a capacidade de compressão da imagem.

Anti-aliasing é uma técnica necessária. Imagens com curvas ou linhas inclinadas sem *anti-aliasing* ficam com uma qualidade inaceitável. A solução é realizar o *anti-aliasing* com o menor número de cores possível.

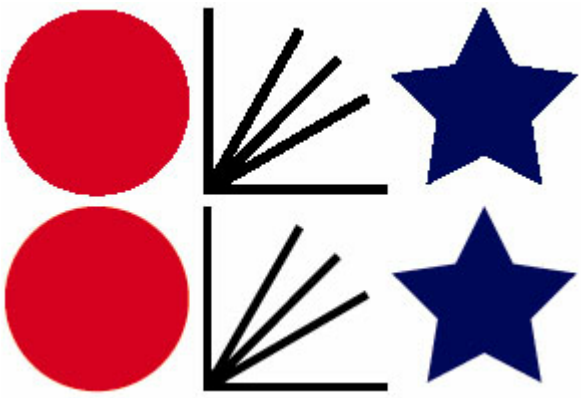


Figura 3 - Sem anti-serrilha (em cima) e com anti-serrilha (em baixo). Fonte: http://www.killersites.com

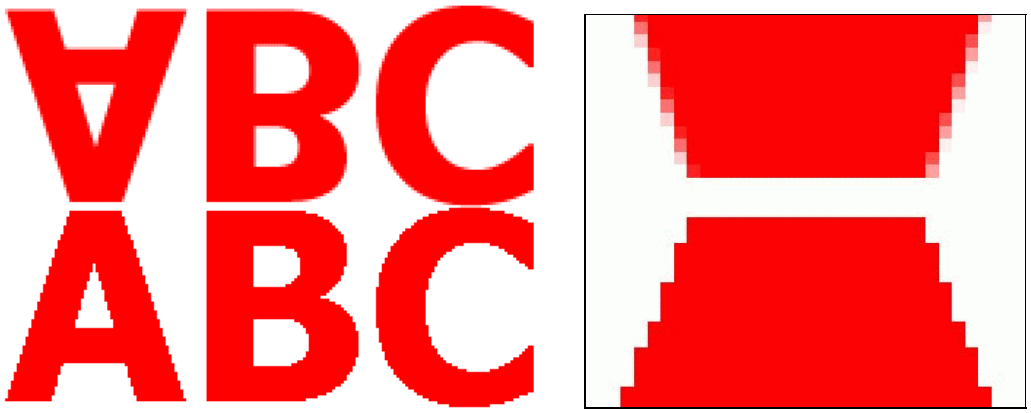


Figura 4 - Outro exemplo com anti-serrilha mostrando detalhe

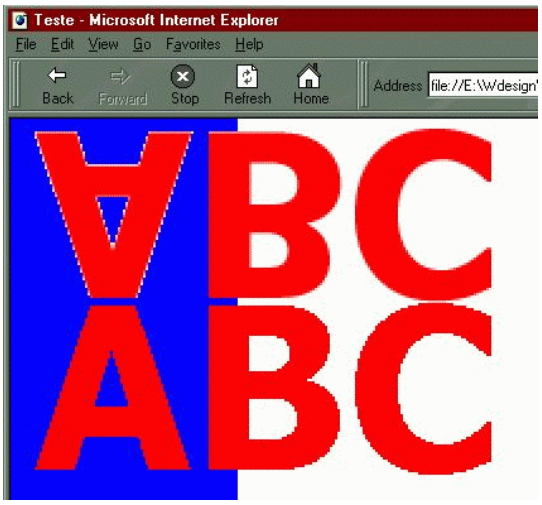


Figura 5 – Exemplo de "A" fantasma que aparece por trás do "A" que foi anti-serrilha.

Anti-aliasing pode causar problemas em imagens com fundo transparente. Quando for necessário por uma imagem na página combinando com a imagem de fundo é necessário criar a imagem, no aplicativo gráfico, sobre uma *réplica* da imagem de fundo. Isto garantirá que a variação de cores será feita corretamente. Como apenas uma cor pode ser escolhida como transparente, as cores intermediárias permanecerão na sua cor original. Se no PhotoShop o desenho for feito sobre o fundo branco, na hora de escolher a transparência, as cores próximas ao desenho permanecerão brancas e não transparentes, aparecendo, portanto, sobre um fundo de outra cor (veja figura).

Pode não haver solução para esse problema se a imagem de fundo for irregular. O *registro* (casamento da imagem de frente com a de fundo) deverá ser perfeito, mas isto é impossível

com os browsers atuais. Nesses casos, é preciso avaliar o que é pior: ficar sem o *anti-aliasing*, ou correr o risco de ter "fantasmas" em volta de partes da imagem. Se ambos forem inaceitáveis, será preciso repensar o projeto.

3.1.4. Compressão GIF

A compressão reduz bastante o tamanho da imagem. O primeiro formato comprimido suportado na Web foi o formato GIF ou *Graphics Interchange Format* . Trata-se de um formato de compressão de bitmaps pertencente à Unisys (SPRY/Compuserve). Suporta imagens indexadas até 8 bits (256 cores). Usa como método de compressão o algoritmo Lempel-Ziv-Welch (LZW) que proporciona uma compressão média da ordem de 4:1 sem perdas. A imagem parece idêntica à original.

Use GIF para comprimir qualquer coisa que não seja fotográfica como desenhos vetoriais, fotos pequenas, etc. Imagens fotográficas comprimem melhor em JPEG.

O algoritmo GIF consiste em substituir seqüências horizontais de pixels da mesma cor com um número indicando o tamanho da seqüência. Linhas horizontais idênticas comprimem linha por linha. Qualquer regularidade horizontal será comprimida. Desta forma, imagens que têm semelhanças horizontais são beneficiadas: uma imagem bitmap de um círculo, comprimido com GIF, tem quase o mesmo tamanho que a imagem GIF de sua metade superior.

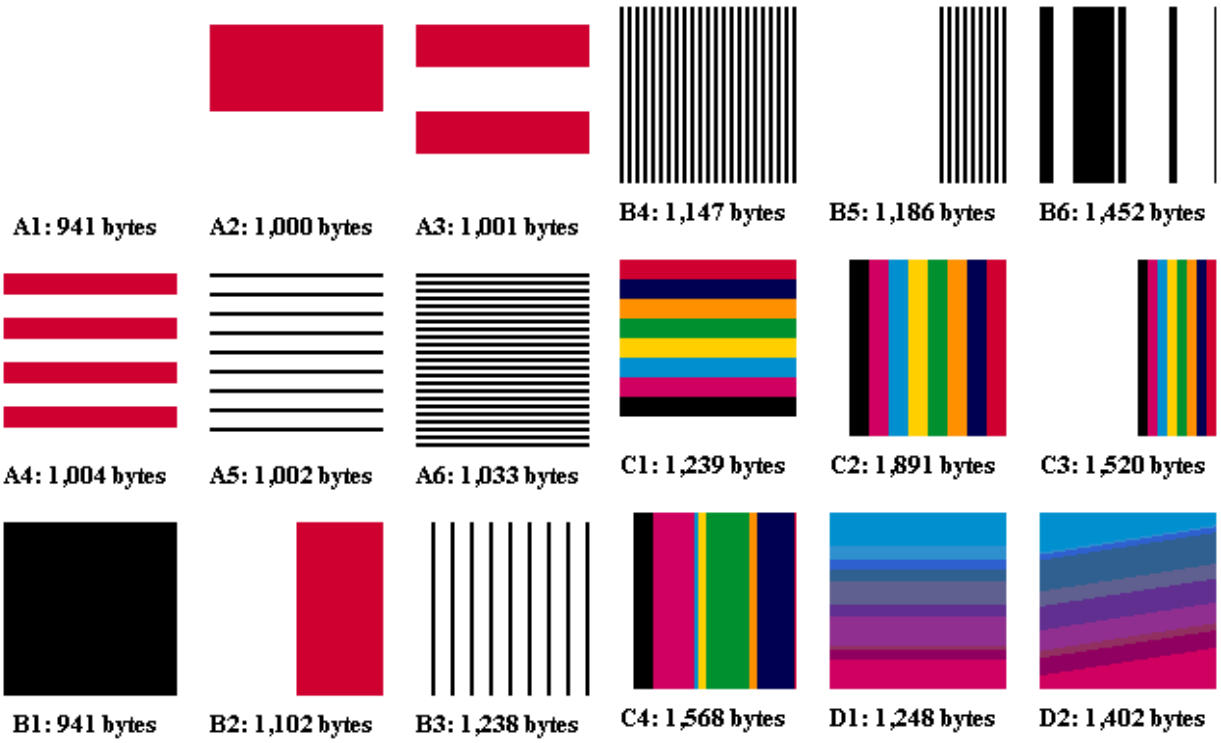


Figura 6 - Várias imagens gifs de mesma dimensão e conteúdo diferente (www.killersites.com)

Entrelaçamento e transparência

O formato GIF mais antigo em uso na Web é chamado de GIF87. O formato mais novo, GIF89, é mais eficiente e possui mais recursos como a capacidade de entrelaçamento, transparência e animação.

Normalmente, GIFs armazenam pixels de cima para baixo. Imagens entrelaçadas armazenam pixels fora de ordem linear. As imagens são transferidas totalmente em quatro passos. Na web, a exibição de imagens entrelaçadas costuma demorar mais que imagens comuns mas produz um efeito que aparenta uma velocidade maior.

Em GIF89, uma cor qualquer pode ser substituída pela cor transparente. Somente uma cor pode ser trocada.. Se houver uma variação pequena da cor escolhida como transparente, está continuará visível como antes. A transparência consiste em escolher o índice de uma cor e defini-la como transparente. A vantagem de imagens transparentes é que elas não são "retangulares", mas aparentam ter a forma do objeto que manteve as cores visíveis.

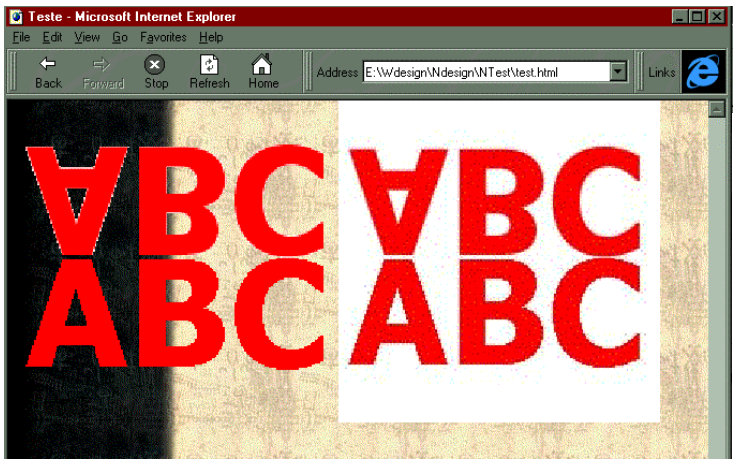


Figura 7 – Imagem transparente versus imagem não-transparente. O branco da primeira imagem foi trocado pelo transparente.

Animação

Várias imagens GIF podem ser armazenadas em um único arquivo para serem exibidas em seqüência. Com isto, é possível simular animação. GIFs animados são bastante versáteis e contém diversos efeitos diferentes: repetição eterna, tempos diferentes para cada imagem, etc.

Redução de cores e tamanho

Reduzir cores em GIF as torna menores. O processo deve ser realizado com cuidado para evitar uma redução excessiva que resulte em grande perda de qualidade. No *PhotoShop*, pode-se usar o *histograma de cores* e influenciar a redução de cores específicas.

Hoje já existem várias ferramentas que fazem automaticamente a redução de cores. Algumas ainda opinam sobre o melhor formato (GIF ou JPEG) para uma imagem.

3.1.5. JPEG

JPEG, ou *Joint Photographers Expert Group*, é a melhor forma de comprimir imagens fotográficas. Com JPEG, é possível obter uma compressão típica que varia de 10:1 a 100:1 dependendo da qualidade desejada.

Diferente de GIF, JPEG suporta o formato RGB e, portanto, pode armazenar informações de cor 24 bit (16,777 milhões de cores). JPEGs progressivas obtém o mesmo efeito que as GIFs entrelaçadas.

JPEG não garante a mesma compressão em imagens pequenas ou desenhos devido ao seu *overhead*. Veja os tamanhos de imagens de um pixel (1x1) de 4 cores em vários formatos:

JPEG:	949 bytes
BMP (sem compressão):	72 bytes
GIF:	41 bytes

Já uma imagem grande e detalhada (uma fotografia) comprime bastante bem em JPEG. Arquivo earthrise.jpg/gif/bmp (400x302 pixels; 256 cores) do CD do ASIT:

JPEG:	8,8 kB
GIF	32 kB
BMP	119 kB

Para reduzir o tamanho, JPEG separa as matizes de cor dos componentes de brilho e mantém uma boa cópia da versão em preto-e-branco da imagem. Joga fora diferenças de cor indistinguíveis. Divide a figura em zonas que são comprimidas individualmente.

A compressão é sempre realizada com perdas, mesmo quando na melhor qualidade. A imagem final, em geral, é menor que a equivalente GIF (desde que não seja muito pequena). Imagens mais nítidas são maiores que imagens mais borradas. A forma de reduzir o tamanho é borrar a imagem (cria mais cores intermediárias que são descartadas) diminuindo o seu fator de qualidade. JPEG não trabalha com cores indexadas (é inviável indexar 16 milhões de cores – o *overhead* seria imenso). Reduzir cores, portanto, não reduz o tamanho.

3.1.6. PNG

O PNG – *Portable Network Graphics* é um novo formato que foi desenvolvido pela W3C especialmente para a Web. É um formato que suporta compressão sem perdas, imagens de 24bits, efeitos progressivos (entrelaçamento) mais eficientes, transparência de vários níveis, canais alfa de 8 bits e uma série de vantagens que devem torna-la a substituição ideal para imagens GIF.

Vários programas já suportam PNG, mas não todos os seus recursos. Uma das vantagens não suportadas pelos browsers é a capacidade de exibir 256 níveis de transparência. Além dos browsers, os programas de desenvolvimento e tratamento de imagens como o *PhotoShop 4* também suportam e manipulam com PNG, que é o formato nativo do *Macromedia Fireworks*.

Veja mais informações sobre PNG em http://www.w3.org/PNG.

3.2. Layout com tabelas

Vimos na primeira parte deste curso os principais usos das tabelas HTML. Neste capítulo mostraremos exemplos de tabelas aplicadas à diagramação da página. Os recursos aqui apresentados são totalmente substituíveis por folhas de estilo, porém, a visualização da página poderá ser prejudicada seriamente se o browser não suportar CSS. Como as tabelas existem há mais tempo, é muito mais raro encontrar browsers que não suportam tabelas, embora os que não suportam posicionamento com folhas de estilo estão bastante próximos, nos browsers versão 3.0.

Para diagramar com tabelas é preciso saber como alinhar objetos dentro de tabelas, como combinar duas ou mais células em uma, remover os espaços entre células e entre a célula e o texto além de criar tabelas dentro de tabelas. As seções seguintes tratarão de alguns desses temas.

Por não ser uma linguagem apropriada à diagramação, HTML impõe diversas limitações quando à organização do texto e imagens em uma página. Como vimos na seção anterior, as tabelas oferecem algum controle. Um pouco mais controle é conseguido utilizando imagens invisíveis (pois a dimensão das imagens é certa enquanto a das tabelas não é). Mas há ainda várias outras limitações também referentes a colocação das imagens que só se resolvem com folhas de estilo. Em programas como o *PageMaker*, você coloca o texto e as imagens onde você quer. HTML sem folhas de estilo não oferece esse tipo de manipulação bidimensional. É necessário usar truques para simular este controle se você não puder depender de folhas de estilo.

3.2.1. GIF de um pixel

Este truque, proposto por David Siegel em seu livro *Criando Sites Arrasadores* consiste da utilização de uma imagem de um pixel (1x1) de cor transparente ou igual à cor de fundo da página. A imagem resultante é invisível e pode ser usada numa página HTML para "empurrar" textos, colunas e linhas de tabelas, imagens, etc.

A imagem pode ser esticada usando os atributos HEIGHT E WIDTH ou usar margens, através dos atributos HSPACE e VSPACE. Por exemplo:

```
<IMG SRC="pixel.gif" HEIGHT=11> ou
<IMG SRC="pixel.gif" VSPACE=5>      Espaço vertical de 11 pixels
```

```
<IMG SRC="pixel.gif" WIDTH=11> ou
<IMG SRC="pixel.gif" HSPACE=5>       Espaço horizontal de 11 pixels
```

A imagem alinha-se geralmente com a base do texto ou da outra imagem que está ao seu lado. Isto pode ser alterado através dos atributos ALIGN. É possível definir larguras mínimas precisas para tabelas criando uma linha <TR> que contenha blocos <TD> contendo, cada um, uma única imagem de 1 pixel de altura/largura pelo valor desejado de largura/altura.

3.2.2. Layout de página com tabelas

Tabelas sem bordas dentro de tabelas sem bordas podem ser usadas para realizar o layout de texto e imagens em uma página HTML. Com essa técnica pode-se construir um espelho de diagramação sofisticado e organizar texto e imagens de forma criativa, fugindo do estilo tradicional das páginas HTML.

Deve-se desligar as bordas da tabela e fazer o `cellspacing` e `cellpadding` iguais a zero. Use uma largura absoluta para a tabela e para as células para se ter maior controle. Havendo necessidade de espaço em branco, pode-se usar tanto `cellspacing` como `cellpadding`.

Imagens recortadas

Para colar imagens horizontalmente ou verticalmente a melhor opção, quando não se tem folhas de estilo, é usar tabelas. O controle é maior entre plataformas. A "fragmentação e colagem" das imagens é uma técnica que visa diminuir o tempo de transferência das imagens, se aproveitando do cache do browser, que não carrega imagens repetidas. A imagem é partida em pedaços de forma que as partes que mudam sejam imagens diferentes das partes que se mantém através das páginas.

Antes de dividir uma imagem pelas células de uma tabela, ela precisa ser recortada. Isto pode ser feito manualmente em um programa como o *Adobe PhotoShop* ou de forma automática no *Macromedia Fireworks*, por exemplo.

Ao acrescentar as imagens dentro das células, elas se acomodarão e ocuparão o espaço que for necessário. Se for preciso colocar duas imagens sobre outra ou imagens ao lado de uma imagem deve-se usar os atributos COLSPAN e ROWSPAN para rearrumar a tabela e permitir que as peças se encaixem. No final, deve-se desligar o espaço entre células, as margens e as bordas para que a imagem volte a ser uma só.

É preciso ter o cuidado de não deixar espaços nem quebras de linha dentro do bloco <TD> ... </TD> que contém a imagem . Se isto ocorrer podem acontecer situações como a página ao lado, onde a imagem não se recompõe totalmente.

3.3. Frames

Frames (quadros) são painéis que dividem a área de visualização do browser em subjanelas, cada uma capaz de exibir uma página diferente. Para mostrar duas páginas ao mesmo tempo dentro de uma janela do browser é preciso ter pelo menos três arquivos. Dois são os arquivos HTML das páginas que serão exibidas. O outro é uma página HTML que contém apenas as instruções para que o browser saiba se dividir em subjanelas.

Logo, para dividir uma janela em frames, é preciso criar essa página HTML que especifica as dimensões relativas ou absolutas das subjanelas em relação à janela que irá conter a página. Uma página de frames não é um *documento* HTML, pois não contém informação. Todo documento HTML, como sabemos, deve ter a forma:

```
<html>
  <head> ... </head>
  <body> ... </body>
</html>
```

O bloco <body> contém a informação da página. O bloco <head>, contém meta-informação, ou seja, informação sobre a página. Páginas que definem a estrutura de subjanelas (páginas de frames) têm uma estrutura diferente:

```
<html>
  <head> ... </head>
  <frameset atributos> ... </frameset>
</html>
```

e não podem conter blocos <body>[1].

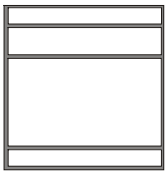
[1] Até podem conter blocos <BODY>, mas isto ou os transforma em páginas de informação, ou não causa efeito algum. Um bloco <BODY> antes do <FRAMESET> faz com que o browser ignore o <FRAMESET>. Um bloco <BODY>

3.3.1. Estrutura básica

O bloco `<frameset>` define a divisão da janela em *linhas* (usando o atributo `rows`) ou *colunas* (usando o atributo `cols`) ou simultaneamente em linhas e colunas (usando ambos os atributos). Os atributos especificam a largura ou altura de cada frame usando valores absoultos, em pixels, ou relativos, em percentagens da largura ou altura da janela principal. Por exemplo, um `<FRAMESET>` com a forma mostrada figura ao lado é definido pelo código

```
<FRAMESET COLS="25%,25%,50%"> ... </FRAMESET>
```

que divide a janela principal em três colunas, tendo as duas primeiras ¼ da largura total, e a última, metade da largura total. De forma semelhante pode-se dividir a janela em linhas. Neste outro exemplo (figura ao lado):



```
<FRAMESET ROWS="100,200,*,100"> ... </FRAMESET>
```

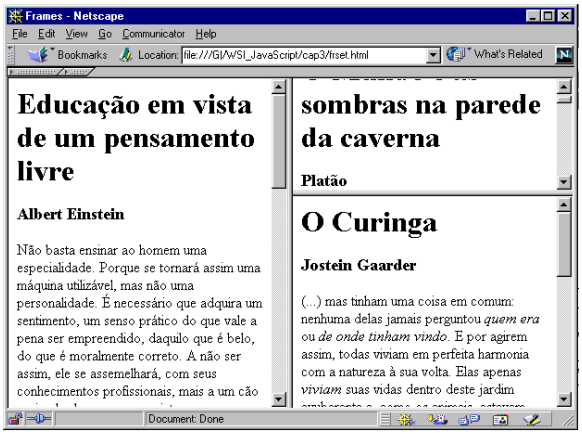
a janela foi dividida em quatro linhas, tendo a primeira e quarta 100 pixels cada de altura, a segunda 200 pixels e a terceira, o espaço restante. Combinando os dois argumentos obtém-se uma janela com 12 subjanelas.

Um bloco `<FRAMESET>...</FRAMESET>` só pode conter dois tipos de elementos:

- descritores `<FRAME>`, que definem a página HTML que ocupará uma janela. A página HTML poderá ser uma página de informação comum ou outra página de frames que dividirá a subjanela novamente em linhas e/ou colunas.
- sub-blocos `<FRAMESET> ... </FRAMESET>` que dividirão outra vez a subjanela (em linhas e/ou colunas) e poderão conter descritores `<FRAME>` e novos sub-blocos `<FRAMESET>`.

O número de sub-blocos para cada `<FRAMESET>` dependerá do número de linhas (ou colunas) definidas. Para dividir uma janela em linhas e colunas de forma irregular (com células que atravessam mais de uma linha ou coluna), pode-se proceder de duas formas:

- usar um *único* `<FRAMESET>`, contendo elementos `<FRAME>` que referem-se a páginas de frames (páginas que também definem um `<FRAMESET>`), ou
- usar vários `<FRAMESET>` em cascata na mesma página.



---

após o `<FRAMESET>` será ignorado por browsers que suportam frames, mas será lido por browsers antigos que não os suportam.

Usaremos as duas formas para montar a janela ao lado. Na primeira versão, utilizaremos *dois* arquivos de frames: `frset1.html` dividirá a janela principal em duas colunas, e `frset2.html` dividirá a segunda coluna em duas linhas. Na segunda versão, precisaremos de apenas *um* arquivo de frames (`frset.html`). As duas versões utilizarão três arquivos de informação: `um.html`, `dois.html` e `tres.html`. Visualmente, o resultado final é o mesmo, mas as duas formas podem ser manipuladas de forma diferente.

Na primeira versão temos *dois* arquivos. Os trechos em negrito indicam as ligações entre eles. O primeiro é `frset1.html`, que referencia uma página de informação:

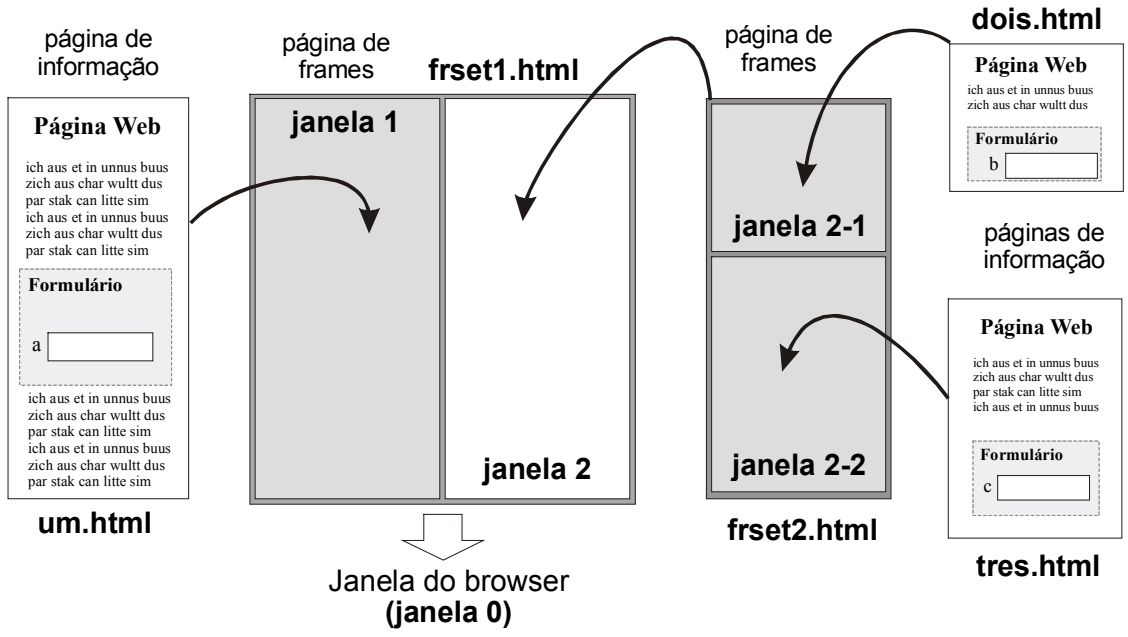
```
<html>
<head> ... </head>

<frameset cols="50%,50%">
   <frame name="janela1" src="um.html">
   <frame name="janela2" src="frset2.html">
</frameset>
</html>
```

e chama `frset2.html`, com mais duas páginas de informação, listado abaixo:

```
<html>
<head> ... </head>
<frameset rows="35%,65%">
   <frame name="janela2_1" src="dois.html">
   <frame name="janela2_2" src="tres.html">
</frameset>
</html>
```

A figura abaixo mostra a organização das páginas de informação e das páginas de frames na janela do browser.



Observe que há *três níveis* de páginas. No nível mais alto está a página `frset1.html`, que ocupa toda a janela do browser. No segundo nível estão os arquivos `um.html` e `frset2.html`. E no terceiro nível, encontramos os arquivos `dois.html` e `tres.html`, que estão dentro de `frset2.html`.

Na segunda versão, temos apenas *um* arquivo de frames contendo referências para os três arquivos de informação. Em negrito está mostrado o segundo *frameset* (observe que ele substitui o elemento `<FRAME>` que chamava `frset2.html` no último exemplo.)

```
<html>
<head> ... </head>
<frameset cols="50%,50%">
   <frame name="janela1" src="um.html">
   <frameset rows="35%,65%">
      <frame name="janela2_1" src="dois.html">
      <frame name="janela2_2" src="tres.html">
   </frameset>
</frameset>
</html>
```

Esta segunda versão possui apenas dois níveis. No primeiro, a página de frames `frset.html`, no segundo, as páginas de informação. A aparência final é a mesma, nas duas versões, mas na `primeira` versão há uma janela a mais (`janela2`) que pode ser manipulada via JavaScript ou HTML.

3.3.2. Controle dos alvos dos vínculos

O comportamento usual dos browsers é o de abrir uma nova página sempre dentro da janela atual. É possível alterar esse comportamento através do atributo TARGET em vínculos de hipertexto <A HREF> e elementos <BASE>.

Se a `janela2` for utilizada como alvo de um link HTML:

```
<a href="pagina.html" TARGET="janela2"> link </A>
```

os frames `janela2_1` e `janela2_2`, que estão em um nível abaixo de `janela2` deixarão de existir e `pagina.html` ocupará toda a segunda coluna da janela do browser. Isto não poderá ser feito na segunda versão, pois ela só possui dois níveis.

Se o link `estiver` dentro da página `dois.html` ou `tres.html`, a sintaxe abaixo, usando o nome especial `_parent` causará um resultado equivalente:

```
<a href="pagina.html" TARGET="_parent"> link </A>
```

3.4. Multimídia

3.4.1. Imagens mapeadas

Para fazer com que uma imagem sirva como origem de um vínculo de hipertexto basta colocá-la dentro de um bloco <A HREF>. Mas se o que se deseja é que partes diferentes da imagem apontem para lugares diferentes, é preciso usar imagens mapeadas ou *imagemaps*. Imagens mapeadas são muito usadas em barras de navegação, mapas, diagramas, etc. Sempre que for necessário selecionar uma área irregular de uma imagem e fazer com que ela esteja relacionada a um destino (página, âncora local, etc.) pode-se usá-la.

Há basicamente dois tipos de imagens mapeadas: as que funcionam no servidor (no seu provedor) e as que funcionam do lado do cliente (no seu computador). O primeiro tipo necessita de um programa CGI e um arquivo especial que contém as coordenadas das áreas ativas armazenados no servidor. É uma solução obsoleta e mais complicada. O segundo realiza tudo sem acessar a rede e portanto é mais rápido e mais eficiente. Todos os browsers mais recentes (desde as versões 2.0) suportam imagens mapeadas do lado do cliente.

Para criar uma imagem mapeada é necessário ter uma imagem e calcular as coordenadas de suas áreas "ativas". A maior parte dos editores HTML possuem ferramentas para gerar coordenadas, mas se a sua não tiver, você pode sempre obter ferramentas gratuitas na rede como o *MapEdit para Windows* (http://www.boutell.com/mapedit/) ou o *MapMaker para Mac* (http://www.kickinit.com/mapmaker/).

Imagens mapeadas são imagens comuns. A diferença é que têm um atributo adicional informando o nome do bloco que define o mapa que será utilizado. O atributo é USEMAP:

```
<BODY>
<IMG SRC="banner.gif" USEMAP="#isoptera" ALT="Menu de Opções">
</BODY>
```

O atributo USEMAP acima possui um valor #isoptera que corresponde ao nome de um mapeamento. Incluímos uma estrutura de mapeamento na página através do bloco <MAP>. Todo o bloco <MAP> é gerado pelo programa que calcula as coordenadas. Para a imagem abaixo (banner.gif) o mapa, sem as coordenadas, é o seguinte:

```
<MAP NAME="isoptera">
 <AREA SHAPE="rect" COORDS="..." HREF="core.html" target=_top>
 <AREA SHAPE="rect" COORDS="..." HREF="livraria.html">
 <AREA SHAPE="rect" COORDS="..." HREF="cdshop.html">
 <AREA SHAPE="rect" COORDS="..." HREF="eventos.html">
 <AREA SHAPE="rect" COORDS="..." HREF="pulgas.html">
 <AREA SHAPE="rect" COORDS="..." HREF="sair.html" target=_top>
</MAP>
```

Para selecionar as áreas ativas da figura, você precisará descrevê-las, informando a *forma* (`POLY`, `RECTANGLE` ou `CIRCLE`) e suas coordenadas (lista de números separados por vírgulas) além da URL que devem seguir (atributo `HREF`). Esses atributos do descritor `<AREA>` definem a área utilizada.

Os polígonos (`SHAPE="poly"`) são definidos em termos das coordenadas de seus vértices. Cada coordenada (par de dois números) corresponde a um vértice (x, y) do mesmo. A origem (0,0) corresponde ao canto superior esquerdo da imagem `banner.gif`. Círculos (`SHAPE="circle"`) devem incluir três números separados por vírgulas como coordenadas. Os dois primeiros são a coordenada do centro do círculo e o último seu raio. Retângulos (`SHAPE="rectangle"`) são definidos em termos de quatro números: as coordenadas no canto superior esquerdo e do canto inferior direito.

O bloco `<MAP>` pode estar em qualquer lugar da página pois é identificado pelo nome e só se torna ativo depois que a página inteira for carregada.

3.4.2. Áudio, vídeo e plug-ins

Componentes Flash e Shockwave, imagens e vídeos podem ser incluídos em uma página usando o elemento <EMBED> ou o elemento <OBJECT>. Embora o último seja recomendado pela especificação do HTML 4.0, o suporte a <EMBED> é mais antigo e funciona tanto nos browsers Internet Explorer como Netscape.

Os atributos de <EMBED> variam conforme o componente instalado. Na maior parte dos casos, o componente ocupa uma parte da página como uma imagem, e pode ser alinhado e redimensionado usando os mesmos atributos disponíveis em <IMG>. A sintaxe mínima de <EMBED> é:

```
<EMBED SRC="URL do componente" HEIGHT=100 WIDTH=150>
</EMBED>
```

Audio

Há vários formatos de áudio suportados na Web. Os que possuem maior suporte são os formatos `.au` e `.mid` suportados pelo componente *LiveAudio* disponível no browser Netscape ou Internet Explorer. Mais recentemente tem crescido o suporte ao formato `.mp3`. O *LiveAudio* exibe (opcionalmente) um controle de tela que permite ao usuário manipular o som.

O elemento `<EMBED>` usado para incluir áudio possui vários atributos adicionais. A sintaxe típica é:

```
<EMBED SRC="musica.mid" CONTROLS=console HEIGHT=60
       WIDTH=150 AUTOSTART=false></EMBED>
```

O atributo `CONTROLS` pode receber os valores `console`, `smallconsole`, `playbutton`, `pausebutton`, `stopbutton` ou `volumelever` e alterar a aparência do *display*. O atributo `LOOP` pode ter os valores `true`, `false` ou um número indicando o número de vezes que o som deve tocar. `AUTOSTART` pode ser `true` ou `false` indicando se o som deve começar imediatamente ou esperar pela decisão do usuário. `VOLUME` varia de 0 a 100 e estabelece o volume do som.

No Internet Explorer (e apenas no Internet Explorer) som de fundo pode ser incluído usando o elemento <BGSOUND>:

```
<BGSOUND SRC="musica.mid" LOOP=3>
```

Vídeo

Assim como os recursos de áudio, vários formatos de vídeo como `.mpg`, `.mov`, `.avi`, `.rm`, etc. podem ser incluídos em páginas HTML usando o elemento `<EMBED>`. A sintaxe típica é:

```
<EMBED SRC="filme.mov" CONTROLS=console HEIGHT=260
       WIDTH=450 AUTOPLAY=false></EMBED>
```

Cada *plug-in* tem seus próprios atributos que podem ser acrescentados ao descritor `<EMBED>`. O formato `.mov` (*QuickTime*) suporta vários outros atributos, entre eles `CONTROLLER=true` ou `false`, para ligar ou desligar o controle do vídeo; `LOOP = true`, `false` ou `palindrome` para fazer o vídeo executar continuamente e de trás para frente.

No *Internet Explorer* (e somente no *Internet Explorer*) é possível incluir vídeos usando o atributo `DYNSRC` de `<IMG>`:

```
<IMG DYNSRC="filme.mov">
```

3.4.3. Flash

Flash é um formato de multimídia desenvolvido pela Macromedia que oferece a possibilidade de criar animações, interatividade e integração com áudio e vídeo. Os arquivos são muito menores que os utilizados nas tecnologias padrão da Web pois o formato das imagens e animações do Flash se baseia em vetores (vector) e não em bitmaps. Imagens vetoriais descrevem os desenhos, fontes e comportamentos em termos de equações, que ocupam bem menos espaço de armazenamento que as animações baseadas em bitmaps, que consistem da repetição de arquivos que contém o mapa completo de cores e pixels de cada tela.

Há diversas vantagens no formato oferecido pela tecnologia Flash. Os arquivos são menores, são escaláveis sem perda de qualidade, suportam som integrado, podem ser desenvolvidos sem a utilização de programação, contam com amplo suporte e ainda podem ser controlados através de parâmetros passados em HTML ou JavaScript.

As desvantagens se baseiam principalmente em se tratar de um formato proprietário. É preciso adquirir um software para produzir arquivos Flash. Para exibi-lo, é preciso fazer o download de um plug-in. Não há suporte em Linux ou Unix e nem para browsers não-gráficos, que perdem totalmente o conteúdo da página.

Para criar componentes Flash é preciso ter o pacote Macromedia Flash 4. Há uma versão de demonstração que vale por 30 dias no site http://www.flash.com/. O objetivo desta seção é apenas mostrar como se pode usar um componente Flash genérico que já foi criado previamente.

Configuração do servidor

Pode ser necessário configurar o servidor Web para que ele saiba servir arquivos Flash. Informe-se no seu provedor sobre a configuração. Se você for o webmaster responsável por um servidor, você deverá acrescentar a seguinte relação tipo MIME/extensão de arquivo para que o servidor reconheça o formato:

- *Tipo/subtipo:* application/x-shockwave-flash
- *Extensão*: .swf

Para acrescentar Flash em uma página HTML, use o elemento `<EMBED>`. A sintaxe mínima é:

```
<EMBED SRC="filme.swf" HEIGHT=260 WIDTH=450>
</EMBED>
```

3.5. Applets Java

3.5.1. O que são applets Java?

A Web é popular porque suas informações, na forma de texto e imagens, são visíveis independente da plataforma (tipo de máquina e sistema operacional) do usuário. Nem sempre é possível ver determinado vídeo dependente de plug-in, ou um certo site interativo dependente de Flash, pois às vezes o programa necessário à execução da aplicação depende de plataforma e pode ser até que não exista para a plataforma onde roda seu browser.

Applets são pequenas aplicações *independentes de plataforma* (por isso populares na Web), geralmente escritas em uma linguagem de programação chamada Java, que têm sua execução iniciada pelo browser. Diferentemente do que ocorre com as linguagens embutidas em blocos `<SCRIPT>` (JavaScript), o *código* Java escrito pelo programador da aplicação não é *interpretado* nem visualizado pelo browser, já que foi antes *compilado* para uma linguagem de máquina. Browsers que suportam Java possuem uma *plataforma virtual*, a "Java Virtual Machine", que simula uma máquina Java em sistemas operacionais diferentes como Windows, Macintosh, Unix, etc. Só a máquina virtual Java é capaz de interpretar a linguagem de máquina Java. Através dela, browsers são capazes de rodar programas sofisticados que podem desde oferecer uma interface de formulários mais segura interativa, até mostrar áudio, vídeo, 3D e sistemas de comunicação visual interativa em qualquer plataforma.

3.5.2. Quando usar applets

Applets podem ser usados para desenvolver aplicações que seriam impossíveis em HTML com ou sem linguagens embutidas (como JavaScript) e fugir das limitações do HTML, do protocolo HTTP e do próprio browser. Com um applet, é possível *estender* um browser fazendo-o suportar, por exemplo, novos protocolos de comunicação e segurança, novos formatos de mídia, etc. O preço dessa liberdade é sua fraca integração com o HTML da página. Aplicações Web baseadas em Java pouco ou nada aproveitam do HTML da página a não ser um espaço gráfico para sua exibição. O browser simplesmente reserva um espaço para rodar o programa, e o programa assume aquele espaço e faz o que bem quiser (limitado apenas por restrições de segurança). É possível aumentar essa integração com linguagens embutidas no HTML como JavaScript.

Applets são aplicações gráficas e precisam de uma página HTML para poderem executar. São exibidos na página de forma semelhante a imagens: carregam um arquivo externo, ocupam um espaço com determinada altura e largura, e podem ter seu alinhamento em relação ao texto definido pelos mesmos atributos presentes em `<IMG>`. A sintaxe do elemento HTML `<APPLET>` está mostrada abaixo. Tudo o que não estiver em negrito é opcional:

```
<APPLET
   CODE="nome_do_programa.class"
   HEIGHT="altura em pixels"
   WIDTH="largura em pixels"
   NAME="nome_do_objeto"
   CODEBASE="diretório base do arquivo de classe Java"
   ARCHIVE="arquivo .jar contendo o programa"
   ALT="texto descritivo"
   HSPACE="margens externas laterais em pixels"
   VSPACE="margens externas verticais em pixels"
   ALIGN="left" ou "right" ou "top" ou "middle" ou "bottom" ou
         "texttop" ou "absmiddle" ou "absbottom" ou "baseline"
   MAYSCRIPT>
      <PARAM NAME="nome" VALUE="valor">
      <PARAM NAME="nome" VALUE="valor">
         ...
      <PARAM NAME="nome" VALUE="valor">
</APPLET>
```

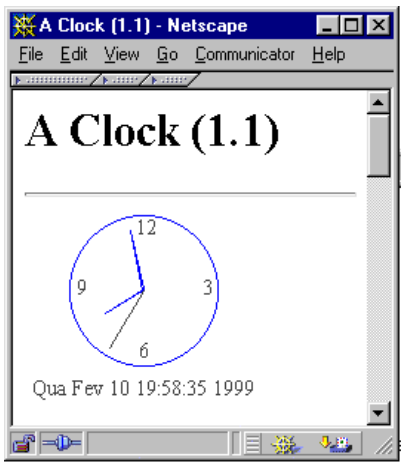
Diferentemente de <IMG>, o elemento <APPLET> é um bloco e possui um descritor de fechamento </APPLET>. Entre <APPLET> e </APPLET> pode haver nenhum ou vários elementos <PARAM>, que contém parâmetros necessários ou não (depende do applet) para o funcionamento da aplicação. Cada elemento <PARAM> contém um par de atributos obrigatórios. O valor do atributo NAME é definido pelo programador do applet. Através dele, o programa pode recuperar um valor que o autor da página (que não precisa saber Java) definiu no atributo VALUE.

Os atributos CODEBASE e ARCHIVE são usados opcionalmente para localizar applets em outros servidores e importar outros módulos do programa, respectivamente. O atributo MAYSCRIPT é necessário se o applet pretende ter acesso ao código JavaScript da página. Sem este atributo, qualquer tentativa do applet de acessar variáveis ou executar funções ou métodos JavaScript causará em uma exceção de segurança no applet.

No HTML 4.0, applets também podem ser incluídos em uma página usando o bloco genérico <OBJECT>, que também serve para importar imagens, *plug-ins* e construir *frames* internos.

Existem muitos applets úteis disponíveis gratuitamente na Web que podem ser usados por autores de páginas Web e programadores JavaScript sem que precisem saber Java. Os mais populares implementam *banners* para rolagem de texto, ícones inteligentes, gráficos, planilhas de dados e interfaces para bancos de dados. A maioria são configuráveis através de parâmetros que o autor da página define em elementos <PARAM>. No apêndice A há uma lista de sites onde se pode encontrar tais applets. Exemplos são o *Gamelan* (http://www.gamelan.com) e o site da Sun sobre Java (http://java.sun.com). A seção a seguir mostrará como incluir um applet obtido em um desses sites.

3.5.3. Como incluir um applet em uma página



O exemplo a seguir mostra como incluir o applet `Clock2` em uma página Web. Este applet é distribuído pela *Sun* gratuitamente em `http://java.sun.com` e juntamente com o ambiente de desenvolvimento Java. O applet pode ser incluído na página da forma *default*, sem especificar parâmetros, ou de forma personalizada, definindo parâmetros que alteram a cor de fundo, dos ponteiros e do mostrador.

Como é que o autor de uma página HTML saberá quais parâmetros usar e configurar? Vendo-o funcionar! O applet é distribuído com uma página HTML que mostra como usá-lo. Ele deve ser incluído na página HTML usando o nome do arquivo executável java, que neste caso é `Clock2.class` e deve ocupar uma área de no mínimo 170x150 pixels (tudo isto está na página):

```
<applet code="Clock2.class" height=150 width=170></applet>
```

Com o código acima, o relógio aparece na página como mostrado na figura, com ponteiros azuis, visor com letras pretas e fundo branco. O autor do applet permite, porém, que o autor da página altere esses parâmetros através de descritores `<PARAM>`. Os três parâmetros modificáveis são:

- `bgcolor` – cor de fundo (branco é *default*)
- `fgcolor1` – cor dos ponteiros e dial (azul é *default*)
- `fgcolor2` – cor dos números e ponteiro de seguntos (preto é *default*)

Todos os parâmetros devem receber como valor um número hexadecimal representando uma cor no formato RGB (mesmo formato usado em HTML): `ff0000` – vermelho, `ffffff` – branco, `0000ff` – azul, etc.

Portanto, para incluir o relógio acima em uma página com um fundo cinza claro, ponteiros marrons e letras douradas, o código seria:

```
<applet code="Clock2.class" width=170 height=150>
      <param name=bgcolor value="dddddd">
      <param name=fgcolor1 value="800000">
      <param name=fgcolor2 value="808000">
</applet>
```

Applets remotos

Caso o applet esteja em um diretório diferente daquele onde está a página, será necessário usar o atributo `CODEBASE`, para informar a URL base. Por exemplo, se o arquivo `.class` que usamos acima estiver em `http://www.abc.com/clocks/`, precisamos usar:

```
<applet codebase="http://www.abc.com/clocks/" code="Clock2.class" ... >
   ...
</applet>
```

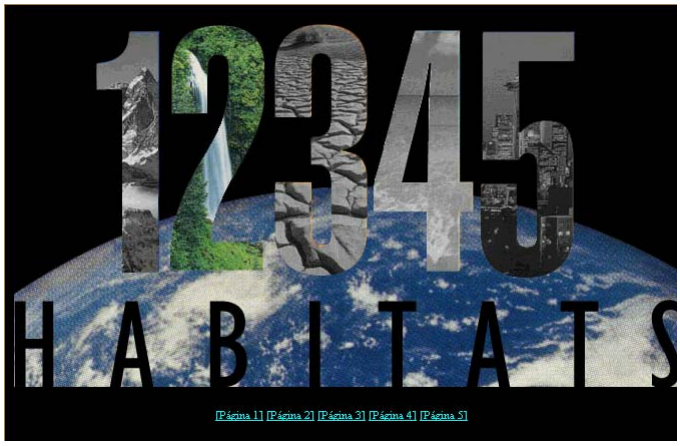
3.5.4. Conclusão

Applets oferecem um meio para escapar das limitações do browser e do HTML sem limitar o público-alvo da página de acordo com a plataforma de computação que possuem. É preciso, porém, lembrar que podem ocorrer problemas. Como applets são programas, podem conter ou provocar erros. Geralmente esses erros não são graves a ponto de comprometer a segurança do usuário, pois os browsers que suportam applets possuem um rigoroso mecanismo que restringe o que applets podem fazer. Eles não podem, por exemplo, escrever no seu disco ou imprimir. Mas os erros podem levar o applet a não funcionar, ou pior, a consumir os recursos de memória do browser e do sistema, forçando o usuário a abandonar sua execução (e possivelmente a do browser). Applets escritos para as versões Java 2 podem não funcionar até em browsers recentes, como o *Internet Explorer 4*. Embora funcionem em plataformas diferentes, applets podem rodar até 30 vezes mais rápido em certas plataformas. Isto depende da qualidade da máquina virtual suportada pelo browser.

Levando em conta todas essas questões, você deve usar applets quando for necessário incluir recursos que o browser e o HTML sozinhos não seriam capazes de oferecer. É preciso, porém, levar em conta o risco de inacessibilidade caso o applet não funcione por um motivo ou outro. Se possível, deve haver uma forma alternativa de obter a informação que ele guarda ou a funcionalidade que ele implementa. Se o seu público-alvo permitir, você talvez decida que é melhor usar outra tecnologia mais simples, visível em menos plataformas, mas que atenda seus objetivos.

3.6. Exercícios

1. Monte o quebra cabeças abaixo com as imagens `im-01.jpg` a `im-05.jpg` do `kit1` (disponíveis no site do curso), juntamente com as imagens, `esquerda.jpg`, `direita.jpg` e `baixo.jpg` para obter o desenho abaixo usando tabelas HTML.



2. Organize o texto (do `kit2`) da forma mostrada nas figuras abaixo