Unidade 5: Posicionamento com CSS

E outras Técnicas Avançadas

Prof. Daniel Caetano

Objetivo: Apresentar recursos adicionais do CSS e como usá-lo para a construção de layout de página.

Bibliografia: W3, 2009; CASCADE, 2006; RAMALHO, 1999; NIELSEN, 2000.

INTRODUÇÃO

Conceitos Chave:
- Estilos "Avançados"
  * Transparência, links visitados etc...
- CSS => não serve apenas para mudar estilos!
  * Posicionamento de Elementos!
  * Mudar o alinhamento... só?
- Auxílio do XHTML
  * É preciso indicar o "início" e o "fim" de cada elemento.
  * É preciso ordenar estes elementos
    + Tags de Divisão de documento

Anteriormente, foi apresentado como usar as CSSs para alterar a aparência de nosso documento. Muitas propriedades do documento podem ser alteradas, como cor de fundo, cor de texto, dentre outras.

Entretanto, o mecanismo visto para modificação de propriedades é um tanto quanto limitado com relação ao posicionamento dos diversos elementos de uma página: podemos mudar alinhamentos e, talvez, a ordem de alguns elementos. Entretanto, se quisermos posicionar o conteúdo de alguma forma menos tradicional, não será possível apenas com o que já vimos até agora.

O objetivo desta aula é, então, apresentar mais alguns recursos do CSS e como definir elementos que permitam posicionar diferentes "blocos" de uma página Web no lugar desejado. Como será visto, tudo isso será feito primordialmente no CSS, com poucas modificações no documento HTML.

1. PSEUDO-CLASSES E PSEUDO-ELEMENTOS

Conceitos Chave:
- Definindo estilos específicos de ações
  * tag:estado { ... }
- Exemplos
  a:hover
  a:visited
  p:first-child

Alguns elementos do HTML, com a tag <A>...</A>, que define um link, possuem diversos "estados": um link pode ser um link não visitado, um link já visitado, um link com o ponteiro do mouse sobre ele, e assim por diante. Assim, para definir características específicar em cada um destes estados, é necessário acrescentar uma informação no nome da tag indicada no CSS.

```
elemento:estado {
        propriedade: valor;
        }
```

Estes "estados" são chamados de "pseudo-classes" e as mais comuns são:

:active	Especifica estilo para um elemento ativo (ex.: a:active)
:focus	Especifica estilo para elemento em foco (ex.: input:focus)
:hover	Especifica estilo para elemento com mouse sobre ele (ex.: a:hover)
:link	Especifica estilo para link não visitado (a:link)
:visited	Especifica estilo para link já visitado (a:visited)
:first-child	Especifica estilos diferentes internos a uma primeira ocorrência de um elemento em uma região da página.

Por exemplo: para fazer com que um link não visitado seja verde e sublinhado, mas se torne vermelho e sublinhado quando o mouse passar por cima, usa-se o seguinte CSS:

```
a:link {
        color: green;
        text-decoration: none;
        }

a:hover {
        color: red;
        text-decoration: underline;
        }
```

1.1. Pseudo-Elementos

Os pseudo-elementos são muito similares às pseudo-classes, mas definem o comportamento de elementos que não estão claramente definidos no HTML. Os pseudo-elementos mais comuns são:

:first-letter	Especifica estilo para a primeira letra de uma tag (ex.: p:first-letter)
:first-line	Especifica estilo para a primeira linha de uma tag (ex.: p:first-line)
:before	Insere algum conteúdo (áudio? vídeo?) antes de um elemento (url:)
:after	Insere algum conteúdo (áudio? vídeo?) depois de um elemento (url:)

2. TRANSPARÊNCIA

Conceitos Chave:
- Transparência/Opacidade de imagens
  * FireFox, IE 9 em diante - CSS3 (0.0 <= x <= 1.0)
    opacity: x
  * IE 5 a 7 (0 <= x <= 100)
    filter:alpha(opacity=x)
  * IE 8 (0 <= x <= 100)
    -ms-filter:"progid:DXImageTransform.Microsoft.Alpha(Opacity=X)
  * IE8 deve vir antes do IE 5-7

Algumas propriedades ainda não padronizadas pelo CSS2 já estão disponíveis nos navegadores. Estas propriedades estão padronizadas no CSS3, mas como o CSS3 não é, ainda, exatamente oficial, cada navegador implementa estas propriedades de um jeito, obrigando o programador a especificar o mesmo efeito em mais de um formato dentro do arquivo CSS.

Uma destas propriedades que é muito importante é a transparência, sendo este um dos mais desejados pelos web masters. Ele é conseguido através dos seguintes atributos:

Propriedade:	**Onde e Como Funciona:**
opacity: *x*	Firefox e IE9+
-ms-filter:"progid:DXImageTransform.Microsoft.Alpha(Opacity=x)	IE8, x de 0 a 100
filter:alpha(opacity= *x*)	IE5 a 7, x de 0 a 100

O padrão CSS3 é o mesmo adotado pelo FireFox.

Considere a página a seguir, que contém um pano de fundo, um texto em um H1 e um texto dentro de um parágrafo. Observe como o texto tem baixa legibilidade quando se encontra sobre a figura do planeta Terra, devido ao baixo contraste entre as cores da imagem e a cor usada no texto.

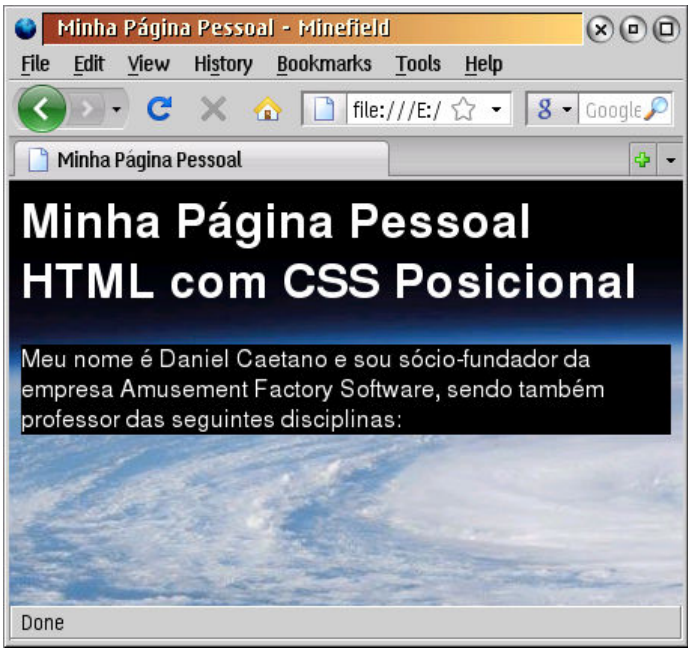
exemplo.html

```
<!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" xml:lang="pt-BR">
<head>
      <meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1" />
      <link href="exemplo.css" rel="stylesheet" type="text/css" />
      <title>Minha página pessoal.</title>
</head>
<body>
      <h1>Minha P&aacute;gina Pessoal HTML com CSS Posicional</h1>
      <p>     Meu nome &eacute; Daniel Caetano e sou s&oacute;cio-fundador da empresa
              Amusement Factory Software, sendo tamb&eacute;m professor das seguintes
              disciplinas:
      </p>
</body>
</html>
```

exemplo.css

```
body {
  background-color: rgb(0,0,0);
  background-image: url("earth2.jpg");
  color: rgb(255,255,255);
}

p {
  background-color: rgb(0,0,0);
}
```



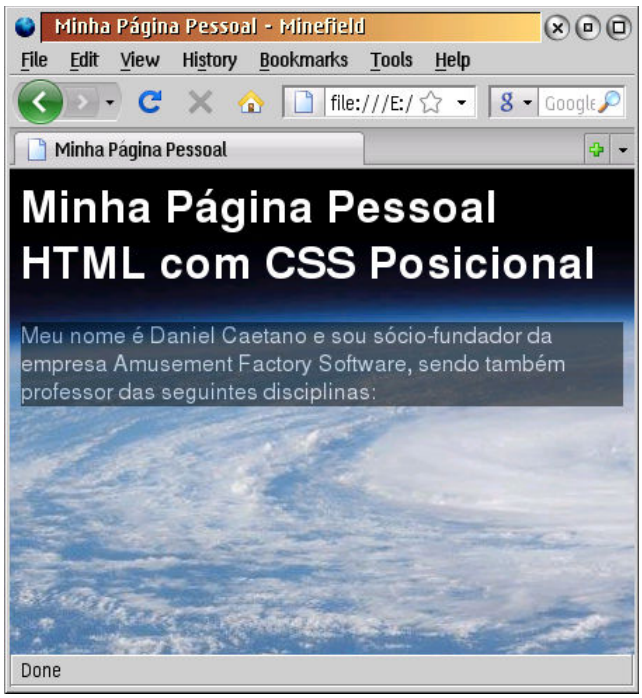
É possível que o WebMaster não queira um fundo "bloco preto" para seu parágrafo, e considere interessante poder defini-lo com um bloco preto com 50% de transparência, permitindo que o fundo seja visível através do bloco. Para isso, basta a seguinte modificação no arquivo .CSS:

exemplo.css

```
body {
  background-color: rgb(0,0,0);
  background-image: url("earth2.jpg");
  color: rgb(255,255,255);
}

p {
  background-color: rgb(0,0,0);
  opacity: 0.5;                                                    // Padrão
  -ms-filter:"progid:DXImageTransform.Microsoft.Alpha(Opacity=50)  // IE 8
  filter:alpha(opacity=50);                                        // IE5 a 7
}
```

O resultado é apresentado abaixo.



Caso não se preocupe com navegadores antigos e fora do padrão, o CSS fica:

exemplo.css

```
body {
  background-color: rgb(0,0,0);
  background-image: url("earth2.jpg");
  color: rgb(255,255,255);
}

p {
  background-color: rgb(0,0,0);
  opacity: 0.5;                                                    // Padrão
}
```

Observe, porém, que em todos os casos a transparência está sendo "herdada" pelo texto, o que a torna quase inútil... mas ela é a única que se aplica tanto a imagens quanto a cor de fundo.

Quando quisermos deixar apenas a cor de fundo de um elemento transparente, podemos definir essa cor com o elemento **rgba(vermelho, verde, azul, transparência)**. Exemplo:

exemplo.css

```
body {
   background-color: rgb(0,0,0);
   background-image: url("earth2.jpg");
   color: rgb(255,255,255);
}

p {
   background-color: rgba(0,0,0,0.5);
}
```

O resultado será esse:

3. DIVISÃO DE DOCUMENTO

Conceitos Chave:
- DIV => <DIV ID="*nome*">...</DIV>
    * Regiões a Reposicionar x Redefinir Estilos
- Exemplo

Até este momento, trabalhamos com corpo do documento HTML como se fosse um elemento único, isto é, não houve uma identificação clara do que seria "menu", "cabeçalho do conteúdo", "área de notícias", "área de conteúdo"...

Ora, se queremos posicionar cada uma destas partes de maneira independente, isto é, queremos indicar exatamente onde cada uma delas deve estar, então precisamos dividir o documento HTML em todas estas partes; entretanto, para evitar confusão e aumentar ainda mais o número de arquivos, não faremos uma divisão física. Como fazer, então?

A idéia é fazer uma divisão lógica do documento, ou seja: mantemos um único arquivo, mas indicaremos com **tags** o que cada trecho representa. Deixaremos de pensar no corpo da página como um documento único, e passaremos apensar nele como um conjunto de blocos, como se definíssemos várias páginas distintas dentro de um mesmo documento.

Pensando assim, como temos várias páginas dentro do mesmo arquivo, passa a ser natural que possamos dar características próprias a cada um dos elementos, não só de posição, mas também de cores, fontes etc. Para tanto, a única modificação que será feita no HTML é justamente a adição de algumas tags que indicarão o que cada trecho da página representa. Por exemplo, na página 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" xml:lang="pt-BR">
<head><meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1" />
	<title>Título da Página</title>
</head><body>
	<h1>T&iacute;tulo da P&aacute;gina</h1>
	<hr />
	<h3>Menu</h3>
	<p>	<li><a href="...">Item 1</a></li>
		<li><a href="...">Item 2</a></li>
		<li><a href="...">Item 3</a></li>
	</p>
	<h2>Conteúdo</h2>
	<p>Esta é uma página pessoal!</p>
</body><html>
```

Claramente existe uma seção de "cabeçalho do conteúdo", que é composto pelo <H1>...</H1> e o <HR>. Em seguida, temos um menu, composto pelo <H3> e o primeiro <P>...</P> e, finalmente, temos o conteúdo da página, composto pelo <H2> e o segundo <P>...</P>. Embora seja possível perceber isso analisando o documento, isso não está claramente identificado.

A identificação clara é precisa é o que será feita agora, usando a **tag** de *DIVisão de documento*, a tag **<DIV>...</DIV>**. Como a tag DIV define uma região, ela tem um início e um final. O documento marcado com as tags fica como indicado 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" xml:lang="pt-BR">
<head><meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1" />
	<title>Título da Página</title>
</head><body>
	<div>	<h1>T&iacute;tulo da P&aacute;gina</h1>
		<hr />
	</div>
	<div>	<h3>Menu</h3>
		<p>	<li><a href="...">Item 1</a></li>
			<li><a href="...">Item 2</a></li>
			<li><a href="...">Item 3</a></li>
		</p>
	</div>
	<div>	<h2>Conteúdo</h2>
		<p>Esta é uma página pessoal!</p>
	</div>
</body><html>
```

Bem, mas isso ainda não está claro o suficiente. Para que possamos dar clareza (e posteriormente modificarmos as propriedades e posição de cada região) é preciso dar um nome, uma IDentificação para cada seção. Isso pode ser feito com o parâmetro **ID="*nome_da_seção*"**. Assim, o documento anterior poderia ficar da seguinte forma:

```
<!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" xml:lang="pt-BR">
<head><meta http-equiv="Content-Type" content="text/html;charset=iso-8859-1" />
	<title>Título da Página</title>
</head><body>
	<div id="titulo">
		<h1>T&iacute;tulo da P&aacute;gina</h1>
		<hr />
	</div>
	<div id="menu">
		<h3>Menu</h3>
		<p>	<li><a href="...">Item 1</a></li>
			<li><a href="...">Item 2</a></li>
			<li><a href="...">Item 3</a></li>
		</p>
	</div>
	<div id="conteudo">
		<h2>Conteúdo</h2>
		<p>Esta é uma página pessoal!</p>
	</div>
</body><html>
```

Um comentário importante é que a tag DIV só terá efeito visual no navegador se realizarmos mudanças efetivas na mesma, por meio do arquivo CSS. Caso contrário, será como se o navegador simplesmente tivesse ignorado a tag. Na página definida na aula anterior, podemos definir várias seções. Por exemplo:

** **

Minha Página Pessoal

** **
** **

ÍNDICE

1. INFORMAÇÕES PESSOAIS
2. NOTAS DE PROGRAMAÇÃO WEB
3. CONTATO

** **
** **

1. INFORMAÇÕES PESSOAIS

AMUSEMENT FACTORY

Meu nome é Daniel Caetano e sou sócio-fundador da empresa Amusement Factory Software, sendo também professor das seguintes disciplinas:

- Programação Web
- Segurança e Auditoria de Sistemas
- Pesquisa Operacional
- Arquitetura de Computadores
- Sistemas Operacionais

** **
** **

2. NOTAS DOS ALUNOS DE PROGRAMAÁÃO WEB

Tabela 1: Notas de 2009/01

NOME DO ALUNO(1)	NOTA FINAL
Sem Dados	Sem Dados

(1) Alunos do 2o. semestre de Análise e Desenvolvimento

Mais informações na PÁGINA DA DISCIPLINA PROGRAMAÇÃO WEB.

** **
** **

3. ENTRE EM CONTATO

Qualquer dúvida pode ser enviada para:
Prof. Daniel Caetano
E-Mail: daniel@caetano.eng.br

** **

Ao carregar esta página no navegador, ela será exibida exatamente como antes. Entretanto, isso só ocorre porque não realizamos qualquer mudança no arquivo CSS. Mas o que deveríamos modificar no arquivo CSS? Resumidamente, devemos indicar **o que** estas seções devem possuir de diferente!

4. MODIFICANDO SEÇÕES COM CSS

Conceitos Chave:
- Definindo estilos de seções
  * #secao { ... }
- Exemplos
  * Titulo
  * Indice
  * Secao1, secao2, secao3

Como agora não iremos mais modificar apenas a apresentação de algumas tags do HTML e sim como serão apresentados pedaços inteiros do documento, a definição no arquivo CSS é um pouco diferente daquela que vimos antes.

O formato geral é:

```
#secao {
  ...
  }
```

Como uma regra geral, também aqui o CSS é muito chato com a sintaxe. Um pequeno erro de digitação, um ponto-e-vírgula faltando ou algo do gênero pode ser responsável por sua página não aparecer corretamente. Por esta razão, muita atenção ao digitar o arquivo .CSS!

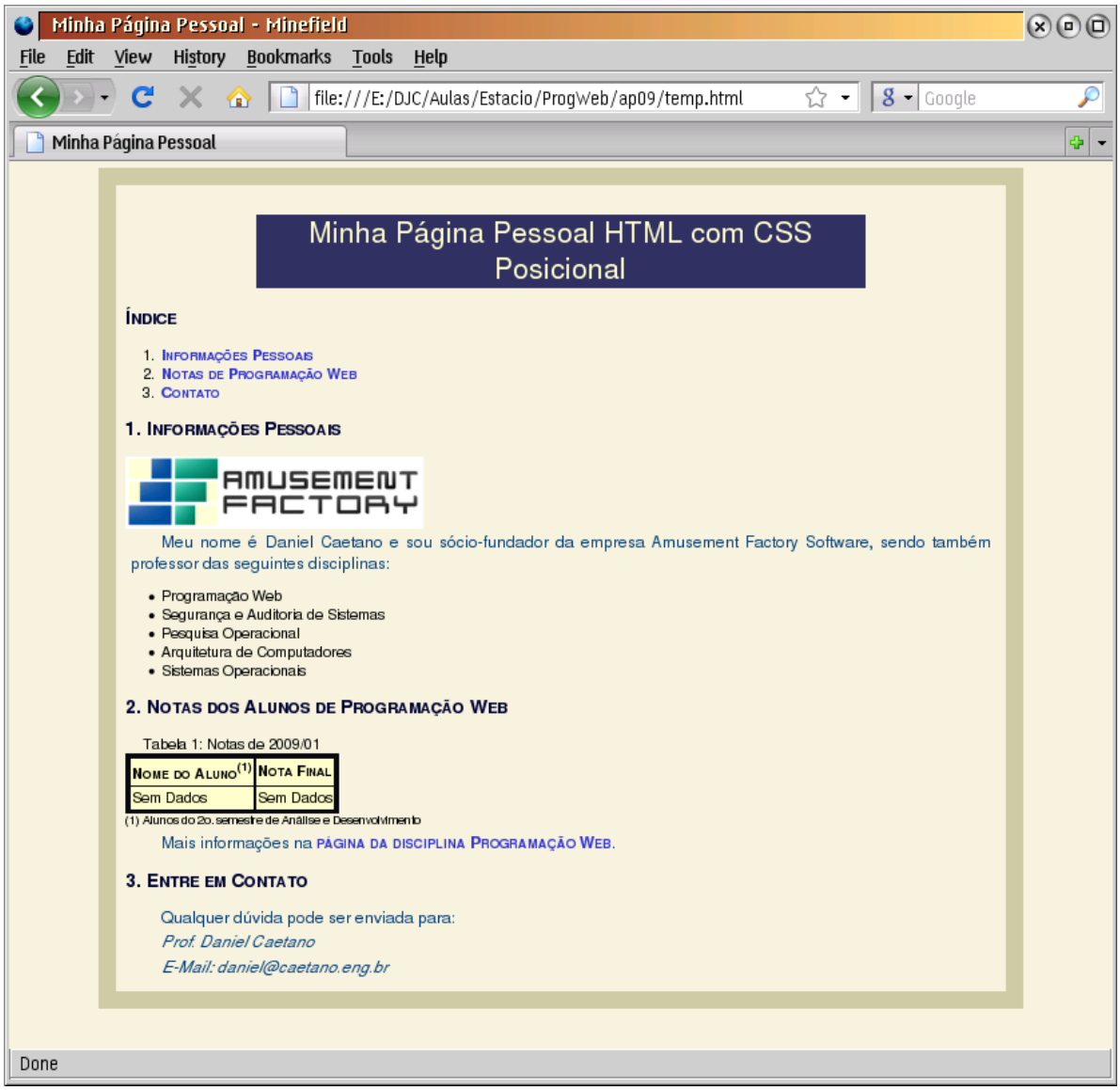
Nas próximas páginas serão apresentados alguns exemplos de modificação de propriedades e posicionamento através de CSS.

4.1. Exemplos Aplicando CSS para Posicionamento

Pegando como exemplo a página da aula anterior, para modificar a seção de título (definida como no item anterior) de forma que ela tenha um fundo azul claro e margens diferentes, devemos indicar no arquivo estilo.css como apresentado a seguir.

```
#titulo {
  background-color: rgb(50,50,100);
  margin-left: 15%;
  margin-right: 15%;
  }
```

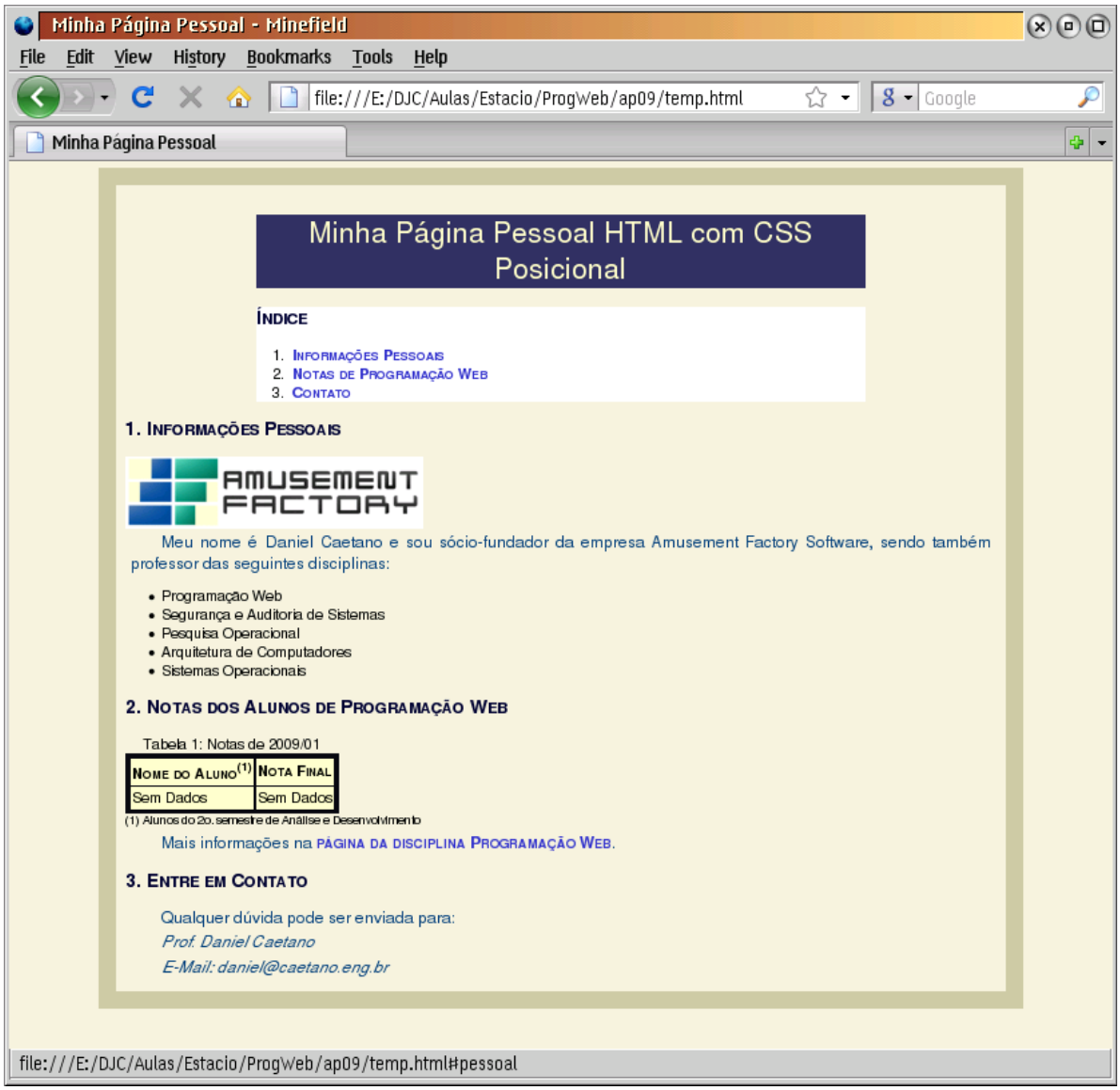
O resultado será o apresentado a seguir.



Um outro exemplo: para fazer com que o índice apareça como um retângulo de fundo branco:

```
#indice {
  margin-left: 15%;
  margin-right: 15%;
  background-color: rgb(255,255,255);
  }
```

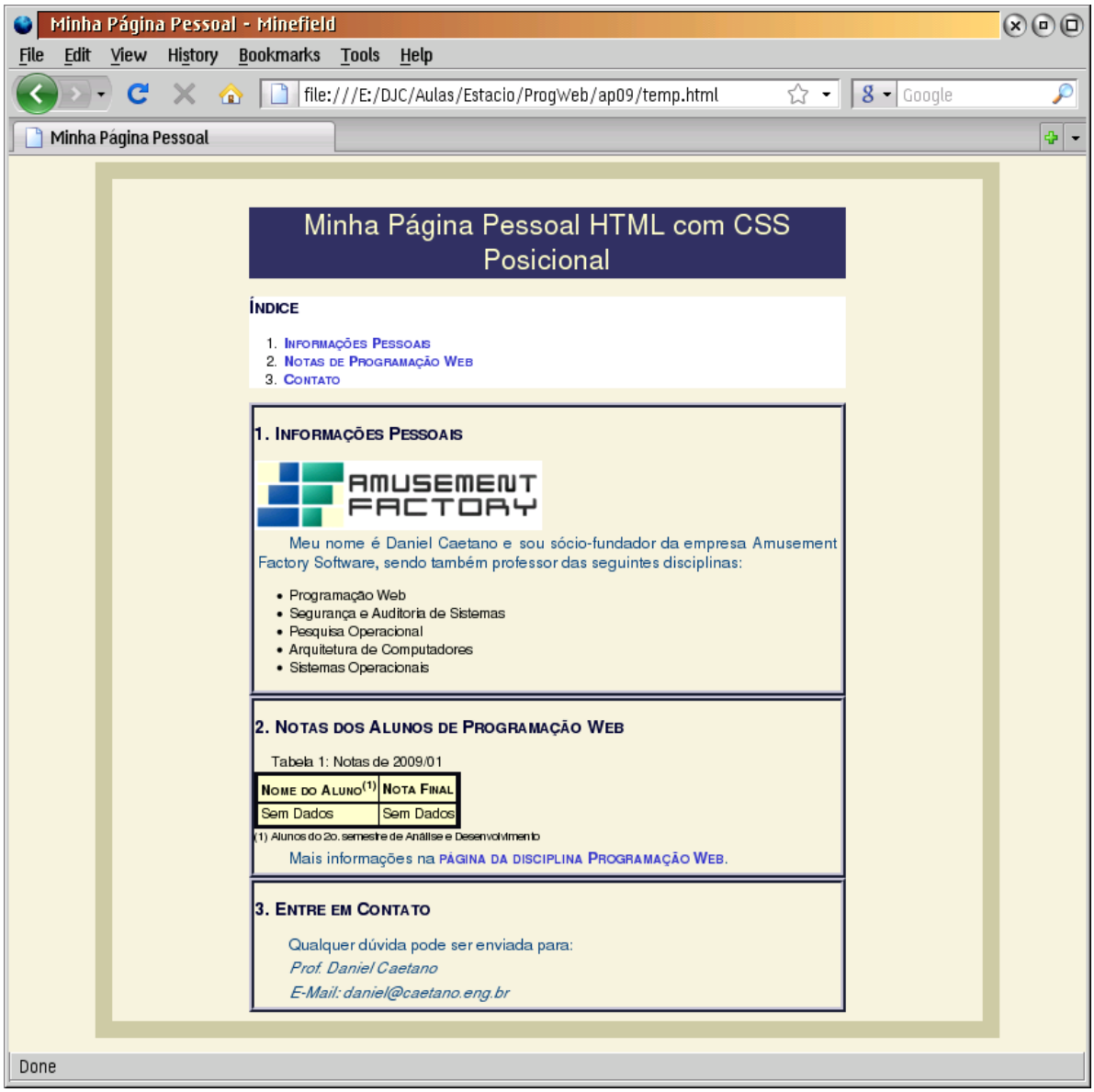
O resultado é apresentado a seguir.



Vamos movimentar fazer com que as seções 1, 2 e 3 recebam uma borda azul escura, de 6 pixels de largura, 3D arredondada:

```
#secao1, #secao2, #secao3 {
  margin-left: 15%;
  margin-right: 15%;
  border-color: rgb(50,50,100);
  border-width: 6px;
  border-style: ridge;
  }
```

O resultado é apresentado a seguir.



Como é possível observar, ficaram uns espaços "estranhos" entre a região de título, índice e as seções no FireFox. Estes espaços não aparecem no Internet Explorer (ao menos até a versão 8.0). Isso ocorre por uma pequena diferença na interpretação das margens dos títulos (H1 a H6).

Se quisermos corrigir esta diferença (e, neste caso, desejamos), basta acrescentar a propriedade "overflow: hidden" em cada uma das regiões DIV:

```
#titulo {
  overflow: hidden;
  background-color: rgb(50,50,100);
  margin-left: 15%;
  margin-right: 15%;
  }
```

```
#indice {
  overflow: hidden;
  margin-left: 15%;
  margin-right: 15%;
  background-color: rgb(255,255,255);
  }

#secao1, #secao2, #secao3 {
  overflow: hidden;
  margin-left: 15%;
  margin-right: 15%;
  border-color: rgb(50,50,100);
  border-width: 6px;
  border-style: ridge;
  }
```

Isso corrige a diferença para o FireFox, mas tem dois efeitos colaterais: se reduzirmos demais a janela, o texto ficará cortado. Da outra forma o texto não é cortado, mas sairá para fora das regiões delimitadas. Ou seja: de qualquer forma o resultado de uma janela muito pequena não será bonito. Veremos no futuro como minimizar este problema. O segundo efeito colateral é muito importante e útil, e será apresentado em breve.

Para melhorar um pouquinho o visual de nossa página, vamos acrescentar alguns espaçamentos nas seções DIV, acrescentando as linhas indicadas abaixo:

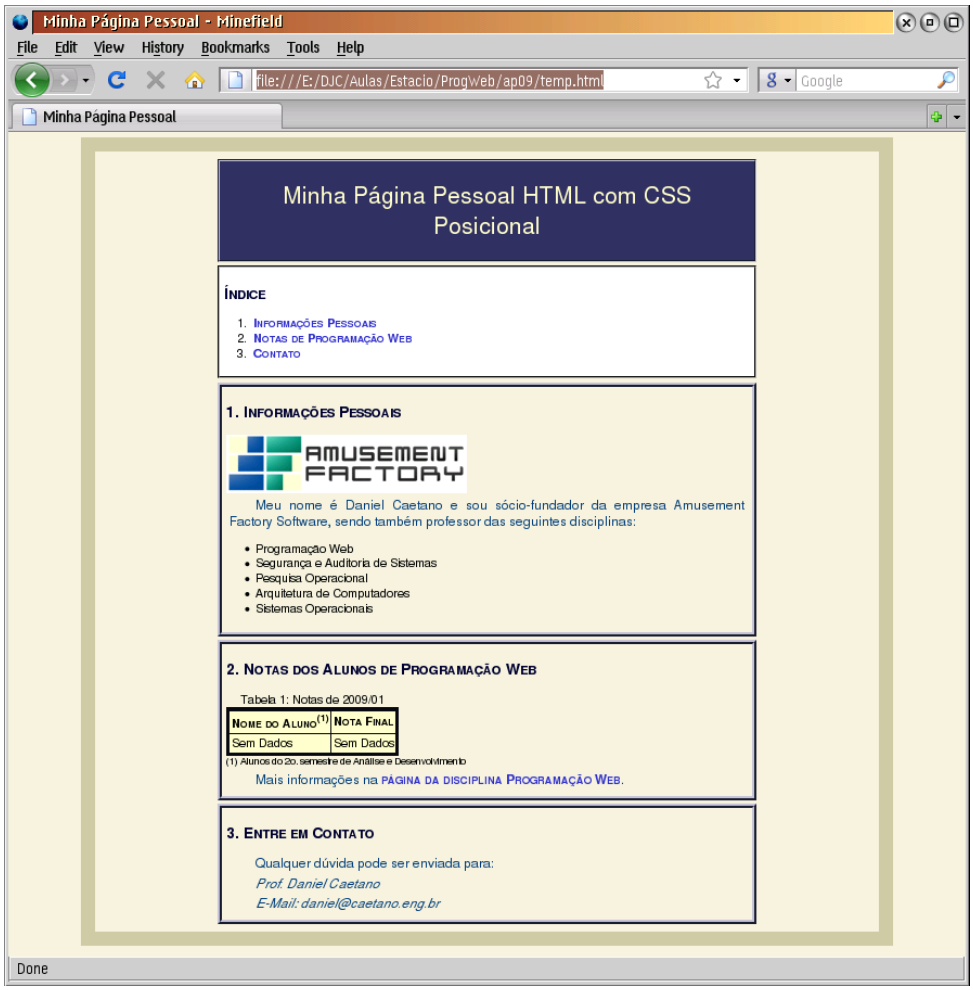
```
#titulo {
  overflow: hidden;
  background-color: rgb(50,50,100);
  margin-left: 15%;
  margin-right: 15%;
  border-style: groove;
  border-color: rgb(50,50,100);
  padding: 5px;
  }

#indice {
  overflow: hidden;
  margin-left: 15%;
  margin-right: 15%;
  background-color: rgb(255,255,255);
  margin-top: 5px;
  border-style: groove;
  padding: 5px;
  }

#secao1, #secao2, #secao3 {
  overflow: hidden;
  margin-left: 15%;
  margin-right: 15%;
  border-color: rgb(50,50,100);
  border-width: 6px;
```

```
    border-style: ridge;
    margin-top: 5px;
    padding: 5px;
    }
```

E o resultado obtido é:



Certo, usamos alguns efeitos interessantes. Mas o layout não mudou muito com relação ao que tínhamos antes. Isso ocorre porque apenas criamos as regiões com os DIVs e mudamos algumas poucas propriedades de cores e posição delas, mas não alteramos significativamente o *fluxo de informações* da página.

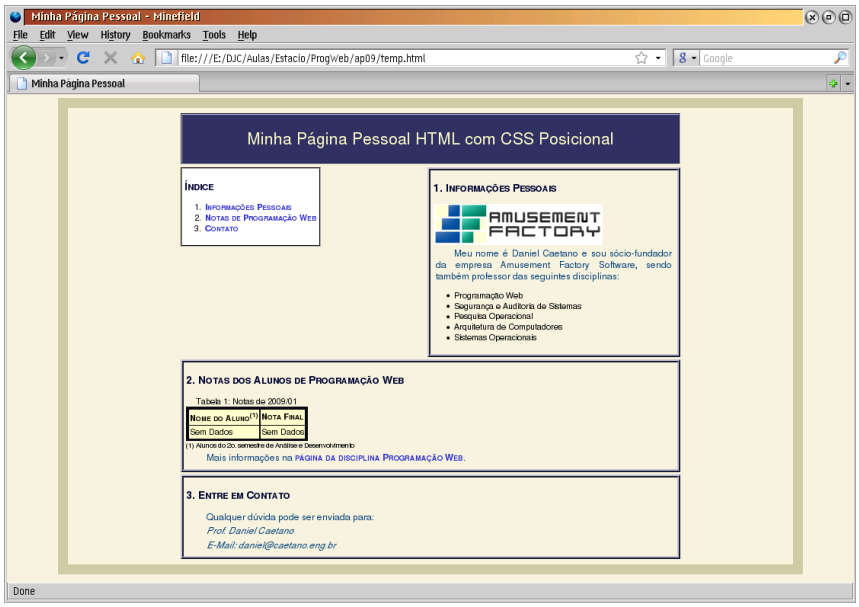
Isso significa que os dados estão sendo dispostos da mesma maneira que o padrão, isto é, de cima para baixo, um elemento em baixo do outro. Esse é o fluxo normal do HTML e do CSS.

Se quisermos, por exemplo, que o índice fique do lado esquerdo da tela, com informações ao seu lado direito, teremos que mudar o fluxo de apresentação das informações na tela. Quando queremos fixar um elemento em um dos lados e permitir texto do outro, dizemos que este elemento está **flutuando**, fixo a um dos lados. A propriedade que permite a

um elemento flutuar é a a propriedade **float**, que pode receber como valor as duas laterais: left (esquerda) ou right (direita). Vamos fazer com que o índice fique flutuando à esqueda:

```
#indice {
   float: left;
   overflow: hidden;
   margin-left: 15%;
   margin-right: 15%;
   background-color: rgb(255,255,255);
   margin-top: 5px;
   border-style: groove;
   padding: 5px;
   }
```

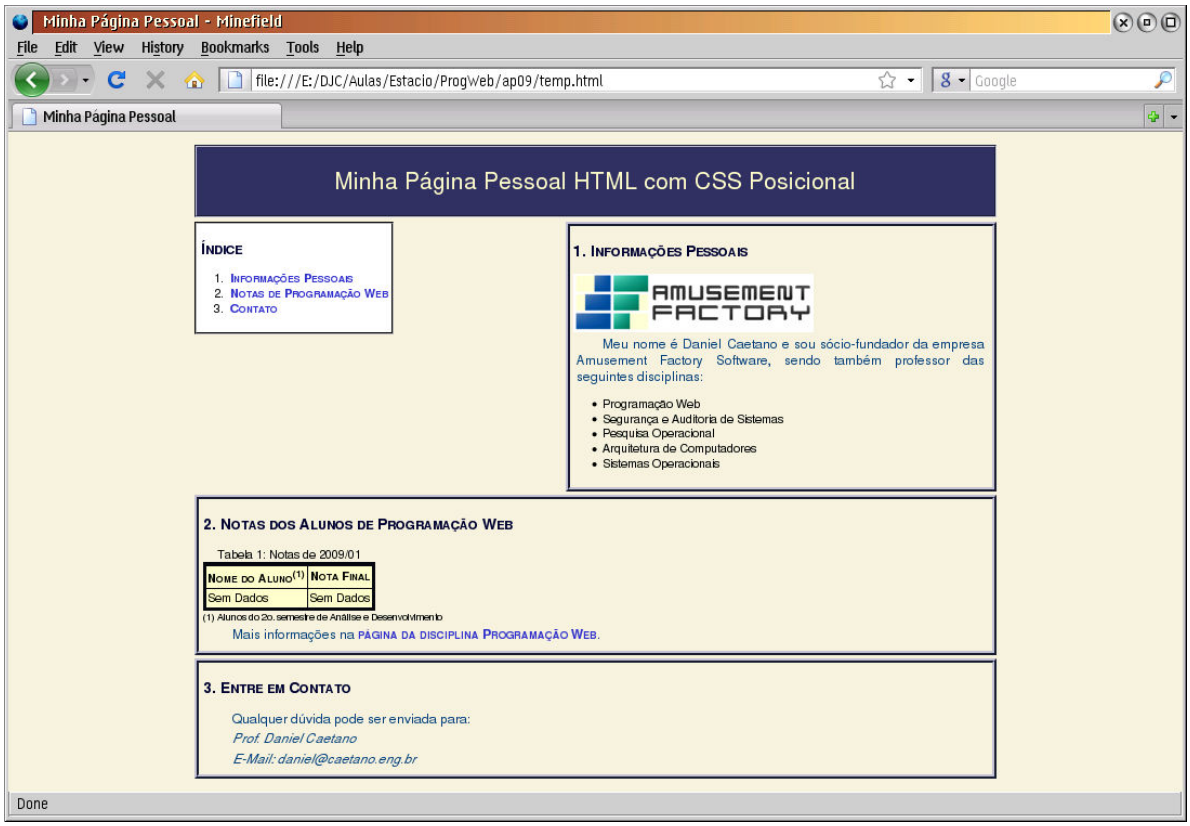
Faça a modificação e veja o que ocorreu:



Interessante, não? Pois bem, para ganharmos um pouco de espaço, vamos agora eliminar a borda que havíamos colocado no corpo da página antiga, e reduzir as margens:

```
BODY {
   font-family: verdana, arial, sans-serif;
   border-color: rgb(200,200,160);
   border-width: 20px;
   border-style: solid;
   margin-left: 10px;
   margin-right: 10px;
   background-color: rgb(240,240,220);
   padding: 10px;
}
```

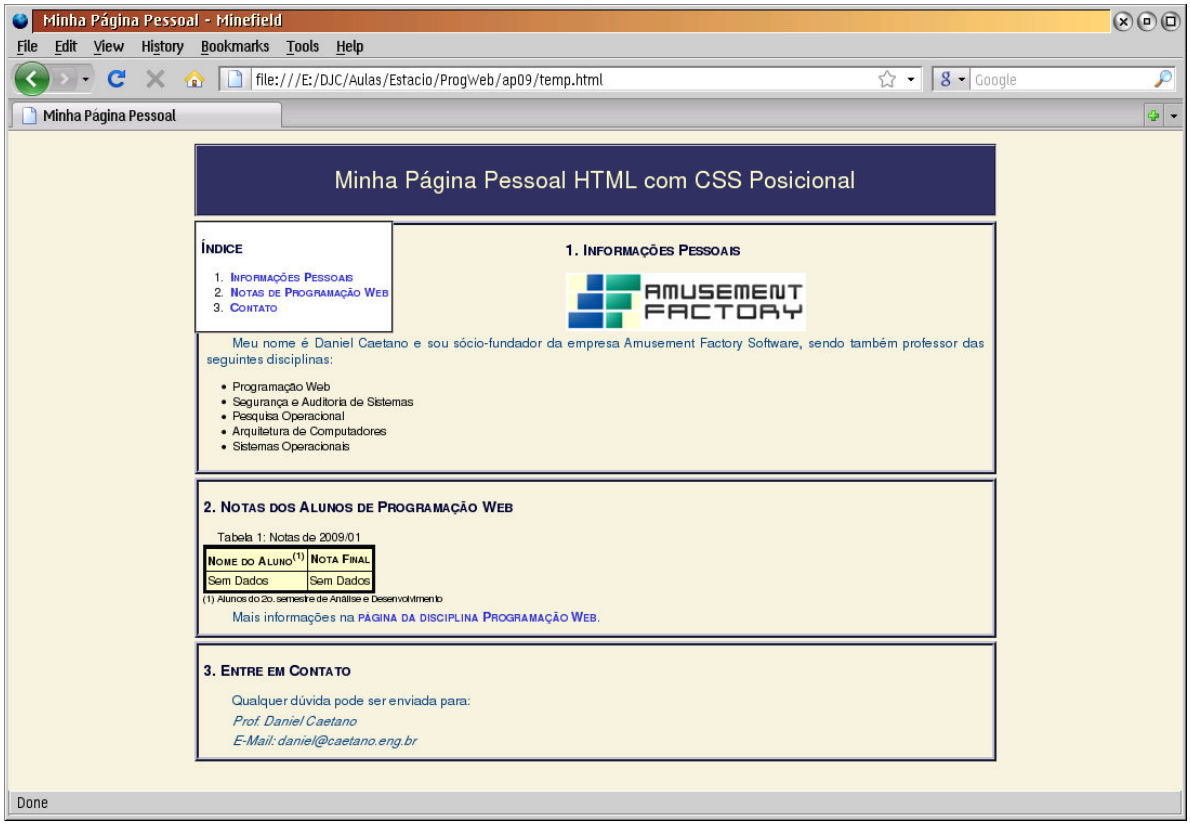
Observe o resultado:



Antes de continuarmos a "ajeitar" nossa página, lembram-se do segundo efeito colateral do "overflow: hidden"? Pois bem: retiremos o "overflow: hidden" das seções 1 a 3:

```
#secao1, #secao2, #secao3 {
  overflow: hidden;
  margin-left: 15%;
  margin-right: 15%;
  border-color: rgb(50,50,100);
  border-width: 6px;
  border-style: ridge;
  margin-top: 5px;
  padding: 5px;
  }
```

Recarregue a página e observe o resultado:



Note que a área da seção 1 ficou sobreposta à área do índice, mas o texto da seção 1 não ficou sobreposto (e nem sobrepôs) o conteúdo do índice. A esta altura você já deve ter identificado o "efeito colateral" do *overflow: hidden*, que é justamente o de impedir que regiões DIV diferentes se sobreponham horizontalmente.

Se não colocamos "overflow: hidden" nas seções 1, 2 e 3, a margem destas seções é contada a partir da borda da janela do navegador. Se colocamos "overflow: hidden" nestes elementos, a margem será contada a partir da borda dos elementos que estiverem aos seus lados; no caso, ao lado esquerdo da seção 1 existe o índice, então, ao colocar "overflow: hidden" na seção 1, fazemos com que a margem seja contada a partir do elemento do índice (e não da borda da janela).

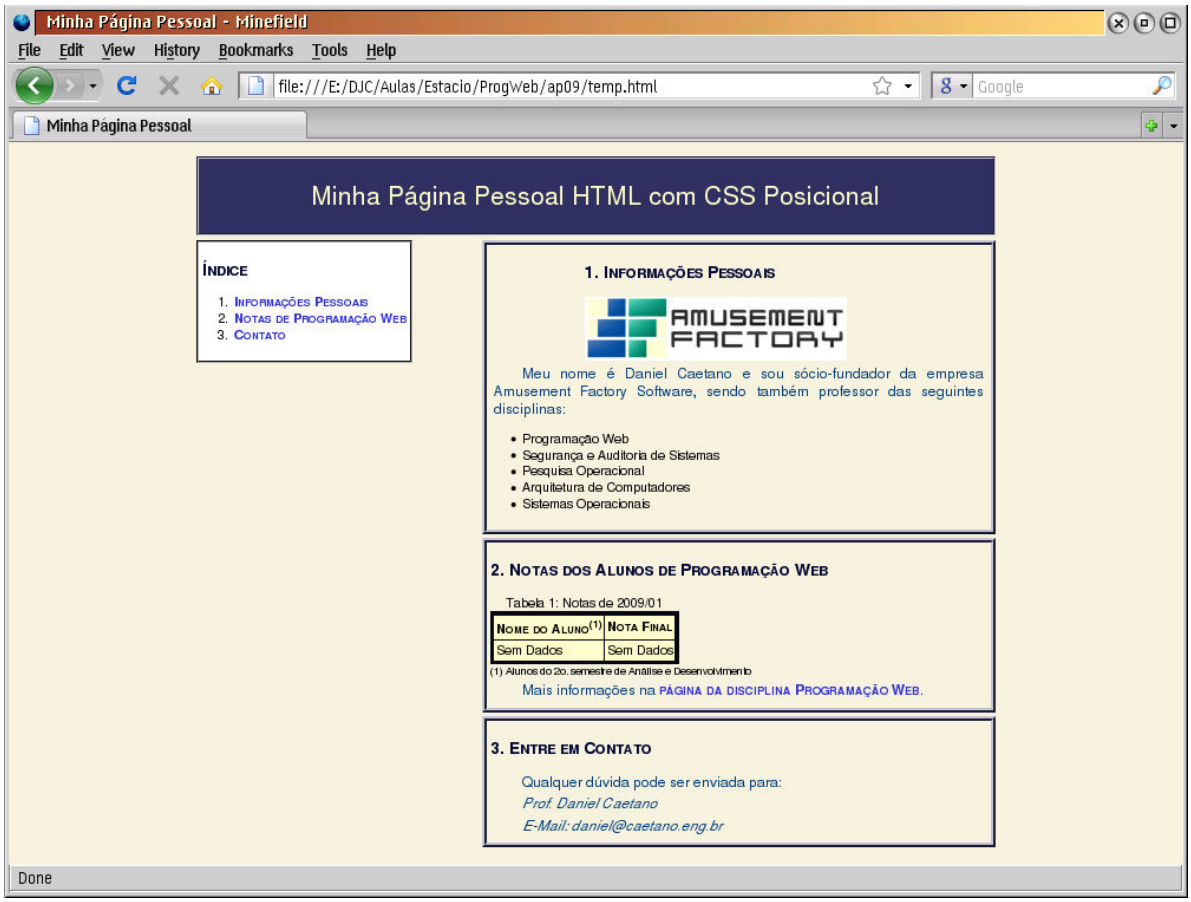
No caso das seções 2 e 3, como não há elementos à esquerda do bloco, a margem continua sendo contada a partir da borda da janela. É muito importante compreender estes "efeitos colaterais" de algumas tags para conseguir produzir exatamente o efeito que desejamos em nossas páginas.

Como queremos todas as regiões da direita alinhadas, vamos alinhá-las todas pela esquerda, a partir da margem e, por isso, continuaremos sem o "overflow: hidden" nas seções. Futuramente veremos como fazer este alinhamento com o uso do overflow: hidden.

Como sem o "overflow: hidden" as regiões estão sobrepostas porque a margens são iguais (a margem esquerda é de 15% da tela, tanto para o índice quanto para as seções 1 a 3), vamos modificar isso. Coloquemos uma margem esquerda de 40% para as seções 1, 2 e 3:

```
#secao1, #secao2, #secao3 {
  margin-left: 40%;
  margin-right: 15%;
  border-color: rgb(50,50,100);
  border-width: 6px;
  border-style: ridge;
  margin-top: 5px;
  padding: 5px;
  }
```

Observe o resultado:

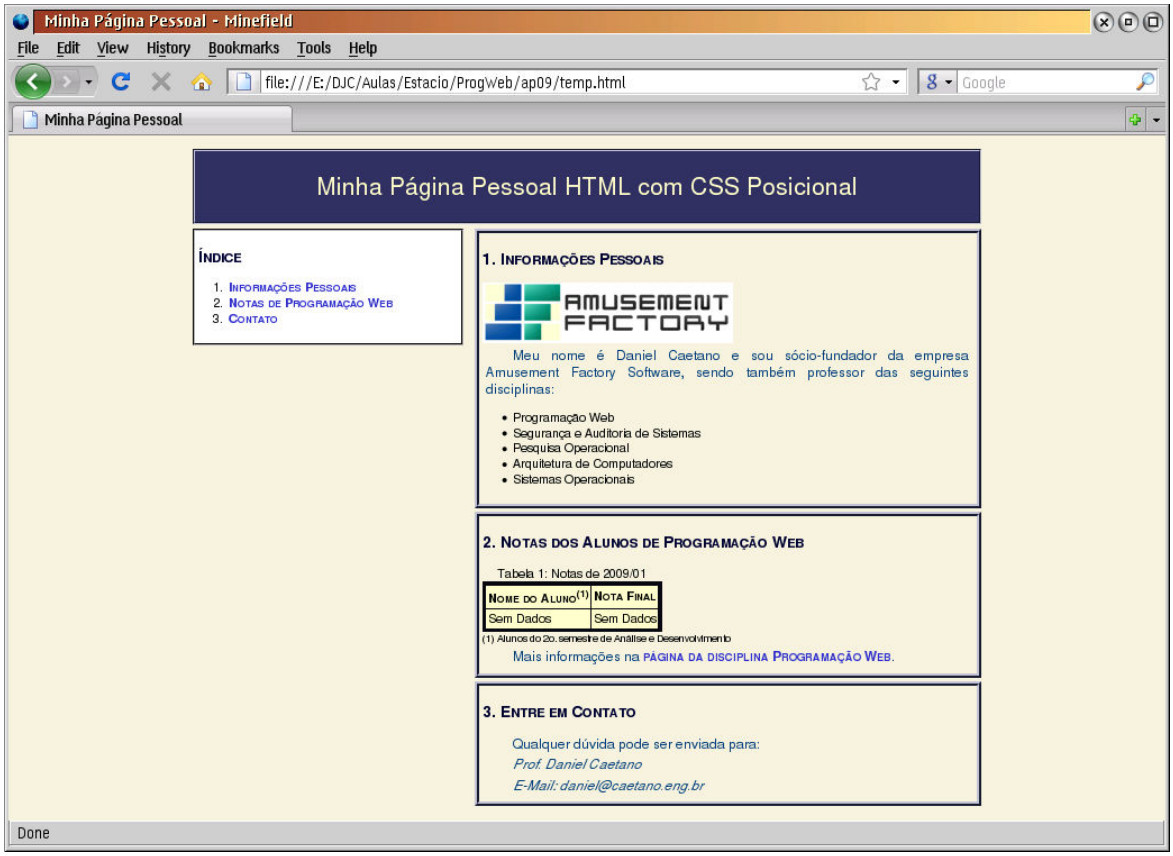


Melhorou bem! Observe que o texto da seção 1 parece meio "empurrado". Isso ocorre porque a seção índice tem uma margem esquerda definida. Vamos mdificar a seção índice de duas formas: a primeira, eliminando esta margem esquerda que está empurrando o nosso texto e a segunda, que será ampliar um pouquinho a largura do índice, usando para isso a propriedade width, com o valor 23%:

```
#indice {
  float: left;
  overflow: hidden;
  margin-left: 15%;
```

```
~~margin-right: 15%;~~
width: 23%;
background-color: rgb(255,255,255);
margin-top: 5px;
border-style: groove;
padding: 5px;
}
```

O resultado é este:



O nosso primeiro layout criado totalmente com CSS foi feito! Agora veremos alguns recursos bastante úteis do CSS, para facilitar a nossa vida. Futuramente voltaremos ao tópico de layouts HTML+CSS, explorando outros recursos destas poderosas tecnologias.

5. DEFININDO ESTILOS DENTRO DE BLOCOS

Conceitos Chave:

- Definindo estilos de tags dentro de seções
  - * tag_bloco tag { ... }
- Exemplos

Algumas vezes pode ser que exista a necessidade de definir uma propriedade diferenciada para um elemento dentro de outro. Por exemplo: foi definido como sendo, em geral, a aplicação de **bold** (negrito) no texto. Entretanto, dentro do parágrafo, gostaríamos que esta tag fosse definida como texto normal, mas com sublinhado. Isso é feito assim:

```
strong {
   font-weight: bold;
   }

p strong {
   font-weight: normal;
   text-decoration: underline;
}
```

Observe que esta definiçao segue um formato:

```
tag_bloco tag {
   ...
   }
```

Isso serve para definir estilos específicos dentro de uma parte da página, também:

```
#secao1 strong {
   font-weight: normal;
   text-decoration: underline;
}
```

6. MÍDIAS ALTERNATIVAS (OPCIONAL)

Conceitos Chave:
- Web é acessada por várias mídias
    * @midia { ... }

Algumas vezes o web master deseja criar alguns efeitos em sua página web que a inviabilizam para impressão (fundos escuros com letras claras, em geral, geram péssimas impressões em papel). Por esta razão, o CSS permite que sejam discriminadas as definições que serão ativas para cada tipo de mídia.

A forma de especificar isso é:

```
@tipo_de_midia {
  tag {
        atributo: propriedade;
  }
}
```

Em um mesmo arquivo CSS é possível definir todos os estilos para diferentes tipos de mídia. Os tipos de mídia possíveis são:

Tipo	Descrição
all	Usado para todos os tipos de mídia e dispositivos
aural	Usado para fala e sintetizadores de som
braille	Usado para dispositivos táteis em braile
embossed	Usado para impressoras braile
handheld	Usado para dispositivos pequenos ou de mão
print	Usado para impressoras
projection	Usado para apresentações projetadas, como slides
screen	Usado para as telas de computador
tty	Usado para mídias de fonte fixa (terminais, teletipos etc.)
tv	Usado para dispositivos tipo TV

7. BIBLIOGRAFIA

CASCADE Style Sheets, level 2 revision 1: CSS 2.1 Specification - W3C Working Draft 06 November 2006. Disponível em < http://www.w3.org/TR/CSS21/ >. Visitado em 21 de Dezembro de 2006.

W3 schools - CSS Tutorial. Disponível em < http://www.w3schools.com/ >. Visitado em 10 de Março de 2009.

RAMALHO, J.A. *HTML 4 Prático e Rápido*. Editora Berkeley, 1999.

BOENTE, A. *Programação Web Sem Mistérios*. Editora Brasport, 2006.

NIELSEN, J. *Projetando Websites*. Ed. Campus, 2000.