

Introdução ao HTML5 e seus novos elementos

1. Caminho até o HTML5

A WEB foi criada numa instituição (CERN) para troca de informações entre cientistas. Em toda a sua existência ela se expandiu e foi se transformando, como também se transformaram os meios de transporte da informação: as páginas de WEB.

Dada a sua utilização inicial, a linguagem HTML oferecia as facilidades necessárias para a criação de documentos com a aparência de artigos científicos. Quando foi necessária uma apresentação diferente, passou-se a utilizar tabelas com bordas invisíveis para permitir um livre posicionamento de elementos dentro de uma página. Isso é o que chamamos de layout baseado em tabelas, o que permitiu a criação de páginas mais bonitas e acabou facilitando a expansão da internet para além dos meios científicos.

A popularização da WEB mostrou que a forma de construir páginas apresentava diversos problemas. A uniformidade de páginas dentro de um site é muito difícil de ser mantida já que a informação de aparência estava misturada ao conteúdo das páginas. O layout de tabelas também não permite que seja definida com clareza a estrutura de um documento, o que é uma informação importante para mecanismos de busca. Por fim, este tipo de layout é um desastre para a acessibilidade.

A definição do padrão CSS (Cascade Style Sheets) permitiu que se retirassem as informações de aparência de dentro do documento HTML. Os estilos de posicionamento contidos na definição do CSS, aplicados sobretudo a tag <DIV> permitiu o livre posicionamento de elementos numa página sem a utilização de tabelas. Isso é o que chamamos de layout sem tabelas (tableless) e é a forma mais correta hoje em dia de se criar uma página.

```
<html>
  <head>
    <title>  </title>
  </head>
  <body>
    <div>  </div>
    <div>  </div>
    <div>  </div>
  </body>
</html>
```

Tab. 1.1 – Estrutura de uma página no HTML4

A utilização da tag <DIV> para posicionamento e organização dos documentos foi um grande avanço mas, para algumas aplicações, ela ainda tem um pequeno problema. Esta tag é muito genérica e, por poder conter qualquer coisa, não diz nada acerca do que ela contém. Esta informação sobre o significado (semântica) de cada ponto de um documento tem importância tanto em termos de organização do próprio documento quanto para auxiliar os sites de busca para melhor encontrar a informação desejada por um usuário.

Imagine um documento típico com uma área de cabeçalho, um menu, uma área de conteúdo e um rodapé. Este documento poderia ser criado com quatro tags <DIV> para cada uma destas áreas e a utilização do atributo id para diferenciar e identificar cada uma delas (Tab. 1.2). Isso resolve o problema de organização interna do documento mas não o problema

dos mecanismos de busca pois não há uma padronização do valor do identificador. O que alguém chamou de “cabecalho” outro poderia simplesmente chamar de “cabec”.

```
<html>
  <head>
    <title> </title>
  </head>
  <body>
    <div id="cabecalho"> </div>
    <div id="menu"> </div>
    <div id="conteudo"> </div>
    <div id="rodape"> </div>
  </body>
</html>
```

Tab. 1.2 – Estrutura com divisões identificadas

Baseado nos tipos de divisão mais comum de um documento, uma das mudanças introduzidas no HTML5 foi a criação de uma série de tags para implementar estas divisões.

2. Estrutura de um documento

Uma série de novas tags foram definidas no HTML5 para suprir esta necessidade de atribuir significado a blocos de conteúdo de um documento: <header>, <nav>, <section>, <article>, <aside> e <footer>. Estas tags não definem por si só nenhuma informação de posicionamento ou de aparência, o que importa nelas é apenas seu significado no documento:

- <header>

O elemento **header** contém informação introdutória para uma página ou para uma seção. Isso pode ser desde um simples cabeçalho de um documento até toda uma tabela de conteúdo.

- <nav>

O elemento **nav** é reservado para uma seção de um documento que contém links para outras páginas ou links para seções da mesma página. Nem todo grupo de links precisa estar contido dentro do elemento **nav**, apenas o menu primário.

- <section>

O elemento **section** representa um documento genérico ou a seção de um documento. Ele age mais ou menos como um **div**, separando partes de um documento. É possível incluir outras **section** ou **article** dentro de uma **section**.

- <article>

O elemento **article** representa uma parte de uma página que pode ficar sozinha com, por exemplo: uma postagem de blog, uma entrada de fórum, comentários submetidos por um usuário ou qualquer item de conteúdo independente. É possível incluir **section** ou **article** dentro de um **article**.

- **<aside>**

O elemento **aside** representa um conteúdo relacionado com a área principal do documento porém fora do fluxo principal do assunto. É muito comum de se encontrar em livros onde inclui notas explicações ou citações sobre o texto principal.

- **<footer>**

Este elemento serve para incluir informações de rodapé numa página ou em cada seção de uma página.

Na estrutura do HTML nenhum destes elementos é obrigatório e eles devem ser utilizados apenas para conter o tipo de dado cujo significado eles representam. Conforme dito anteriormente, estes elementos não definem por si só nenhuma informação de posicionamento ou de aparência, porém é comum representá-los como blocos dispostos conforme mostrado na figura 2.1. Um erro muito comum é imaginar que um documento qualquer em três colunas deve necessariamente ser representado desta forma. Muitas vezes cada coluna deve ser criada simplesmente como uma **section** ou mesmo como um **div**, dependendo do significado de seu conteúdo para o documento.

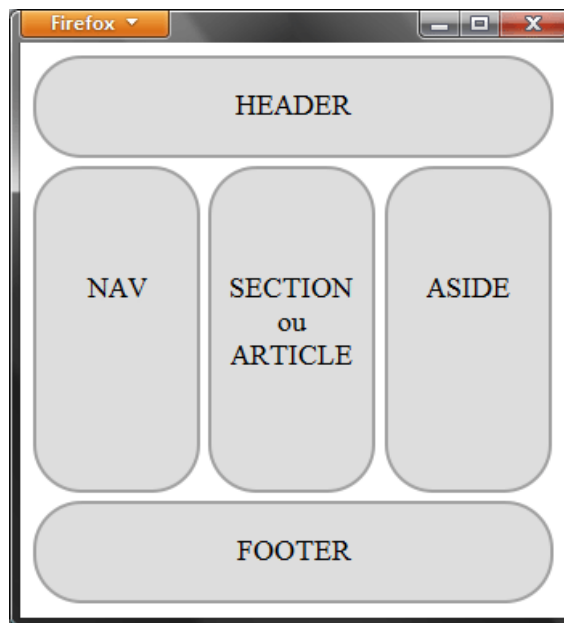


Fig. 2.1 – Layout padrão de um documento HTML5

3. O DOCTYPE

Antes de colocarmos isso em prática e sairmos por aí criando documentos em HTML5 é necessário incluirmos uma tag especial no início do nosso documento. Um documento HTML nada mais é que um documento XML (o HTML nada mais é que uma linguagem que segue a sintaxe XML) e por isso deve começar como um comando XML informando em qual “dialetto” de XML está codificado. Isso é feito simplesmente incluindo na primeira linha do documento:

```
<!DOCTYPE html>
```

Simplesmente isso já indica para o navegador que estamos com um documento HTML5. Nas versões anteriores de HTML já existia o DOCTYPE mas a sua sintaxe era mais complicada e

muitas vezes era mais simples omitir sua presença que explicá-lo. Na nova versão não é mais possível omiti-lo. Um documento para representar o layout padrão mostrado na Figura 2.1 pode ser escrito então conforme o código da Tabela 3.1 a seguir:

```
<!DOCTYPE html>
<html>
  <head>
    <title> </title>
  </head>
  <body>
    <header>
    </header>
    <nav>
    </nav>
    <section>
      <header>
      </header>
      <article>
        <hgroup>
        </hgroup>
      </article>
      <footer>
      </footer>
    </section>
    <aside>
    </aside>
    <footer>
    </footer>
  </body>
</html>
```

Tab. 3.1 – Estrutura de uma página com novos elementos do HTML5

A tag <hgroup> é mais uma das novas tags do HTML5 e é usada para agrupar cabeçalhos de vários níveis em um só bloco.

4. O CSS para isso funcionar

Ao chegar neste ponto do texto você já deverá ter entendido as mudanças introduzidas pelo HTML5 e deve estar ansioso para por a mão na massa e começar a utilizar as novas tags. Porém, se você digitar o código da Tabela 3.1 e mandar abrir em um navegador vai ficar decepcionado pois não encontrará nada parecido com o layout da figura 2.1. Para incluir informações de posicionamento e aparência é necessário utilizar estilos CSS. Diversas propriedades de CSS podem ser aplicadas a estas tags mas destacarei três delas pela sua utilidade para posicionamento:

a. Propriedade *display*

Uma questão fundamental para a aparência de qualquer elemento da linguagem HTML é se ele se comporta como parte de uma linha (*display* com valor *inline*) ou se ele se comporta como um bloco (*display* com valor *block*). A diferença entre os dois é basicamente a diferença entre a tag (inline) e a tag <div> (block).

Para que nossas novas tags possam ser exibidas com o layout mostrado na figura 2.1 é necessário que elas se comportem como *block*, e isso já é verdadeiro para a última versão dos navegadores mais populares. Porém versões mais antigas destes navegadores, mesmo aquelas já aptas a trabalhar com HTML5, tratavam estas novas tags como inline. Para estes

casos deve-se incluir o seguinte estilo no CSS da página (o que não causa nenhum prejuízo caso o navegador já trate estas tags como *block*):

```
<style>
  header, nav, section, article, aside, footer {
    display: block;
  }
</style>
```

b. Propriedade *float*

A propriedade *float* indica se um elemento ficará preso a uma lateral do bloco que o contém e a qual lado (*left* ou *right*).

```
<style>
  nav, section {
    float: left;
  }
  aside {
    float: right;
  }
</style>
```

c. Propriedade *clear*

A propriedade *clear* indica se um elemento não vai permitir nenhum elemento agarrado a seu lado direito (*right*), esquerdo (*left*) ou a nenhum dos dois (*both*).

```
<style>
  footer {
    clear: both;
  }
</style>
```