**Escrevendo menos e fazendo mais**

O que mundou para melhor foi o Doctype antes era quase um monologo em um tag, agora temos uma simplificação, veja como era feito antes:

Versão 4

<!DOCTYPE HTML 3 PUBLIC "-//W3C//DTD HTML 4.01//EN" 4 "http://www.w3.org/TR/html4/strict.dtd">

Versão XHTML

<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" 8 "http://www.w3.org/TR/xhtml1/DTD/xhtml1-transitional.dtd">

Agora na versão 5

<!DOCTYPE html>

Outra coisa que mudou foi ao referenciar outras bibliotecas, antes teríamos que dizer de que se trata o arquivo, agora isso é opcional.

Opcional

<link rel= "stylesheet" href= "normalize.css" type= "text/css" />

<script src= "prefixfree.min.js" type= "text/javascript" ></script>

Pode ser escrito assim:

<link rel= "stylesheet" href= "normalize.css" >

<link rel= "stylesheet" href= "normalize.css" >

Tags que não precesam de fechamento não precisam mais da barra (/)

Opcional

<br />

Pode ser escrito assim:

<br>

Outra característica que mudou foi que você personalizar tags que não existem no HTML.

Um exemplo muito comum é fazer com que alguma biblioteca leia uma tag à procura de alguns atributos específicos, como por exemplo, um atributo indicando que a requisição de um link deve ser feita assincronamente via Ajax.

<a href= 'posts/2' remote= 'true' > Apagar </a>

O atributo remote não existe na definição da tag a . Esta abordagem funcionaria em páginas de HTML 4, por exemplo, mas o seu código estaria inválido de acordo com a ferramenta de validação do W3C.

Uma forma de resolver isso são com atributos personalizados, que fazem parte do domínio do seu projeto e ao mesmo tempo mantêm o seu código válido.

Por exemplo, para definir um link que irá apagar um post do seu blog, podemos utilizar algumas configurações via atributos personalizados com o prefixo data-.

<a href= 'posts/2' data-remote= 'true' data-method= 'delete' data-confirm= 'Você tem certeza disto?' > Apagar </a>

Desta forma, o seu JavaScript pode ler os atributos de data-\* e executar o código necessário para apagar o post, pedindo uma confirmação do usuário com uma mensagem personalizada.

**Tags novas para elementos antigos**

Para dar mais estrutura e semântica as paginas o HTML5 trouxe tags especificas para estrutura de um site, deixando as divs para a estrutura o conteúdo.

Veja algumas delas:

section - utilizado para representar uma seção genérica, geralmente com um cabeçalho próprio e o seu conteúdo;

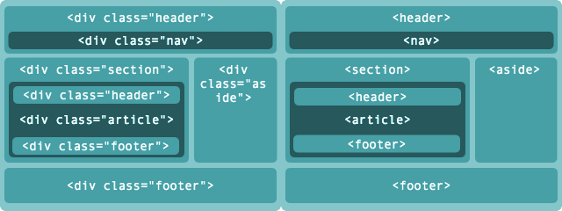
nav - representação de um bloco principal de links de navegação - nem todo grupo de links deve ser tratado como um nav ;

aside - a tag pode ser utilizada para representar uma seção de conteúdo secundário ou auxiliar outro pedaço de maior importância. Citações, links de referência ou notas adicionais, por exemplo;

header - referente ao cabeçalho de uma seção específica (ou da própria página), contendo títulos, introduções e outros elementos similares;

footer - o rodapé referente a um bloco de conteúdo;

article - identifica o conteúdo em si, como uma notícia de um portal, um post em blog ou um comentário em uma lista de comentários.



O antes de como era feito do lado esquerdo e como é feito hoje em dia.

**Refatoração da página de São Paulo**

Para a classe .place você pode substituir por artice, para a classe .sidenote podemos utilizar a tag aside e para o rodapé substituir a classe .footer para a tag footer.

Todas essas mudanças devem também refletir no CSS.

Também podemos substituir a classe .places por section e colocar o header para o H1 mais tudo isso fica a critério do desenvolvedor.

Para forçar essas tags novas em navegadores antigos tem um script em Javascript que pode ser encontrado em:

<https://github.com/aFarkas/html5shiv/>

**Seja pragmático**