2
Conceitos básicos

Para tornar a leitura deste documento mais simples, é necessário entender alguns conceitos importantes sobre a criação e a manipulação de documentos HTML. Por esse motivo, na Seção 2.1 apresentamos alguns conceitos sobre a HTML, um resumo sobre a história da linguagem e suas principais características. Na Seção 2.2 são apresentadas as formas de visualização de um documento HTML. Para concluir este capítulo, na Seção 2.3 são apresentados alguns conceitos sobre a manipulação e tratamento de um documento HTML em uma árvore DOM, principal estrutura de manipulação que é utilizada neste trabalho.

É importante ressaltar que os conceitos apresentados neste capítulo são utilizados na discussão das técnicas exploradas nesta dissertação.

2.1
A linguagem HTML

A HTML *(Hyper Text Markup Language)* foi diretamente influenciada pela SGML *(Standard Generalized Markup Language)*, sendo considerada uma SGML logo em sua primeira versão publicada (HTML2.0) em 1995. Em 1997, junto ao crescimento da Web, foram feitas várias mudanças na descrição da HTML, pois navegadores, principalmente o *Netscape*, adicionaram marcações que não eram descritas na especificação da linguagem. Com essas mudanças, a normalização e agregação de marcações propostas pelos navegadores, a HTML4.0 tornou-se uma linguagem madura, mantendo sua estrutura e suas marcações até hoje. (Connolly, 2000, Longman 1998)

Depois do ano 2000, a Web voltou a ter um crescimento acelerado, trazendo novidades para a HTML, como o surgimento da XHTML *(eXtensible Hypertext Markup Language)* (Pemberton, 2000), uma reformulação da HTML4.01. Essa nova especificação possibilitou a utilização de *parser* XML em documentos XHTML. Um exemplo mais recente desse crescimento é o lançamento da quinta versão da linguagem, HTML5, que apresenta diversos novos elementos como vídeo,áudio, section, article, time, dentre outros.

Toda a evolução da HTML tornou a Web mais dinâmica, porém criou

PUC-Rio - Certificação Digital Nº 0821371/CA

também um universo confuso, onde diversos padrões são encontrados em conjunto. É comum a existência de documentos sem a especificação do padrão utilizado para sua criação. Também é comum a utilização de recursos incompatíveis de duas versões diferentes em um mesmo documento. A existência de documentos antigos e a falta de conhecimento dos autores são os principais problemas encontrados quando trabalhamos com documentos HTML da Web.

Para que seja possível o aprofundamento em tarefas que envolvem a manipulação de documentos HTML, é necessário entender a estrutura desses documentos. A seguir são listados os principais elementos HTML e seus atributos. Por ser a referência para especificações mais modernas, utilizamos a especificação da HTML4.1, que pode ser encontrada no site da W3C[1].

Cada marcação no documento é apresentada entre < (menor que) e > (maior que) e é chamada de *tag*. Por exemplo, <body> é uma *tag* presente em todos os documentos. Um **elemento** é descrito por três partes: *tag* inicial (por exemplo, <body>), conteúdo (que pode ser um texto ou conter elementos) e *tag* final (por exemplo, </body>). Cada **elemento** pode conter atributos descritos dentro da sua tag inicial por um par nome e valor, como em <*body style="color: blue"*>, que define a cor do texto contido no elemento HTML como azul. Todos os elementos, atributos e tipos de atributos da HTML são definidos pela W3C.

Um ponto importante quando se observa a descrição da HTML é que essa é destinada a descrever os elementos necessários para a autoria de hipertexto, isto é, documento textual com elos *(links)*. No entanto, atualmente, notamos sua utilização para a criação de documentos com grande volume de diagramação visual (*layouts*). Essa adaptação na utilização da linguagem implica em novas necessidades, forçando a extensão da linguagem. Porém, enquanto novos elementos e extensões não são disponibilizados, os autores, muitas vezes, utilizam alguns elementos para funções que eles não foram inicialmente planejados.

Os elementos apresentados pela W3C são separados em:

- Textos (*Text*): elementos que manipulam o conteúdo textual como alinhamento do texto, utilizando tags como <p>, <cite>, , <var>, <pre>, <ins>.
- Listas (*Lists*): elementos de criação de listas utilizando tags como <ul> e <li>, <dl> e <dt> ou <dd> e <ol>.;

[1]http://www.w3.org/TR/html4/

PUC-Rio - Certificação Digital Nº 0821371/CA

- Tabelas (*Tables*): elementos utilizados para apresentar informação de forma tabular, utilizando a tag <table> e diversas outras como <caption>, <th>, <tr> e <td> para a organização das tabelas ;
- Ligações (*Links*): dois elementos que criam elos entre documento sendo as tags, <a> e <link>, para a criação de elos entre documentos distintos ou em um mesmo documento;
- Objetos, Imagens e Applets (*Objects, Images, and Applets*): elementos que possibilitam adicionar objetos multimídia em um documento HTML, como imagem (<img>), videos, applets, código, dentre outros (<object>);
- Folha de estilo (*Style Sheets*): elementos que proporcionam adicionar estilos às páginas (page designers), o que resolveu alguns problemas relacionados à diagramação dos documentos HTML. Esse estilo pode ser adicionado diretamente junto aos atributos dos elementos (atributo style) ou pode ser anexado ao documento (com o elemento <link>);
- Alinhamento, Tamanho de Fonte e Regras Horizontais (*Alignment, font style, and horizontal rules*): elementos de estilo, porém, atualmente quase todos os elementos de estilo não são mais recomendados, pois a W3C sugere o uso de folhas de estilo;
- Quadro (*Frames*): que apresenta elementos (<frame> e <iframe>) que permitem criar uma ou várias regiões de um documento que apresentam outros documentos;
- Formulários (*Forms*): elementos especiais chamados de controle (buttons, input text, checkboxes, radio buttons, menus, etc), esses elementos possibilitam a aquisição de informação e seu envio;
- Extensões (*Scripts*): elementos que tornam possível a adição de código que será executado na máquina do cliente que estiver visualizando o documento HTML. Esses *sctipts* podem ser escritos em várias linguagens como vbscript, javascript, tcl e também podem modificar o documento em tempo de execução, como alterar o tamanho de uma fonte, a cor de fundo ou qualquer outra informação do documento;

PUC-Rio - Certificação Digital Nº 0821371/CA

2.2 Formas de visualização de um documento HTML

Um documento HTML pode ser visualizado de várias formas, em forma de código (no seu formato original), em forma de grafo (onde é possível ver as ligações de um documento), em forma de árvore (utilizando estruturas como DOM, SAX) ou até mesmo várias formas de renderização definidas pelo tipo de mídia que o documento se destina (tela, impressora, tv, projetor, dentre outras). A maneira mais utilizada é a visualização em um navegador (tela), portanto várias etapas de processamento são comumente realizadas para sua exibição. É interessante notar que dentre as várias formas de visualização, existem etapas de processamento que se repetem. Normalmente o processamento do documento é feito em cascata, sempre adicionando informações às estruturas em memória que, no final, gera a visualização desejada. A Figura 2.1, retirada de (WebKit), ilustra o fluxo de processamento de um documento HTML no WebKit, o *framework* que gera a visualização de documentos HTML, utilizada pelo Safari e pelo Chrome. (WebKit, Chrome)

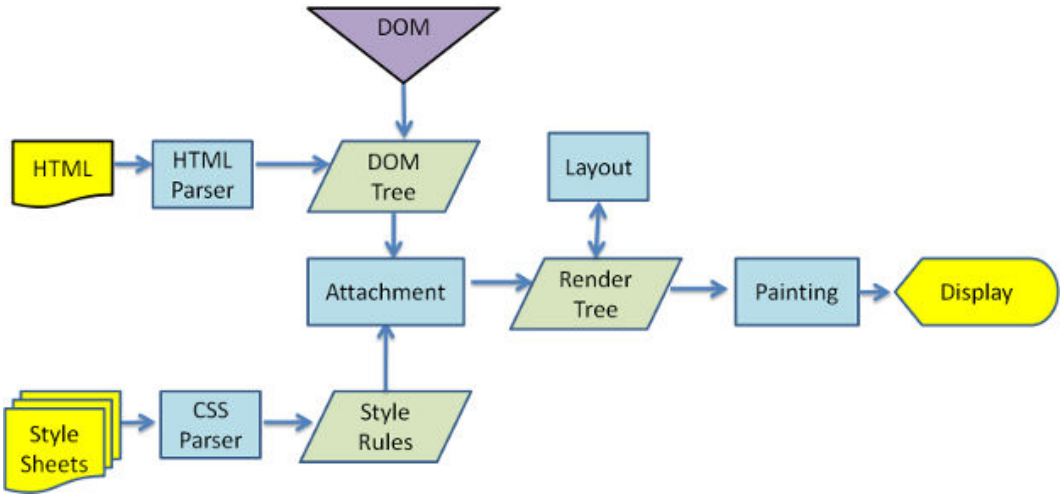

Figura 2.1: Fluxo principal do WebKit

Como pode ser observado no fluxo de processamento do WebKit, o resultado apresentado pelo navegador envolve outros documentos que estão ligados/anexados ao documento HTML, como a folhas de estilo (*style sheets*) por exemplo. Para trabalhar com o documento sem a necessidade de aplicar todo o processamento utilizado para gerar a visualização do documento HTML em um navegador, é interessante entender o que são alguns componentes do fluxo. A seguir é explicado cada item do fluxo, para, em seguida, apresentarmos a ordem de processamento existente no fluxo, tornando mais clara a necessidade de cada passo para a geração da visualização e tornar possível o entendimento

PUC-Rio - Certificação Digital Nº 0821371/CA

de como cada etapa agrega tempo de processamento, pois tem maior custo computacional.

- HTML - Documento HTML que será processado, descrito conforme a especificação da W3C;
- HTML Parser - Parser que verifica a sintaxe e semântica do documento e constrói em memória a árvore DOM, que representa o documento;
- DOM Tree - Estrutura em memória do documento HTML, a descrição dessa estrutura em árvore é disponibilizada pela W3C. Por ser uma estrutura importante, essa será descrita com detalhes na Seção 2.3;
- Style Sheets - Folhas de estilo vinculadas ao documento HTML, seja através do elemento LINK ou mesmo apresentada dentro do documento como conteúdo do elemento STYLE;
- CSS Parser - Parser que verifica a sintaxe e semântica do documento CSS criando uma estrutura em memória Style Rules que será anexada à árvore DOM;
- Style Rules - Estrutura em memória que armazena as regras contidas no CSS para que essas possam ser aplicadas ao documento;
- Attachment - Liga as regras existentes na estrutura Style Rules à árvore DOM;
- Render Tree - Organiza a árvore DOM, gerada pelo documento, junto às modificações providas pela Style Rules. Essa é a estrutura que guarda todas as informações necessárias para que seja possível gerar a visualização do documento pelo navegador. A W3C especifica uma estrutura DOM que deve armazenar essas informações, porém não foi encontrado junto à documentação se a estrutura DOM level2 style é utilizada nessa etapa, ou se o WebKit implementa sua estrutura seguindo uma especificação própria;
- Printing - Gera a informação necessária para exibir o documento no navegador, como coordenada onde cada elemento deve aparecer e tamanho em pixels de cada elemento;
- Display - Módulo responsável pela exibição do documento tornando possível sua visualização pelo usuário final.

É interessante mencionar que foram buscados *parsers* para a criação da estrutura DOM level2 style (chamada de Render Tree pelo Webkit), a estrutura que junta o documento HTML com as informações contidas no CSS, porém nenhum *parser* foi encontrado. Foi possível notar que diversos trabalhos, que

PUC-Rio - Certificação Digital Nº 0821371/CA

utilizam da informação provida por esse nível de processamento, utilizaram ferramentas como o WebKit para obter a informação de estilo anexada à árvore DOM. A principal crítica a esse tipo de processamento é a necessidade de gerar a visualização final (*Display*) para obter informações teoricamente intermediárias. Esse tipo de abordagem é desvantajosa, já que é necessário aplicar etapas de processamento, não triviais, que são ignoradas, gerando desperdício de processamento.

Vale ressaltar que cada componente do fluxo é um processamento independente, então teoricamente é possível interromper o processamento ou aplicar somente algumas etapas, respeitando a sequência de pré-requisitos existentes, apresentada na Figura 2.1.

Abaixo listamos as etapas do processamento que geram pontos de trabalho interessantes, antes de aplicar todo o processamento necessário para a exibição de um documento HTML em um navegador.

1. É possível utilizar somente a árvore DOM do documento, antes de anexar as informações contidas na folha de estilo ou scripts.
2. É possível utilizar a estrutura DOM junto às informações obtidas na folha de estilo, o que torna necessário o processamento da folha de estilo e o casamento das regras com os nós específicos da árvore DOM.
3. Utilizar a Render Tree, que contém todas às informações do documento já com as regras da folha de estilo aplicadas. Esse ponto de trabalho é claramente mais caro, já que é necessário o processamento do HTML e do CSS para então aplicar as informações do CSS no documento.
4. É possível utilizar a visualização do documento, depois que ele foi impresso. Com isso, pode-se obter informações visuais e detalhes de posicionamento exato, como x e y de um elemento na tela. Naturalmente esse ponto de processamento é o mais caro, já que todas as informações anteriores são necessárias e além disso é calculado o posicionamento de cada elemento na tela.

Nesta dissertação utilizamos apenas a árvore DOM, não aplicando nenhum outro tipo de processamento. Essa abordagem é adotada, pois a criação da árvore DOM tem um custo computacional pequeno, se comparado à adição das informações da folha de estilo ou ainda do processo completo, utilizando a imagem gerada pelo navegador. Acreditamos que nossa abordagem deve ser aplicada como um pré-processamento e por esse motivo a velocidade de processamento de cada documento é importante.

Na Seção 2.3 é apresentada a árvore DOM, sendo exemplificada sua criação. Além disso, também são apresentadas algumas informações que se

PUC-Rio - Certificação Digital Nº 0821371/CA

fazem necessárias para o entendimento de porque é possível extrair informações importantes de sua estrutura.

2.3 Document Object Model (DOM)

Document Object Model (DOM) define uma interface comum para a manipulação de documentos HTML. Essa interface fornece o acesso lógico ao conteúdo, a estrutura e ao estilo de uma página através de uma árvore ordenada de nós de elementos, textos, comentários e atributos. As principais informações que são apresentadas de um documento HTML estão nos nós de elementos e texto da árvore DOM. Atributos e comentários são informações não visíveis quando o documento HTML é renderizado, sendo as informações visuais, em maioria, contidas nas folhas da árvore DOM.

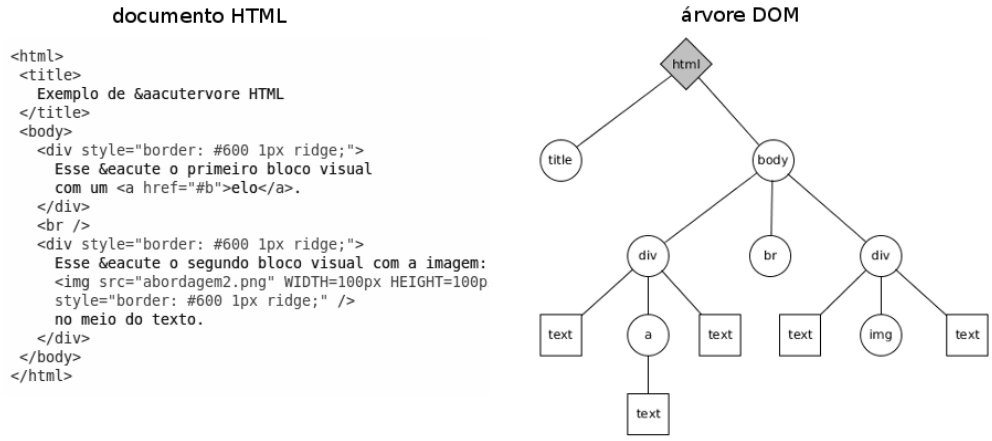

Figura 2.2: Comparação entre o documento HTML e sua árvore DOM

Na Figura 2.2, são apresentados um documento HTML e sua representação DOM. Note que a árvore DOM respeita a ordem de abertura dos elementos, adicionando os elementos na árvore na mesma ordem em que aparecem no documento HTML. Essa noção de ordem também é mantida na renderização do documento HTML, quando é gerada a representação visual do documento em um navegador. Isso pode ser observado na Figura 2.3, que apresenta a visualização da árvore DOM em um navegador.

Note que nós subsequentes na árvore DOM formam regiões visuais subsequentes como o bloco 1 e o bloco 2 da Figura 2.3. É importante reparar que algumas marcações como o br não geram blocos visuais. Esse padrão também é muito importante, pois cria um vínculo entre a estrutura do documento HTML e sua representação visual. Esse vínculo motiva a criação

PUC-Rio - Certificação Digital Nº 0821371/CA

de procedimentos que sejam capazes de obter informações aparentemente visuais da estrutura do documento, diminuindo a quantidade de processamento necessário para a identificação de segmentos em documento HTML.

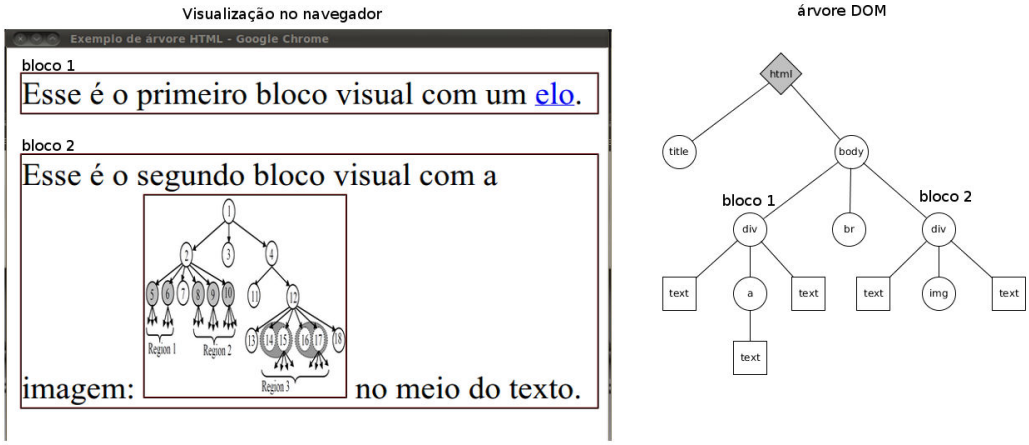

Figura 2.3: Comparação entre a visualização do documento em um navegador e sua árvore DOM

PUC-Rio - Certificação Digital Nº 0821371/CA