Análise e Desenvolvimento de Sistemas

Ferramentas de Desenvolvimento *Web*

Aula 2

Prof. Silvio Bortoletto

CONVERSA INICIAL

Olá, caro aluno!

Preparado para nossa segunda aula?

O Desenvolvimento *Web* tem crescido a cada dia mais. Com o advento das novas tecnologias cada vez mais pessoas possuem acesso à *Internet*, por meio de vários dispositivos. Se, há algum tempo, o desenvolvimento *Web* era considerado a "profissão do futuro", hoje em dia ele já é uma realidade em nossa sociedade. Criar *Websites* dinâmicos, responsivos e funcionais se torna cada vez mais complexo, exigindo esforço e dedicação!

Bons estudos!

CONTEXTUALIZANDO

A relevância de se estudar o desenvolvimento *Web* se dá devido à ampliação do acesso à *Internet* e sua abrangência no mundo atual. A cada dia mais pessoas possuem acesso direto à rede, utilizando-se de um computador, *tablet* ou *smartphone*.

A *Internet* já é uma realidade e estamos inseridos nesse contexto, pois, à medida que mais pessoas possuem acesso, precisamos estar sempre atentos às novidades e tendências do mundo da tecnologia, para que possamos compreender de que maneira essas tendências influenciam nosso cotidiano.

Para criarmos uma página *Web*, precisamos conhecer a linguagem de marcação HTML – é por meio dela que todas as páginas *Web* são construídas.

Através do HTML podemos criar títulos (`<h1>`, `<h2>`...), parágrafos de texto (`<p>`), links para outras páginas (`<a>`), listas (`<ol>`, `<ul>`), e organizar a página de acordo com o design proposto.

Através da linguagem de formatação CSS podemos melhorar a interface das páginas *Web*, aplicando formatações, animações e interações com o

usuário. É também através da identidade visual que sua página poderá se destacar.

PESQUISE

Seletores

Bem, como o assunto desta aula é o CSS, vamos iniciar com uma pergunta bem geral:

Você sabe exatamente o que é o CSS?

Que tal pesquisar o significado dessa sigla e o que ela representa?

http://www.tecmundo.com.br/programacao/2705-o-que-e-css-.htm

Toda vez que usamos o CSS, nós usamos os **seletores**. Isso porque as formatações CSS serão definidas pelos seletores, que poderão ser de 3 tipos:

- **Seletores de elementos**

  É um comando usado para especificar um estilo para determinado elementos (ou pseudo-elementos) presentes no documento HTML.

- **Seletores de classe**

  Seleciona elementos com uma classe específica aplicada. Exemplo: `.destaque` seleciona todos os elementos com a classe "destaque".

- **Seletores de id**

  Seleciona o elemento com a id especificada. Exemplo: `#cabecalho` irá selecionar o elemento com o id "cabecalho". Cada id é único e não pode ser repetido no mesmo documento.

Para todos os seletores, existem regras de utilização e herança. A concatenação, como é chamada, poderá combinar formatações, deixando sua página única!

Vamos, agora, conhecer um exemplo de funcionamento dos diferentes seletores?

http://www.w3schools.com/cssref/trysel.asp

E agora, acesse o vídeo introdutório disponível no material *on-line* para ver o que o professor Sílvio tem a acrescentar ao nosso estudo dos seletores!

Ordem de aplicação

As formatações definidas pelo CSS seguem uma ordem de aplicação, de acordo com a importância atribuída. As regras podem se sobrepor, dependendo do tipo de concatenação que é feita. Estabelecer uma ordem é importante sobretudo quando o trabalho é desenvolvido por mais de um *designer* ou programador.

Algumas estratégias simples podem ser utilizadas para deixar o seu CSS mais organizado, consistente e de fácil manutenção. Vamos conferir que estratégias são essas?

http://tableless.com.br/6-estrategias-para-melhorar-a-organizacao-do-seu-css-2/

O arquivo HTML interpreta as formatações CSS de acordo com prioridades:

- Utilização de `!important`
- Formatação `Inline`
- Formatação `Head`
- Folha de estilos `<link>`

A prioridade também depende da posição da formatação no documento, tanto nas *tags* HTML quanto no arquivo CSS.

Pode ocorrer de o arquivo ser muito complexo e as regras acabarem se sobrepondo. Mas você sabia que há estratégias para que você possa apontar que regra vale para cada situação? Há, inclusive, um cálculo para isso, como você pode perceber com a leitura da matéria a seguir!

http://www.richardbarros.com.br/blog/css/como-saber-qual-regra-css-tem-prioridade

Agora, retomemos o conteúdo deste tema com a fala do professor Sílvio sobre o assunto. Acesse o material *on-line*!

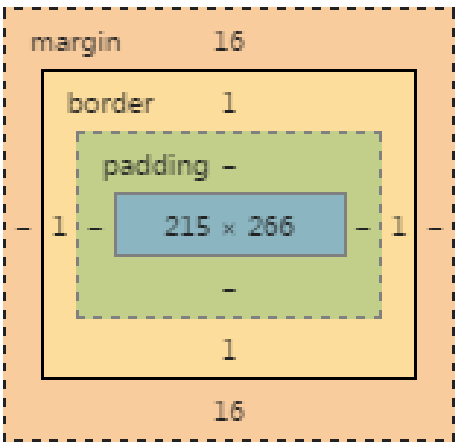
Box Model

Na verdade, todos os elementos do HTML podem ser considerados boxes. No CSS, o *Box Model* diz respeito ao *design* e leiaute. Foi criado para observação dos atributos ligados aos elementos. *Box Model* é um diagrama imaginário em CSS, criado para contextualizar cada elemento dentro do DOM.

DOM? Mas o que é DOM?

http://tableless.com.br/tenha-o-dom/

A imagem a seguir representa um *Box Model*. Vejamos:



- **Content** – É onde o conteúdo do box aparece, com textos e imagens.

- **Padding** – É a área transparente ao lado do conteúdo que o separa do resto.
- **Border** – É a borda que envolve o preenchimento e o conteúdo.
- **Margin** – Limpa uma área fora da borda. A margem é transparente.

O *Box Model* nos permite adicionar uma borda em torno de elementos e, também, definir o espaço entre os elementos.

```
div {
    width: 300px;
    padding: 25px;
    border: 25px solid navy;
    margin: 25px;
}
```

Largura e altura de um elemento

Para definir corretamente a largura e altura de um elemento em todos os navegadores, você precisa saber como o modelo de caixa funciona.

Quando você define as propriedades largura e altura de um elemento com CSS, você apenas define a largura e altura **da área de conteúdo**. Para calcular o tamanho total de um elemento, você também deve adicionar espaçamentos, bordas e margens.

Suponha que você deseje que um elemento `<div>` tenha uma largura total de 350px:

```
div {
    width: 320px;
    padding: 10px;
    border: 5px solid gray;
    margin: 0;
}
```

Confira como é feito esse cálculo:

320px (largura)
+ 20px `padding` (esquerda + direita)
+ 10px `border` (esquerda + direita)
+ 0px `margin` (esquerda + direita)
= 350px

A **largura** total de um elemento deve ser calculada da seguinte forma:

`element width` total = `width` + `left padding` + `right padding` + `left border` + `right border` + `left margin` + `right margin`

A **altura** total de um elemento deve ser calculada da seguinte forma:

`element width` total = `height` + `top padding` + `bottom padding` + `top border` + `bottom border` + `top margin` + `bottom margin`

Internet Explorer 8 e versões anteriores incluem preenchimento e borda na propriedade de largura. Para corrigir esse problema, adicione um `<!DOCTYPE html>` para a página HTML.

Propriedades e atalhos

Por meio das propriedades definidas é que as regras de formatação CSS são aplicadas dentro da página HTML. A combinação dessas propriedades faz com que cada elemento seja único dentro do seu documento!

Confira, na videoaula disponível no material *online*, quais são essas propriedades.

Vamos recapitular quais são as principais propriedades do CSS?

- **`Float`**

  A propriedade `Float` define onde o elemento HTML será posicionado, movendo-o para um ponto específico. Os elementos com `Float` se alinham com o ponto do elemento `parent`, e então se movem para o ponto disponível mais próximo.

- **`Clear`**

  A propriedade `Clear` se aplica para "limpar" os elementos abaixo dos elementos com `Float`. Utiliza-se `Clear` quando todos os elementos filhos estão com `Float`.

- **`Position`**

  A propriedade `Position` define a posição dos elementos que possuem um valor estático.

- **`Z-Index`**

  A propriedade `Z-Index` define o posicionamento do elemento em relação ao fundo da página e aos elementos que o sobrepõem.

- **`Display`**

  A propriedade `Display` define de qual maneira o elemento estará disposto na tela:

  - **`None`**: o elemento não aparecerá na tela.
  - **`Block`**: o elemento estará disposto em bloco.
  - **`Inline`**: o elemento estará disposto em linha.

- **`Inline-Block`**: o elemento estará disposto em blocos alinhados em linha.

- **`Overflow`**

  A propriedade `Overflow` define o tratamento da sobra do elemento:

  - **`Visible`**: valor padrão; mantém a sobra do elemento visível.
  - **`Auto`**: adiciona uma barra de rolagem quando o elemento filho sobra.
  - **`Hidden`**: esconde a sobra do elemento.
  - **`Scroll`**: sempre adiciona uma barra de rolagem, mesmo quando não for necessário.

Esses atalhos e muitos outros com a respectiva descrição você poderá conhecer com o material a seguir:

http://araguaina.ifto.edu.br/chamilo/courses/ENSINOMEDIOAPLICATIVOSWEBEWEBDESI/document/3o_Bimestre/propriedadescss.pdf

CSS e HTML

Quando um navegador lê uma informação do CSS, ele irá decodificar essa informação e formatar o documento HTML de acordo com as informações constantes no CSS.

Há três maneiras de o inserir regras de formatação em um arquivo HTML. São elas:

- Formatação *Inline*
- Formatação na *tag* `Head`
- Folha de Estilos ligada ao HTML

Formatação *Inline*

A formatação *Inline* é a formatação dentro das *tags* HTML, como atributo. O CSS *inline* é pouco utilizado em *websites* por deixar o código "poluído" com muita informação, mas é muito utilizado na criação de *mailings* e *E-mail marketing*.

Sua utilização é bem simples, para usá-lo, basta inserir o `style` dentro da própria *tag* html que ele receberá o efeito.

Como ele é aplicado diretamente naquela *tag* específica, é impossível reutilizá-lo em outra *tag*, tendo que sempre copiar e colar ou criar um novo para cada *tag* que você queira estilizar.

Formatação na tag Head

A formatação Head é a formatação dentro da *tag* `Head`, como um conteúdo não-visível ao usuário. Também conhecida como **folha de estilo interna**, essa regra de formatação pode ser usada se uma única página tem um estilo único.

```
<head>
<style>
body {
    background-color: linen;
}

h1 {
    color: maroon;
    margin-left: 40px;
}
</style>
</head>
```

Folha de Estilos ligada ao HTML

Esta é a declaração de CSS mais utilizada – a **externa**. Basicamente, ela consiste em colocarmos todo o código CSS em um arquivo externo .css e no HTML apenas "puxarmos" esse arquivo para que o código seja estilizado.

Este é o método que apresenta maior versatilidade. Um arquivo externo CSS pode ser ligado a quantas páginas desejarmos, desta forma deixando a manutenção de um site muito mais fácil (apenas um arquivo CSS pode controlar o visual de um site inteiro). Para este método, utilizamos o elemento `link`, da seguinte forma:

```
1<link href="css/arquivo.css" rel="stylesheet">
```

A *tag* `link` é uma *tag* que auto-fecha, assim como `br` e `meta`. O atributo `href` indica o endereço do arquivo CSS (hiper-referência), neste exemplo um arquivo chamado "arquivo.css" dentro de uma pasta "css". O atributo `rel` determina a relação deste *link* com a página, aqui sendo `stylesheet` ou folha de estilos. Se estivéssemos utilizando a sintaxe XHTML, também é necessário o atributo `type` com o valor `text/css`.

Atenção: a *tag* `link` não é utilizada para fazer *links* no *site*, como dizemos informalmente. Para *links*, é utilizada a *tag* `a` (que vem de *anchor*, ou âncora).

Este método é o que tem menor precedência em relação aos outros. Ou seja, estilos aplicados utilizando os outros métodos abaixo terão preferência de aplicação em caso de um conflito (estilos aplicados a um mesmo elemento).

Vejamos, agora, um exemplo dos três métodos aplicados em uma mesma página.

Importante: não deixe de ver o código fonte da página para perceber a diferença entre eles!

http://guilhermemuller.com.br/pt/elearning/exemplos/html_basico/1/inserindo-css

E agora vamos acessar o material *on-line* para assistir à videoaula deste tema!

TROCANDO IDEIAS

A W3Schools possui vários fóruns de discussão e explicação sobre conceitos e propriedades (sozinhas/combinadas) CSS. Em português, o *site* do Maujor possui boas dicas, explicações e fóruns sobre as formatações CSS.

Que outras fontes de pesquisa e de troca de experiências você conhece? Que tal compartilhar essas fontes com seus colegas acessando o fórum disponível no AVA?

NA PRÁTICA

O desafio que proporemos será: criar um arquivo HTML empregando as três formatações que acabamos de conhecer, a saber:

- Formatação *Inline*
- Formatação na *tag* `Head`
- Folha de Estilos ligada ao HTML

Está preparado?

SÍNTESE

Nossa segunda aula está chegando ao fim, mas o estudo dos conteúdos que você viu até agora não pode parar!

Pesquise mais sobre esses assuntos na *Internet*, converse com profissionais da área e revise o conteúdo estudado.

Mantenha-se atualizado!

Bons estudos!

Referências

6 estratégias para melhorar a organização do seu CSS. Disponível em: http://tableless.com.br/6-estrategias-para-melhorar-a-organizacao-do-seu-css-2/. Acesso em: 22/02/2016.

Como saber qual regra CSS tem prioridade. Disponível em: http://www.richardbarros.com.br/blog/css/como-saber-qual-regra-css-tem-prioridade. Acesso em: 22/02/2016.

PIVETTA, E. M. **Lista de propriedades CSS**. CAFW/UFSM. Disponível em: http://www.cafw.ufsm.br/~elisa/desenvweb/propriedadescss.pdf. Acesso em: 22/02/2016.

O que é CSS? Disponível em: http://www.tecmundo.com.br/programacao/2705-o-que-e-css-.htm. Acesso em: 22/02/2016.

Tenha o DOM – Entenda o que é o *Document Object Model* e tenha o DOM. Disponível em: http://tableless.com.br/tenha-o-dom/. Acesso em: 22/02/2016.

W3Shools.com*. *The World's largest Web developer site. Disponível em: http://www.w3schools.com/css/default.asp Acesso em: 22/02/2016.