BEM: Uma Arquitetura CSS do Bem

Leonardo Antonio de Deus[1]

[1]Bacharelado em Engenharia de Computação – Centro Universitário de Anápolis (UniEVANGÉLICA) – Anápolis – GO

[1]talkto@theleoad.com

Resumo. *O presente texto apresenta uma abordagem sobre uma arquitetura CSS que visa facilitar e agilizar a escrita do código, além de melhorar o entendimento em grupos, fazendo com que esse código seja escalável e tenha melhor reuso.*

Palavras-chave: *CSS; Desenvolvimento web; Webdesign; Front-end; BEM.*

1. Introdução

Ao iniciar algum projeto de aplicação web, faz-se necessário usar estilos. E hoje, mesmo com as mais modernas ferramentas e frameworks, a única maneira de adicionar estilos na web é usando CSS (e agora podemos ver isso em aplicações nativas também). O CSS tem a escrita muito simples e apresenta uma curva de aprendizado fácil, e por esses motivos apresentam várias armadilhas. Muitas vezes simplesmente pensamos na parte mais óbvia, evitando uso de tags, id ou os famosos !important. Mas não discutimos uma arquitetura CSS que ajude nosso código ser escalável, ter um bom reuso e principalmente aumentar a produtividade da equipe.

"*There are only two hard things in Computer Science: cache invalidation and naming things*" - Phil Karlton

Nomear "coisas" é uma das tarefas que fazemos o tempo todo quando escrevemos folhas de estilo. Definir quais serão nossos seletores e usá-los da melhor forma possível é uma tarefa que pode parecer simples. No entanto, sabe-se que pode virar uma grande bagunça se não houver o mínimo de planejamento e padrão na equipe que estiver desenvolvendo. Nesse contexto que está inserida a ideia de Arquitetura CSS: um conjunto de princípios, técnicas e regras para escrita do código.

2. Desenvolvimento

2.1. Contextualizando o BEM

BEM é a sigla para *Block Element Modifier* (Bloco Elemento Modificador), que nada mais é do que um padrão usado para melhorar a tarefa de nomear classes (www.getbem.com). Esse padrão se baseia na função estrutural do elemento dentro do HTML em detrimento da sua função de apresentação. O BEM foi criado pela Yandex (www.yandex.com), que é o maior motor de buscas na Rússia. Quando foi criado tinha os seguintes objetivos:

- Desenvolvimento rápido e resultados duradouros de padronização - Os projetos devem ser criados rapidamente, utilizando uma arquitetura que garante sustentabilidade e longevidade para o desenvolvimento;
- Um projeto pode envolver muitas pessoas - A capacidade de organizar de forma eficiente o trabalho das pessoas em uma equipe é importante, seja essa equipe formada por apenas duas pessoas ou por dezenas de desenvolvedores;
- Equipes escaláveis - Adicionando mais pessoas para uma equipe, o desempenho deve melhorar. O código deve ser cuidadosamente estruturado para garantir a sua sustentabilidade ao longo do tempo e através das mudanças da equipe.

- A reutilização de código - Cada elemento novo projeto ou interface não deve ser iniciado a partir do zero. Código não deve ser dependente do contexto, mas deve ser fácil de transferir para outros lugares.

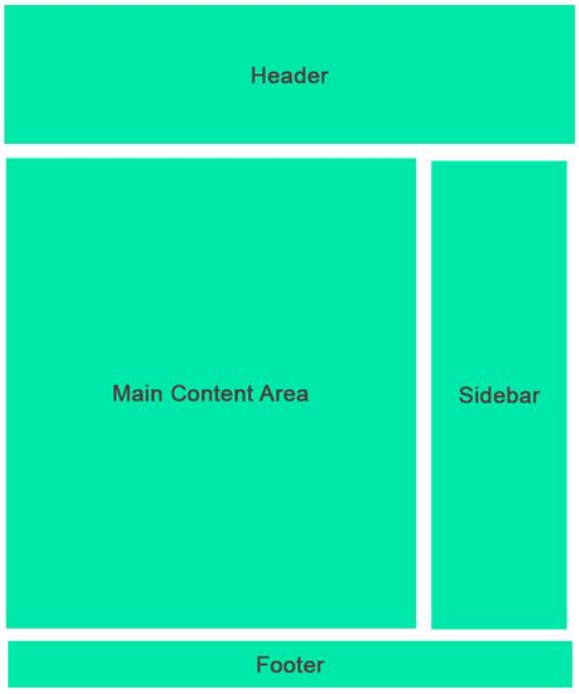
Figura 1. Exemplo de bloco de elementos.

Fonte: Website do BEM - Block Element Modifier em 30/11/2018.

2.2. BEM – Blocos

O Bloco é o container onde o elemento se encontra. Podemos pensar no bloco como sendo trechos estruturais "maiores" do código. Um bloco pode ser simples ou composto (pode conter outros blocos). Você pode ter um cabeçalho, rodapé, barra lateral e área de conteúdo principal; cada um dos quais seria considerado um bloco.

Figura 2. Exemplo de blocos numa estrutura de HTML5.



Fonte: O Autor.

No BEM nomeamos a classe de um bloco com um nome que o identifique. Portanto depois de definir seu bloco, você estará pronto para começar a nomear seus elementos.

Veja os códigos abaixo para entender como funciona o bloco no BEM:

```
HTML
<div class="block"></div>
      CSS
.block { }
```

2.3. BEM – Elementos

Um elemento é uma parte de um bloco. Definimos o bloco como sendo trechos estruturais "maiores" do código, assim podemos dizer que os elementos são as peças que formam o bloco. Dessa forma elementos não fazem sentido estarem sozinhos, eles devem estar ligados aos seus parent elements, que são os blocos.

A maneira correta de nomear Elementos na metodologia BEM é colocar o nome do bloco, logo depois o nome do elemento separando-os por dois underscores (__). Vamos visualizar os códigos abaixo para entender como funciona na prática:

```
HTML
<div class="block__element"></div>
      CSS
.block__element { }
```

No começo você pode considerar essa sintaxe um pouco estranha. Mas a verdade é que dessa maneira você consegue visualizar com rapidez e com pouco esforço do que se trata o código. Essa clareza e simplicidade do código é o que torna o BEM uma ferramenta extremamente poderosa e entrega o que promete: desenvolvimento rápido, integração da equipe, escalabilidade e reutilização do código.

Veja abaixo mais exemplos de como funciona os Elementos no BEM:

```
HTML
<form class="form" action="/">
      <input class="form__search" name="s">
      <input class="form__submit" type="submit">
</form>
      CSS
      .form__search {}
.form__submit {}
```

Uma observação relevante baseia-se em fazer é que ao usar Blocos e Elementos para nomear nossas classes, resolvemos um problema importante que evitar o uso dos seletores aninhados. Isso faz uma mágica no nosso CSS: todos os seletores em um projeto BEM têm o mesmo peso. Isso significa que é muito mais fácil redefinir os estilos escritos de acordo com o BEM. Agora, para usar o mesmo formulário em outro projeto, você pode simplesmente copiar seu layout e estilos.

2.4. *BEM – Modificadores*

Como em tudo que se faz usando código, necessitamos escrever de maneira simples e usando o que chamamos de DRY – Don't Repeat Yourself (compreendido como Não se repita). É

nesse contexto que utilizamos as classes no CSS, para evitar a repetição. Mas como alterar o estilo de um elemento específico dentro da metodologia BEM? A resposta é simples: usamos os modificadores.

Os Modificadores no BEM são escritos utilizando dois hifens (–) após o nome do elemento ou do bloco a ser modificado.

Veja abaixo como utilizar os modificadores:

```
CSS
.block--modifier { }
      .block__element--modifier { }
```

É importante ter em mente que é importante manter tudo simples e evitar criar classes extras, a menos que seja absolutamente necessário. Vamos observar um exemplo com um Header de um site com dois menus, um primário e um secundário:

```
CSS
.header__navigation { }
      .header__navigation--secondary { }
```

No menu secundário, provavelmente o layout e o espaçamento sejam os mesmos do primário, mas teremos uma cor diferente, por exemplo. A lógica é que você duplique os atributos do menu primário no secundário, mas usando um pré-processador podemos fazer isso de uma maneira mais interessante. O SASS pode nos ajudar bastante na escrita do CSS e nesse caso específico podemos fazer um @extend da classe e dessa maneira todas as propriedades são herdadas e se altera só o que for necessário.

```
CSS
.header__navigation {
      background: #008cba;
      padding: 1rem 0;
      margin: 2rem 0;
      text-transform: uppercase;
}
.header__navigation--secondary {
      @extend .header__navigation;
      background: #dfe0e0;
}
```

Figura 3. Exemplo de dois menus com mudança apenas no BG.

Fonte: UniEVANGÉLICA.

A primeira impressão que temos do BEM é que "o nome da classe é muito grande". Mas essa é maneira com que o BEM se mostra para que os nomes das classes sejam específicos, claros,

fáceis de ler dentro de HTML e comunicam claramente para que servem. O que é bastante interessante com esse padrão é que você só vai precisar usar apenas um nome de classe para cada tag html. E muitas vezes o que parece ser grande, se torna melhor do que a nomenclatura padrão que podemos estar acostumados a utilizar. Veja o exemplo:

Sem utilizar o BEM:

```
.label .label-default {}
.label .label-alert {}
```

Com o BEM

```
.label {}
.label--alert {}
```

3. Considerações Finais

A metodologia BEM foi criada para auxiliar na escrita de código mais inteligente, além de oferecer um enorme suporte para definir e projetar uma hierarquia para o desenvolvimento front-end como o conhecemos. Ferramentas que promovem desenvolvimento rápido, integração da equipe, escalabilidade e reutilização do código, são sempre bem-vindas.

O BEM é só um padrão com regras simples, mas sua equipe de código pode criar mais padrões de uma maneira que isso faça que as coisas fiquem mais simples. A Metodologia BEM é realmente encantadora e quando adotada por uma equipe de código realmente faz com que as coisas pareçam mais simples.

Referências

BELAYA, Inna. *BEM For Beginners: Why You Need BEM*. Disponível em: https://www.smashingmagazine.com/2018/06/bem-for-beginners/. Acesso em: 25 de nov. de 2018.

BEM. *BEM — Block Element Modifier*. Disponível em: http://getbem.com/. Acesso em: 25 de nov. de 2018.

EMER, Jean Carlo. *OOCSS, SMACSS, BEM, DRY CSS: afinal, como escrever CSS?*. Disponível em: https://tableless.com.br/oocss-smacss-bem-dry-css-afinal-como-escrever-css/. Acesso em: 25 de nov. de 2018.

MEDESKI, Josh. *An Introduction to the BEM Methodology*. Disponível em: https://webdesign.tutsplus.com/articles/an-introduction-to-the-bem-methodology--cms-19403. Acesso em: 25 de nov. de 2018.

SILVA, Maurício Maujor Samy. *Metodologia BEM para criar código legível*. Disponível em: https://maujor.com/tutorial/metodologia-bem-para-criar-codigo-legivel.php. Acesso em: 25 de nov. de 2018.