**REGRAS DE NOMENCLATURA**

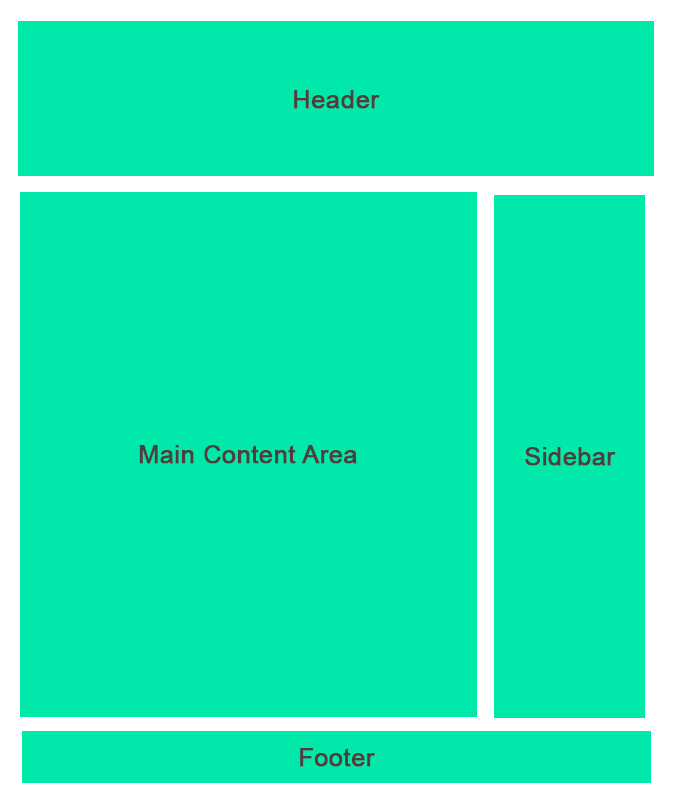
Metodologia BEM

Todos queremos tornar nossos códigos fáceis de serem lidos. Fazer isso nos ajuda a trabalhar mais rápido e eficientemente, e, quando outros trabalharem conosco, poderemos manter uma semântica coerente e clareza de código.

BEM é a sigla de Block Element Modifier que, em português, significa Bloco Elemento Modificador. Ela sugere uma forma estruturada de nomear suas classes, baseando-se nas propriedades do elemento em questão. Se você, alguma vez, viu alguma classe nomeada dessa forma: header\_\_form—email, é a BEM em ação. Ao usar a metodologia BEM, tenham em mente que usaremos, somente, classes. Nada de IDs. Classes permitem que você repita o nomenclatura BEM quando necessário, além de criar uma estrutura de codificação mais consistente (tanto nos arquivos HTML quanto nos CSS/Sass). Hora da explicação.

**Bloco**

O bloco é o recipiente ou contexto em que o elemento se encontra. Imagine isso sendo as grandes partes estruturais do seu código. Você deve ter um cabeçalho, um rodapé, uma barra lateral e uma área para conteúdo. Cada um deles pode ser considerado um bloco. Vejamos a imagem abaixo:



O elemento bloco é a raiz da classe e sempre virá primeiro. Apenas lembre que, uma vez o bloco definido, você estará pronto para nomear seus elementos.

**Elemento**

O elemento é parte do bloco. O bloco é tudo e os elementos são as partes. Cada elemento é escrito após o nome do bloco, conectado por dois traços sublinhados.

|  |  |
| --- | --- |
| 1 | .block\_\_element |

Sei que parece bem diferente, mas, uma vez que você começa a usar, você se perguntará como conseguiu codificar em CSS sem ele! Os dois traços sublinhados permite você navegar rápida e visualmente, além de manipular o código.

Eis alguns exemplos de como a metodologia funciona:

|  |  |
| --- | --- |
| 1  2  3  4 | .header\_\_logo {}  .header\_\_tagline {}  .header\_\_searchbar {}  .header\_\_navigation {} |

Como pode ver, há bastante espaço para ser criativo e moldar a metodologia ao seu gosto. “Navigation” poderia ser “nav”, “tagline” poderia ser alterada para “tag” ou “tagLine”. A chave é manter a nomenclatura simples, clara e precisa. Não pense demais e, uma vez que seu HTML e CSS estejam DRY (don’t repeat yourself - não se repita), não será problema caso precisasse alterar os nomes das classes, se encontrar algum nome mais semântico que funcione para você (apenas tente manter um padrão). Os elementos farão parte da base do nome das classes, ajudando você a entender como estruturar suas folhas de estilo e como administrar seu layout.

**Modificadores**

Agora fica interessante (caso não esteja impressionado ainda!). Quando você nomeia uma classe, o ponto é permitir que aquele elemento seja repetível, de forma que não precise escrever novas classes em outras áreas do site se os estilos dos elementos são os mesmos. Quando precisar modificar o estilo de um elemento específico, você pode usar um modificador (claro!). Para fazer isso, você adiciona um hífen duplo -- após o elemento (ou bloco). Eis um exemplo simples:

|  |  |
| --- | --- |
| 1  2 | .block--modifier {}  .block\_\_element--modifier {} |

Tenha muito cuidado com esses! Lembre-se que você quer manter o máximo de simplicidade possível para não precisar se repetir ou criar classes extras a menos que seja extremamente necessária. Falemos sobre o código usando o cabeçalho (header) de um site, como exemplo:

|  |  |
| --- | --- |
| 1  2 | .header\_\_navigation {}  .header\_\_navigation--secondary {} |

Se estiver usando uma navegação secundária, é provável que o espaçamento e o layout sejam os mesmos da navegação primária, mas a cor diferente. Você pode duplicar os estilos originais ou, melhor ainda, usar um préprocessador. Com Sass, você usaria o @extend, para estender o elemento principal (de forma que o elemento secundário herde todas as propriedades) e mudaria só os estilos necessários.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11 | .header\_\_navigation {      background: #008cba;      padding: 1rem 0;      margin: 2rem 0;      text-transform: uppercase;      }    .header\_\_navigation--secondary {      @extend .header\_\_navigation;      background: #dfe0e0;      } |

Você talvez esteja pensando: “mas os nomes das classes são tão grandes!”. Eu já vejo assim: A nomenclatura BEM é específica, clara e fácil de ler dentro do HTML e comunica, perfeitamente, seu intuito.

O que também gosto da nomenclatura BEM é que você só precisa usar uma classe para cada tag HTML. Veja como ela funcionaria para rótulos. Com seletores padrões seria assim:

|  |  |
| --- | --- |
| 1  2 | .label .label-default {}  .label .label-alert {} |

Agora, com seletores BEM:

|  |  |
| --- | --- |
| 1  2 | .label {}  .label--alert {} |

Linguagens como o Sass (o Scss, especificamente), permite-nos que criemos elementos que compartilhem os mesmos estilos com algumas diferenças. O exemplo acima previne que dupliquemos estilos, porém, permite que alteremos o que for preciso. O que gosto de verdade da metodologia BEM é que não preciso combinar nomes de classes ambíguos como “panel panel-default col-md-3”. Se você usa uma framework como a [Foundation](http://foundation.zurb.com/), lance mão das mixins. Mas, só para exemplificar, estilizemos esses rótulos que acabamos de definir.

|  |  |
| --- | --- |
| 01  02  03  04  05  06  07  08  09  10  11  12 | .label {      background: #eee;      border-radius: 505;      color: #333;      font-size: 1rem;      }    .label--alert {      @extend .label;      background: #da4531;      color: #fff;      } |

**Conclusão**

E o BEM, em resumo, é isso. Como pode ver, há muito mais coisas a se explorar. BEM é um sistema em constante evolução que permite você trazer clareza ao seu código, além de ajudá-lo a definir e configurar a hierarquia do seu desenvolvimento front-end.

Ela me ajudou bastante na construção mais eficientes de protótipos, e a transição para código de produção foi ainda mais rápido!

**Leitura Complementar**

* [Site Oficial da BEM](http://bem.info/)
* [MindBEMding – Entendendo a sintaxe do BEM](http://csswizardry.com/2013/01/mindbemding-getting-your-head-round-bem-syntax/)
* [Uma Nova Metodologia para o Front-End: BEM](http://coding.smashingmagazine.com/2012/04/16/a-new-front-end-methodology-bem/)