MÉTODO BEM PARA ORGANIZAÇÃO DO CSS:

BEM é uma metodologia CSS (ou convenção de nomenclatura) usada no mundo todo. Continue lendo para saber o que é CSS BEM, vantagens em se usar, melhores práticas com BEM, exemplos de escrita e muito mais.

BEM nasceu como parte de um framework completo de desenvolvimento do Yandex, buscador com maior market share da Rússia — apesar de desconhecido na Terra de Santa Cruz, ele é um dos mais usados buscadores do mundo, com centenas de milhões de buscas por dia!

Esse padrão de nomes ficou tão bom e tão eficiente, que só ficou atrás de petróleo e gás nas exportações da Rússia para o mundo. 😂

O que é BEM?

Pessoas do mundo todo tentam chegar a melhores práticas para se trabalhar com desenvolvimento web. Afinal, todos querem conseguir resultados melhores com esforços menores, certo?

O pessoal do Yandex, o buscador russo de maior relevância, conseguiu chegar a um padrão de nomenclatura notável, conseguindo aliar simplicidade e facilidade com muitas vantagens de implementação.

E aí vem o primeiro segredinho: BEM, na verdade, é um acrônimo para Bloco, Elemento e Modificador. Esses são os 3 pilares do padrão BEM, tendo cada um seu significado, sua função e regras próprias de escrita.

Segundo Varya Stepanova, uma das desenvolvedoras front-end que já fez parte do time BEM da própria Yandex.

BEM faz alusão a Programação Orientada a Objetos (POO), uma maneira de descrever a realidade no código, com uma variedade de padrões e uma maneira de pensar nas entidades do programa.

E, realmente, é possível perceber esta alusão a POO no uso de BEM para nomear “entidades” no CSS. Com a prática e tempo de uso esse princípio vai ficando mais evidente.

Regras de Bloco, Elemento e Modificador

Como citado, cada uma das letrinhas do acrônimo BEM tem sua função específica dentro da metodologia, ora sendo usada em separado, ora em coordenação com as demais.

E são justamente essas regras de escrita e respectivas funções de Bloco, Elemento e Modificador que serão explicadas agora.

Bloco

Em BEM, um Bloco é uma entidade independente, um “bloco de construção” de uma aplicação; uma abstração mais geral, com seu próprio significado.

Se você está acostumado com outras convenções/metodologias (como SMACSS), para facilitar, pode considerar um Bloco como um Componente ou Módulo.

Bloco é a parte da trinca com mais flexibilidade na nomenclatura, significando que você pode atribuir o nome à "entidade" da maneira que quiser — inclusive em conjunto com outras convenções, como Namespaces CSS.

Elemento

Um Elemento faz parte de um Bloco; é um elemento-filho. Um Elemento não tem função/significado independente e está semanticamente vinculado ao seu respectivo Bloco.

Elementos têm uma regra específica de escrita, sendo uma classe CSS composta de Nome do Bloco + 2 underlines + nome do Elemento: .[Bloco]__[Elemento].

Por exemplo, se você tem um Bloco .c-menu, um Elemento dele pode ser .c-menu__item ou .c-menu__link.

Dentro das regras de escrita de BEM, esses 2 underlines representam um Elemento. Perceba que isso não viola as regras de nomenclatura de CSS, ao mesmo tempo em que deixa claríssimo do quê se trata, pois é de conhecimento geral (entre devs front end) que __ é um Elemento de BEM.

Modificador

Como sugere o próprio nome, um Modificador de BEM serve para… Modificar. Mas modificar o quê? Pode ser um Bloco ou um Elemento.

Isso mesmo: um Modificador pode ser aplicado tanto em um Bloco, quanto em um Elemento, servindo como uma variante usada para alterar sua aparência, variar uma propriedade ou atribuir um estado.

Sua regra de escrita é uma classe composta pelo nome de um Bloco ou Modificador + 2 hífens + o nome do Modificador: [Bloco|Elemento]--[Modificador].

Seguindo o exemplo, poderia haver um modificador para o menu .c-menu--dark para um versão escura; uma versão do menu fixa com .c-menu--fixed ou; para links desabilitados, .c-menu__link--disabled.

Melhores práticas para escrever BEM

Se estiver conhecendo BEM agora, ou mesmo se já usa há algum tempo, é interessante conhecer melhores práticas recomendadas para BEM para garantir que está escrevendo código de qualidade.

Não replique a hierarquia HTML no nome das classes BEM

Percebeu, no exemplo acima, que o link não recebeu a classe .c-menu__item__link? A hierarquia do HTML era exatamente essa (ul > li > a), mas a classe do Elemento omitiu "__item" do nome.

Isso porque não é preciso replicar exatamente a estrutura do HTML para compor os nomes das classes em BEM. Se fosse assim, imagine o tamanho dos nomes de classes de Blocos mais complexos…

Inclusive, uma das principais críticas negativas a BEM é que adotar a metodologia faz do código muito verboso, com nomes muito compridos.

Em partes, isso pode ser verdade, mas muitos dos críticos desconhecem essa boa prática de escrita, deixando de usar nomes bem curtinhos, como os do exemplo.

Lembre-se que pode usar nomes compostos

Mas existem casos em que, sim, cairia bem escrever algo com diversos níveis de hierarquia (mas não exatamente replicando a coisa toda).

Imagine um caso como .c-card__wrapper__info__image.

Apesar da extensão, o nome da classe está altamente descritivo e com bom nível de Especificidade CSS. Parece que a única desvantagem é, realmente, o tamanho que tem este nome de classe.

Mas nomes tão grandes realmente podem cansar a leitura (além de aumentar o peso final dos arquivos HTML). E imagine este elemento com 2 ou 3 Modificadores, cujos nomes devem conter o nome do Bloco…

Para casos como este, você só tem que se preocupar com o seguinte: prover um nome único, que deixe o elemento inequivocamente identificável dentre os demais.

Uma sugestão possível seria o nome composto .c-card__info-image (supondo que o componente terá mais de uma imagem).

Se for o caso de haver somente 1 imagem em todo o componente, até mesmo .c-card__image daria conta do recado, mesmo, no HTML, o elemento constar em um nível de hierarquia mais profundo.

Blocos dentro de Blocos

É bastante comum no dia-a-dia do desenvolvimento front end ser necessário haver Blocos dentro de Blocos — ou Componentes dentro de Componentes, se estiver mais acostumado com esta nomenclatura.

Então, pode acontecer de ser necessário modificar ligeiramente a aparência de um Bloco interno quando ele está contido em um externo.

Qual a melhor abordagem? Criar um modificador para o botão?

Até poderia ser o caso de criar um Modificador .c-button--large, mas, em casos assim, você poderia se perguntar: esse modificador para deixar o botão maior poderá ser usado em outros lugares do site ou somente dentro do Card?

Se sim, então, realmente, vale a pena que se crie o Mod; se não, é o caso de um estilo que afetará o Botão somente quando ele estiver dentro do Card.

Conclusão

Então esta é a famosa metodologia BEM, acrônimo para Bloco, Elemento e Modificador, trinca famosa por sua facilidade e simplicidade no uso que traz inúmeras vantagens aos projetos front-end que a adotam.

Alguns desenvolvedores têm uma reclamação: “Além de verboso, BEM é feio!”. Parece que ter vários __ e -- por todo o HTML e CSS não agrada a todos.

Se, para você (ou seu time), deixar de usar essa sintaxe compensar mais que:

- Código desacoplado;
- Reúso automático de código;
- Menos repetições;
- Rápida identificação de estruturas HTML através do CSS (e vice-versa);
- Independência de classes;
- Seletores menores e mais performáticos;
- CSS mais manutenível.

Então, realmente, você não deveria usar BEM…

A decisão fica por sua conta. Esperamos que escolha BEM! ;)