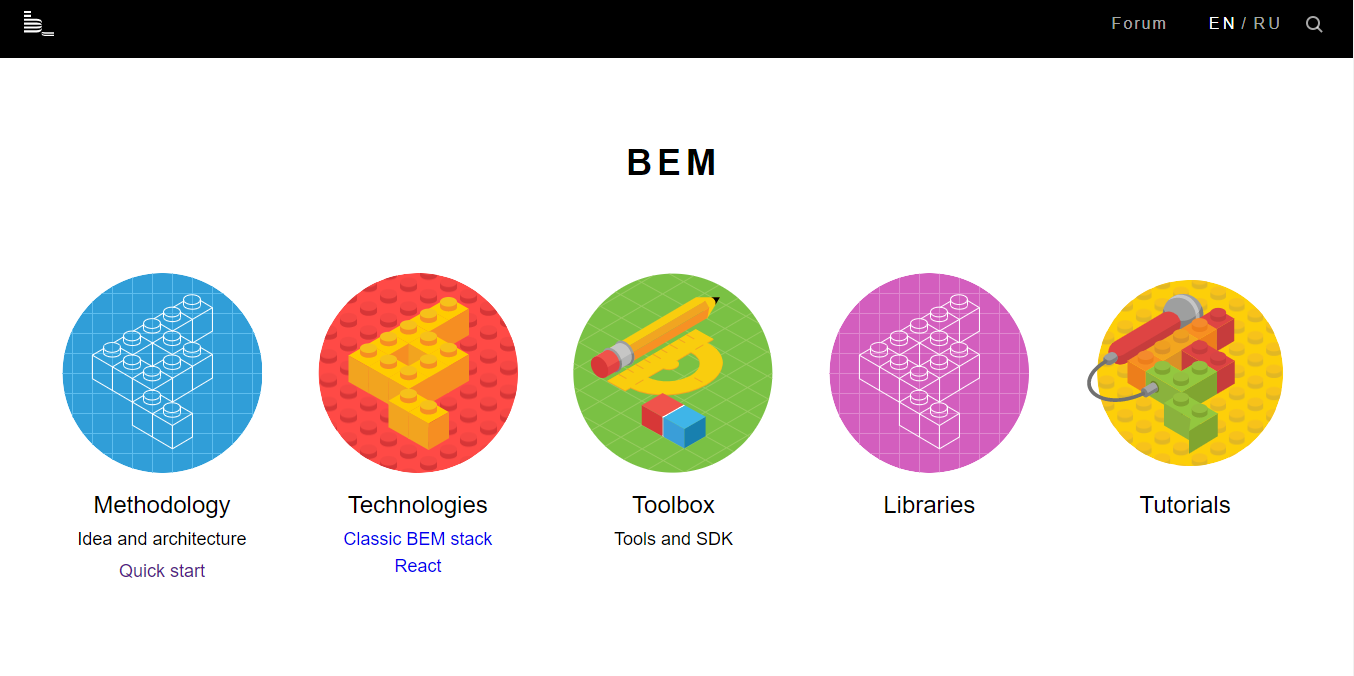
# BEM

## Úvod

Už jste viděly v překlepu, že jsem použila název třídy **picture\_\_description**

* Důvod – příprava na tuto lekci – metodika BEM, hned se dozvíte proč
* Prezentace od Luďka – podrobná – my to prolítneme (ukázky a praxe)
* <https://en.bem.info/> - uvodní stranka – další pošlu do odkazů

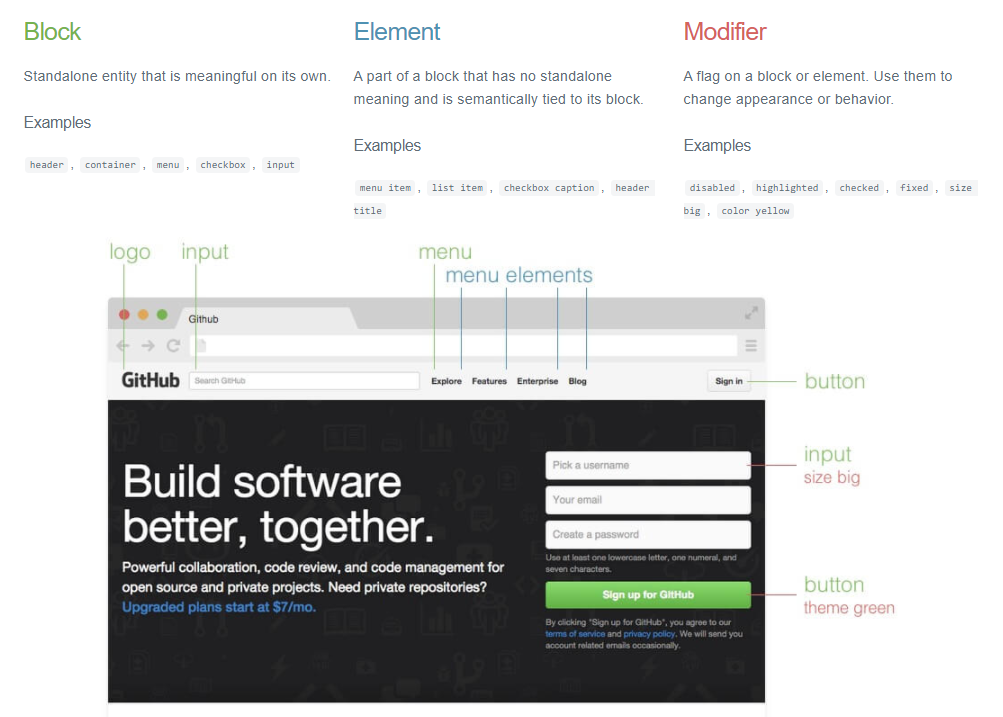


## Co je to BEM?

* BEM je metodika pro pojmenování tříd a organizace HTML/CSS.
* Vzniknul v týmu v Yandexu (ruský google)
* Použití jako metodika, nebo konvence s vlastními úpravami a přizpůsobením pro daný projekt
* Hodí se na větší projekty se složitější strukturou (klidně i na vaše projekty)

-samozřejmě lze stylovat sémantické HTML, ale

* Velmi populární, protože dovoluje vytvářet znovupoužitelné komponenty
* Vysvětlíme si, z čeho se BEM skládá a jak si organizovat stránku nejdřív vizuálně
* BEM je zkratka Block Element Modifier <http://getbem.com/introduction/> - *scroll down*



## Block

* samostatná logická a funkční jednotka. Má sama o sobě význam.
* Začneme o těchto pomalu mluvit jako o komponentách
* Ukázat – logo, input 2x!! search a na straně

## Element

* jednotky, ze kterých je blok tvořen. Elementy samy o somě nemají význam, jsou spojené s blokem.
* V html to jsou potomci – ne jen přímí potomci!!!
* Elementy v bloku se NEZANOŘUJÍ!! Vždy se jedná pouze o block a element

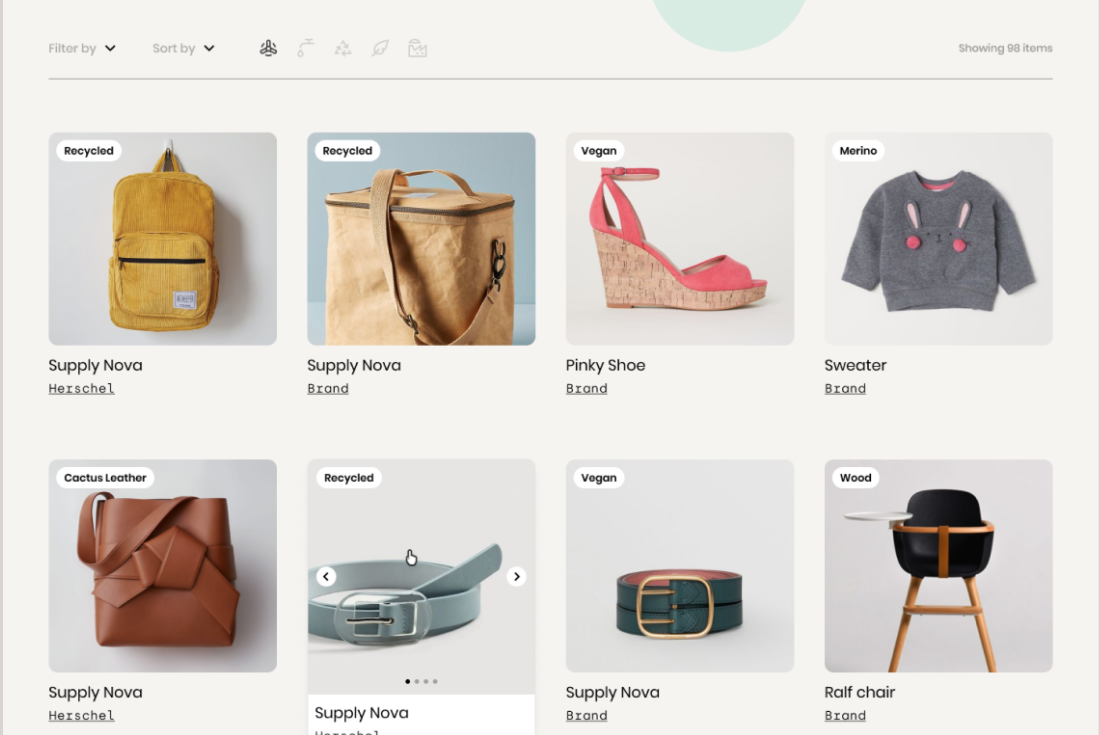
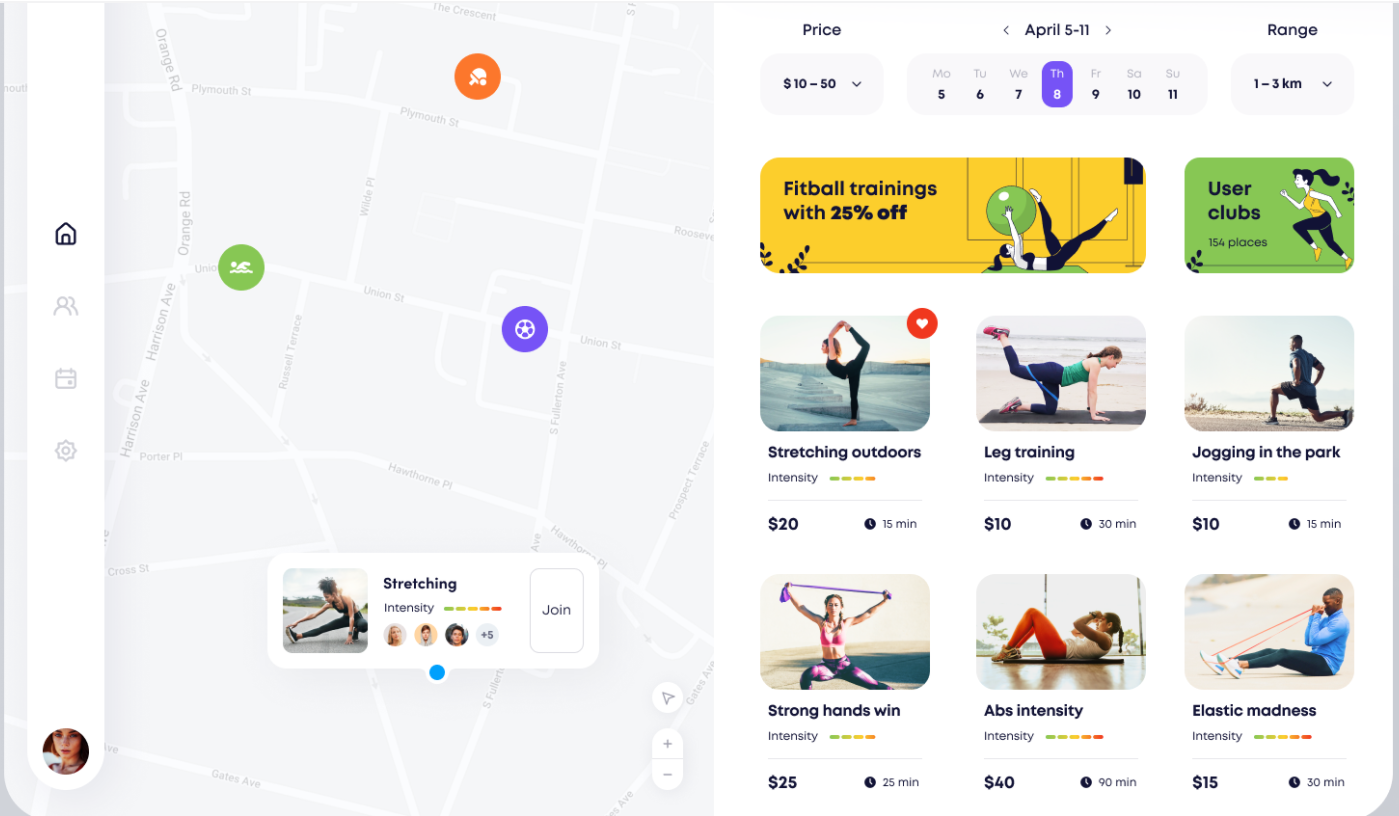
## Modifier

* přívlastek bloku nebo elementu, který mění nějakou jejich vlastnost nebo chování
* Modifier mění UŽ existující element nebo block. Proto prvek, který je modifikovaný musí mít třídu svojí PLUS třídu modifikátoru

TVOŘÍ REUSABLE COMPONENTS! ZNOVU POUŽITELNÉ – vizuál a potom zápis tříd

<https://dribbble.com/shots/14296759-Training-Finder-Dashboard/attachments/5952199?mode=media>   
<https://dribbble.com/shots/14135972-Search/attachments/5760102?mode=media>

ukázka B E M – elementy se nezanořují



Zápis samotných tříd

<https://codeburst.io/understanding-css-bem-naming-convention-a8cca116d252> - otevřít v anonymním okně!!!!

// Blocks are named as standard CSS classes  
**.block {}**

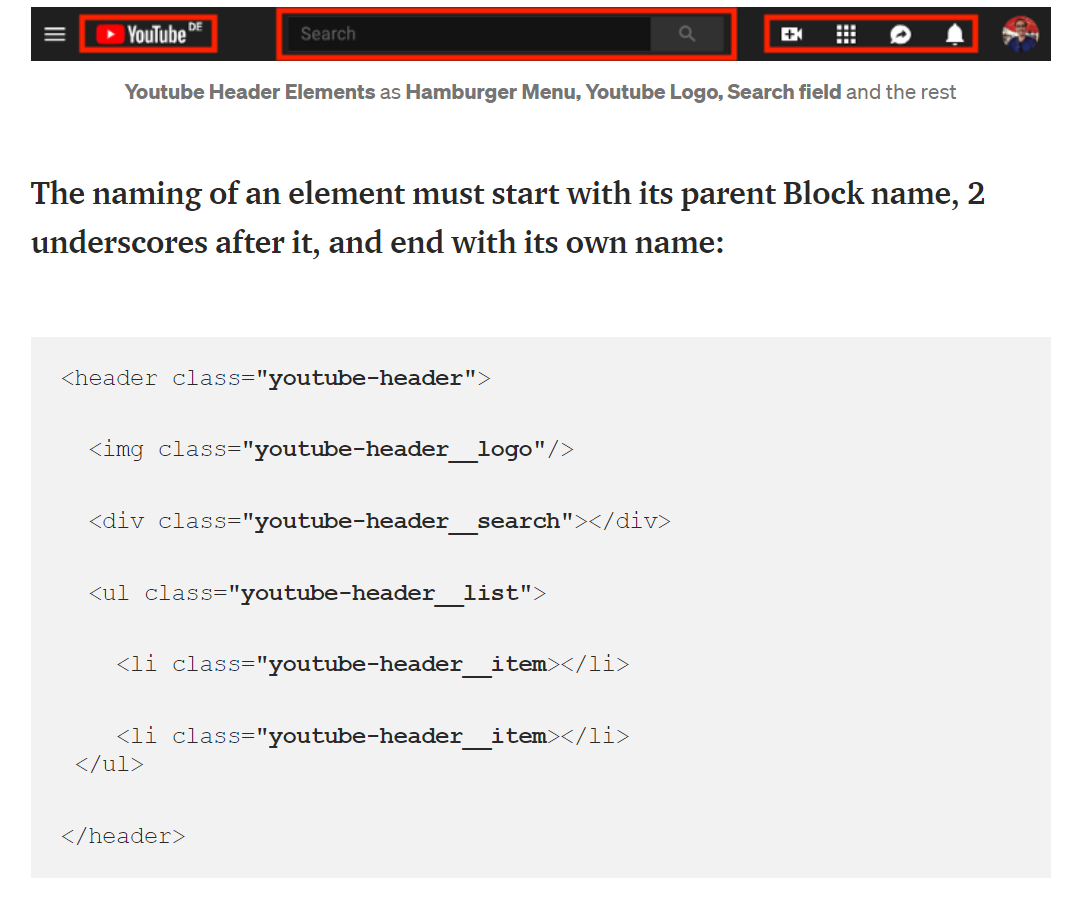
// Elements declared with 2 underscores, after block  
**.block\_\_element {}**

// Modifiers declared with 2 dashes, after block or after element  
**.block--modifier {}**

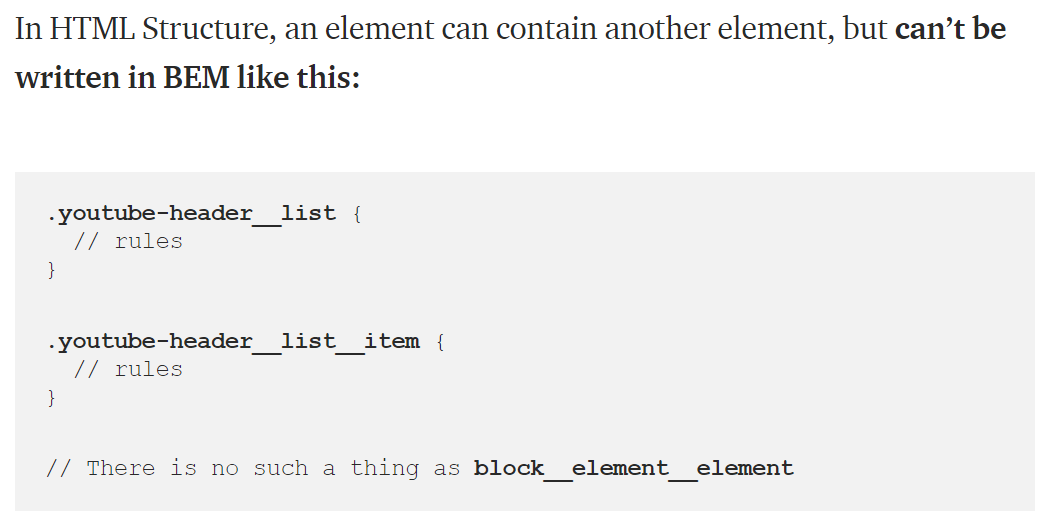
//element and modifier together  
**.block\_\_element--modifier {}**

Důvod proč jsou \_ \_ nebo - - je protože pokud je třída víceslovný název, oddělují se pomlčkou

Ukázka na stejné stránce

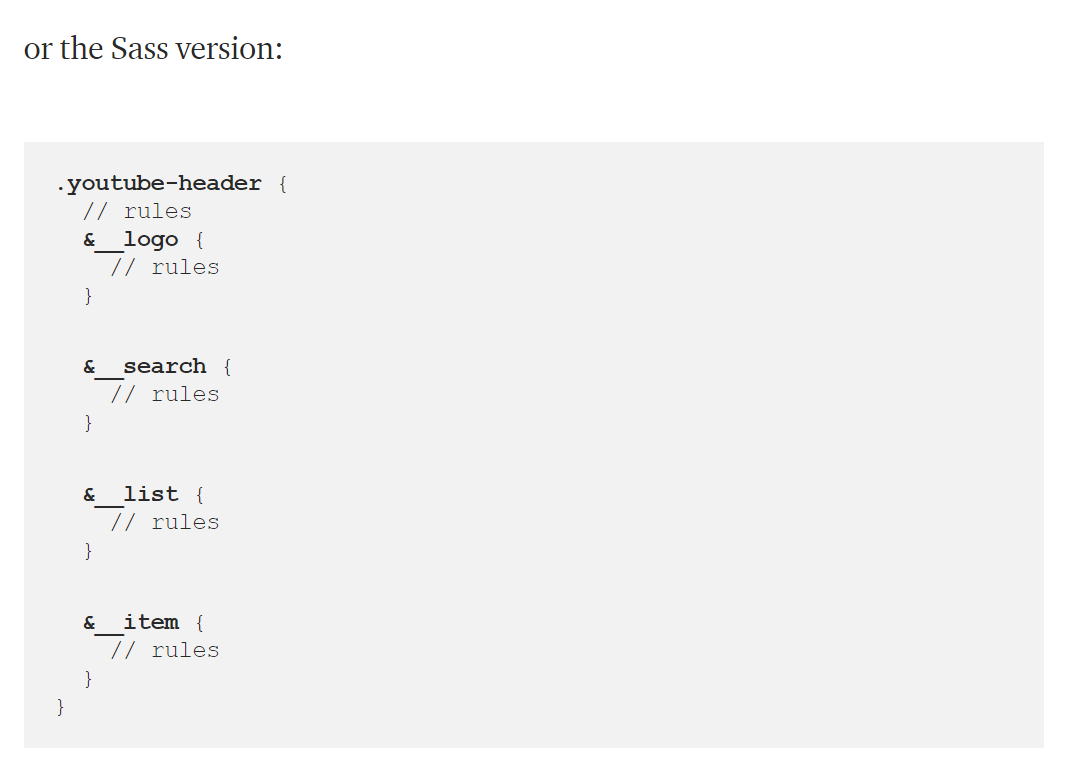


Opět NEZANOŘUJEME!!!!



Proč?

* Můžeme recyklovat CSS na jednotlivých komponentách
* Rychle se zorientujeme jak na sobě prvky závisí
* Lepší komunikace v týmu
* Bez BEMU může CSS nabobtnat a stát se velice křehkým a náchylným na rozbití
* Snadno se upravuje, adaptuje na nové úpravy
* Jednotlivé vlastnosti se nepřepisují z různých míst a tříd – píšeme „ploché css“
* Přehledný zápis za pomocí Sass or Less preprocesorů



Ukázky na stránkách:   
<https://www.linkedin.com/feed/>

<https://kodim.cz/>

ukázka v kódu

* protože už nám chybí kavárny
* kouzlo bude, až se to dynamicky všechno vypíše

cvičení

* protože už nám chybí cestování
* Je potřeba si to osahat a prozkoušet, není to vždycky jasné

Závěr : je dobré si na to zvykat, ale není potřeba nasílu malé úkoly psát do BEMu, ať nad tím nestrávíte zbytečně moc času   
doporučuji dívat se na stránky a zkusit si je v hlavě rozložit na komponenty, případně se podívat, jestli to jde najít v devtools