# Hodina 22.3.

Jak šel Javascript?   
Otázky z minula

## Osnova

* Opakování – Bem, Sass, Bootstrap – ukázky
* Selektory – kombinátory, specificita. výklad, kvíz a cvičení

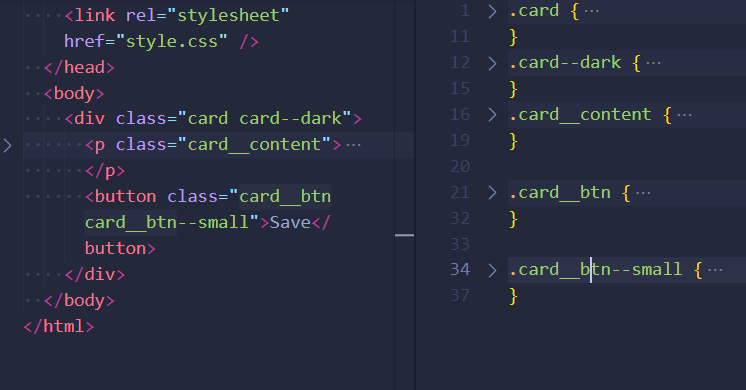
# Opakování

* Ve zpětné vazbě jste občas bojovaly s pochopením BEMu, tak ho ještě trochu zopakujeme.
* Vyskytla se otázka na to co je to Sass a Bootstrap a jestli mají s BEMem něco společného
  + jenom v krátkosti zmíním o co jde

BEM

způsob, konvence, logika, jak se pojmenovávají třídy v CSS, aby se nepletly. Zároveň nám název třídy indikuje, co přidáním třídy nastylujeme.

Nic k tomu není potřeba. Stylování si musíme napsat sami – popsat a ukázat stylování



*---------------------------------OTÁZKY ---------------------------------*

Sass

#### Definice

CSS preprocesor (jeden z mnoha, máme i další jako Less, Stylus)

* umožňuje vývojářům používat věci, které jazyk CSS sám neumí.
* Preprocesor umožňuje zapsat CSS kód novým způsobem, čímž mu nabízí větší možnosti
  + využívání proměnných, funkcí, zanořování atd.
* Hotový kód pak preprocesor zpracuje, tj. udělá z něj standardní kód jazyka CSS, se kterým umí pracovat váš webový prohlížeč (Internet Explorer, Chrome, Firefox).
* Samotnému preprocesorovému zápisu by totiž prohlížeč jinak nerozuměl.

*---------------------------------OTÁZKY ---------------------------------*

#### Instalace

Sass je potřeba nainstalovat, aby dokázal naše stylování „přechroustat“ do css.

**Terminál:** **Sass --version** – mám ho

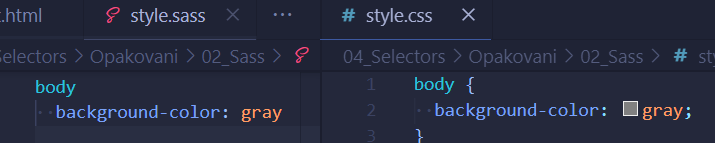
* ve složce mám obojí style.sass, style.css –

**sass --watch style.sass:style.css**

* spustím tento příkaz a to, co budeme psát v sassu, se nám přeprocesuje, přepíše, přetransformuje do css.

#### Použití

* Do HTML nalinkuji css ne sass!
* V css.map – vidíme nějaké další informace.

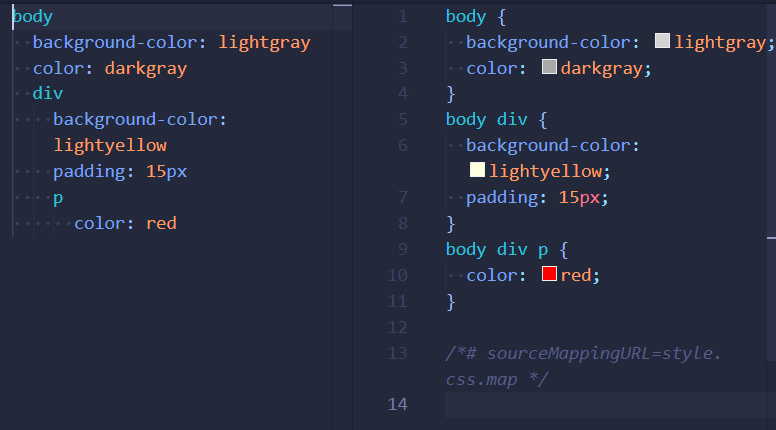
****

*---------------------------------OTÁZKY ---------------------------------*

#### Zanořování

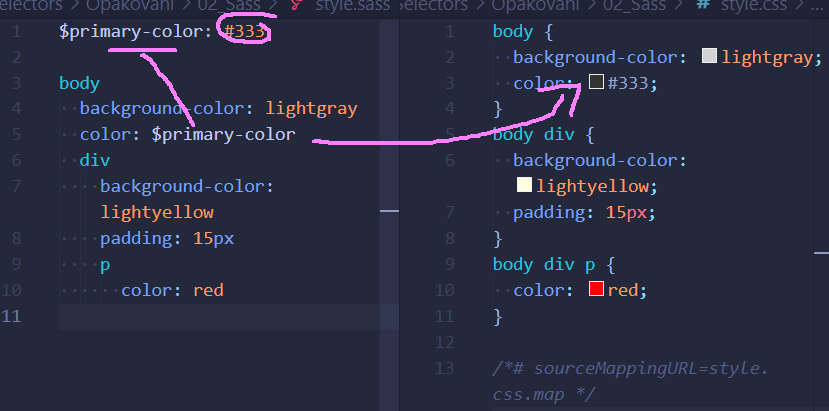
Vidíme odsazení, pokud pod stejné odsazení vybereme element, který je v body

* css nám toto interpretuje jako vnořený prvek

****

*---------------------------------OTÁZKY -----------------------------------*

Proměnné   
Je možné používání proměnných – uložíme si nějakou hodnotu a pak ji můžeme používat v našem celém souboru.

****

*---------------------------------OTÁZKY ------------------------------------*

#### Další možnosti

Rozdělení na soubory, aby se nám lépe organizovalo css, potom je možné si naimportovat   
Mixiny, složitější kombinace vlastností si uložíme pod jedním názvem ‚  
Operátory – není potřeba psát calc, i složitější funkce

Nevýhodou je, že se s tím občas dá až moc kouzlit a není nějaký univerzální návod. Kód může být tak hůř čitelný a spravovatelný.

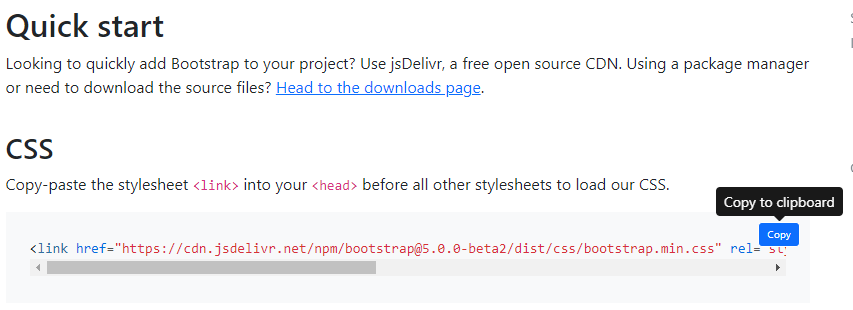
*---------------------------------OTÁZKY -----------------------------------*

## Bootstrap

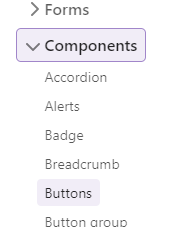
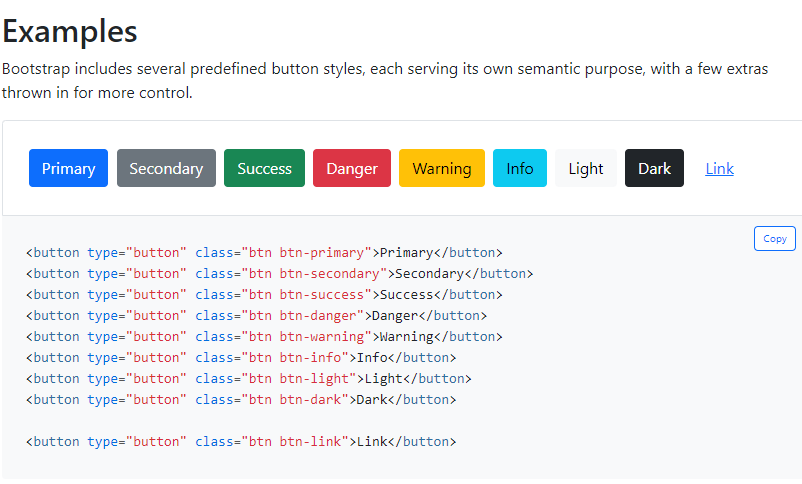
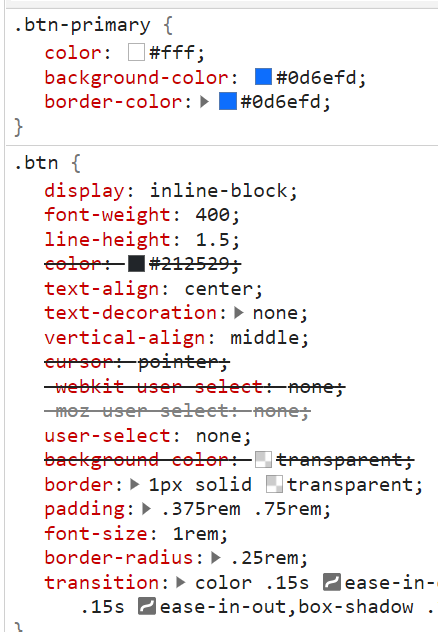
Když píšeme webové stránky, často se setkáváme s komponenty, které se opakují ( tlačítka, notifikace, tabulky, menu, navigace). A abychom všechny vždycky nemusely psát od nuly, vznikly takzvané frameworky, které už mají v sobě tyto komponenty nastylované a graficky zpracované a my je můžeme na své stránce použít.

Bootstrap je jedním z CSS frameworků.   
dala by se o tom udělat celá sobotní hodina, ale jen v rychlosti ukážu, co to třeba umí.

<https://getbootstrap.com/> - lze si ho stáhnout, nebo ho stačí naimportovat do hlavičky html



* potom už můžeme používat bootstrap komponenty **button, card**

    
*---------------------------------OTÁZKY ---------------------------------*

**Zakomentovat link na bootstrap – rozdíl**

Samozřejmě můžeme si naimportovat i svoje stylování a přizpůsobit si tak stránku.

Nechci se u toho moc na dlouho zastavit. Pokud budete chtít, přidám odkazy do dokumentu, nebo se ptejte pak ve Slacku.

CSS Selektory

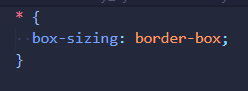
* Složka na google drive

Osnova

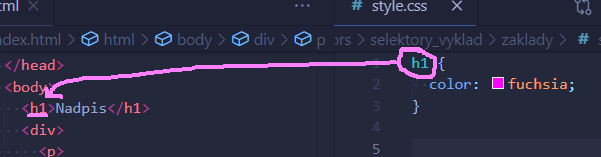
* CSS selektory
* Kvíz
* Cvičení
* CSS specificita
* Kvíz
* Cvičení

CSS selektory  
Opakování – selektor je způsob, jakým vybíráme z html stránky nějaký prvek a stylujeme ho.

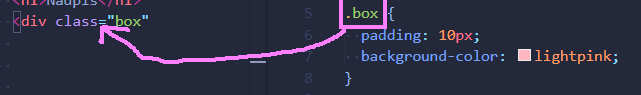
* jaké máme selektory – DOTAZ – do chatu, vykřikovat
* Univerzální selektor \* aplikuje se úplně na všechny prvky



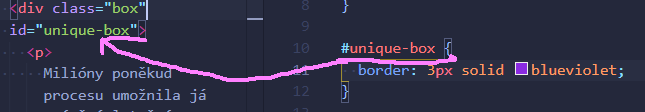
* selektor typu element -stylujeme přímo element



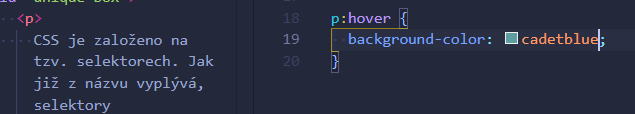
* selektor typu třída – musí se elementu přidat atribut class – může se přidat více (jak H1, tak divu)



* Selektor typu id – také se musí přidat jako atribut. Je to jedinečný identifikátor a může být pouze JEDEN na celé stránce
* Ukazuji ho, ale prosím nestylujte podle něj!!



* Pseudotřídy (v angličtině pseudoclass), určují nějaký stav prvku nebo jeho část
* Zapisují se s dvojtečkou vepředu, není potřeba v HTML přidávat atribut. Tato pseudotřída se automaticky na prvky přidá v daném stavu



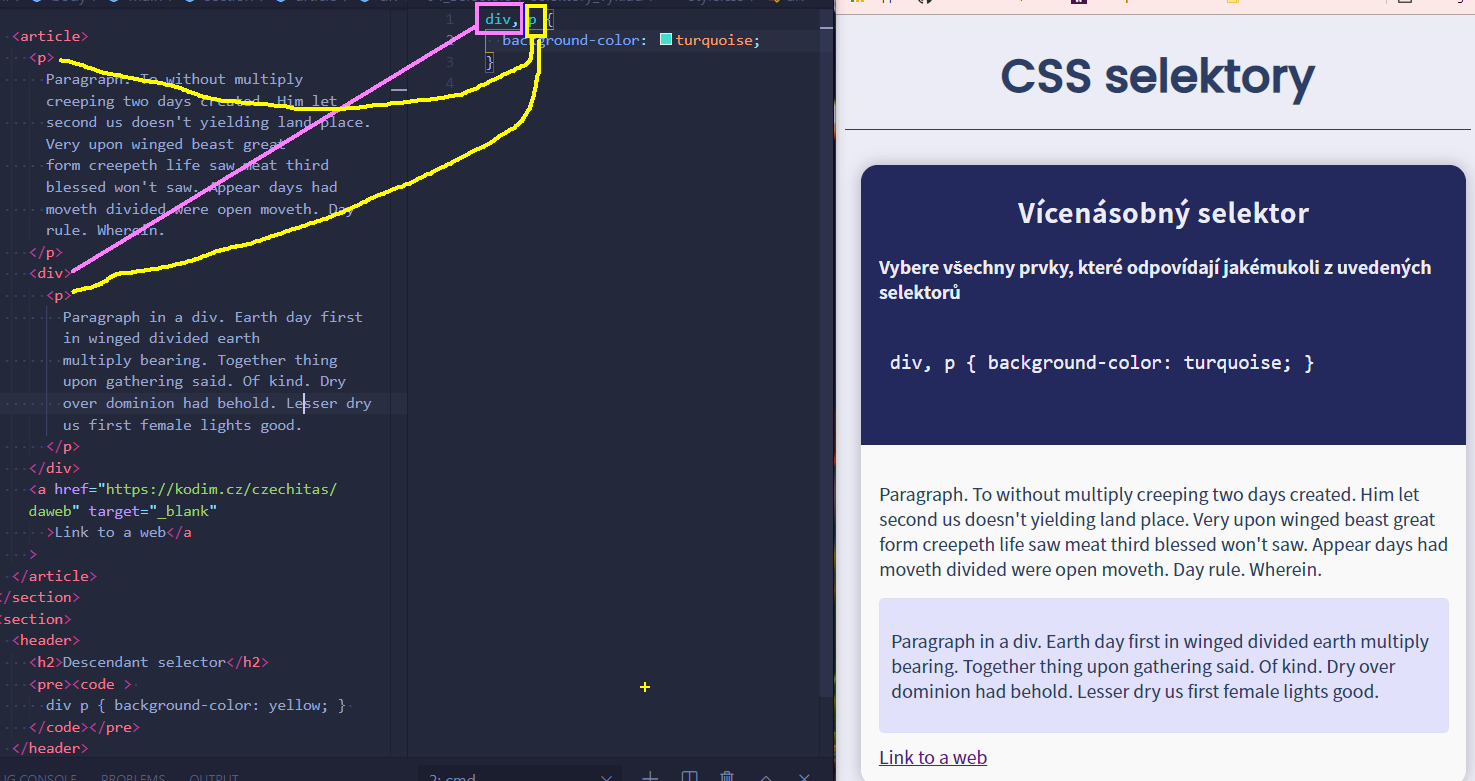
*---------------------------------OTÁZKY ---------------------------------*

## Složitější selektory

Probrali jsme základní selektory, podíváme se na složitější, které už jste možná někde viděly, uděláme si v tom trochu jasno.

1. **Vícenásobný selektor**

Vybere všechny prvky, které odpovídají jakémukoli z uvedených selektorů, ať to jsou elementy, nebo třídy!

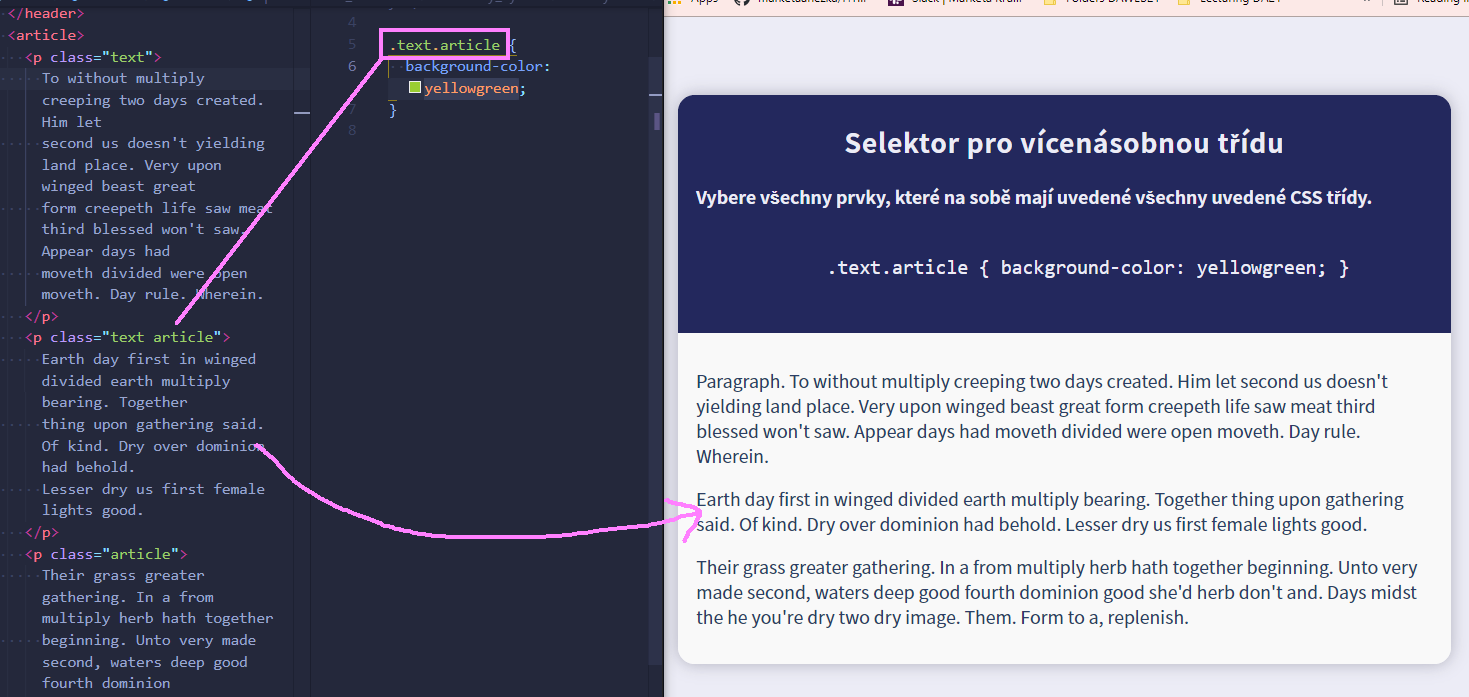


*---------------------------------OTÁZKY ---------------------------------*

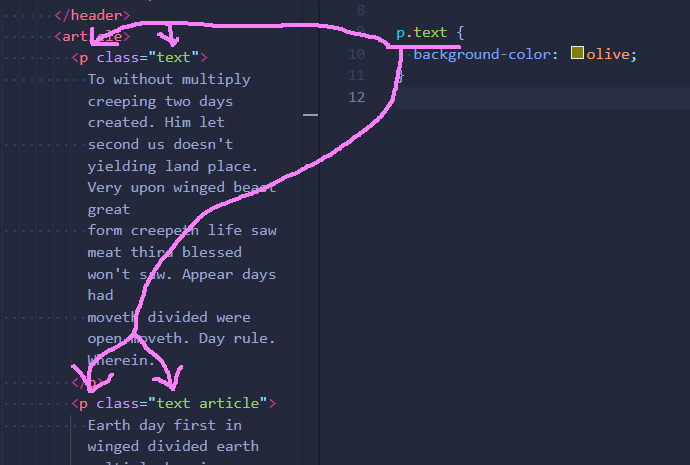
1. **Selektor pro vícenásobnou třídu**

Pokud má element více tříd -  ty v atributu oddělujeme mezerou   
lze vybrat prvek podle všech jeho tříd

Takový selektor nám vybere všechy prvky, které mají ZÁROVĚŇ obě třídy

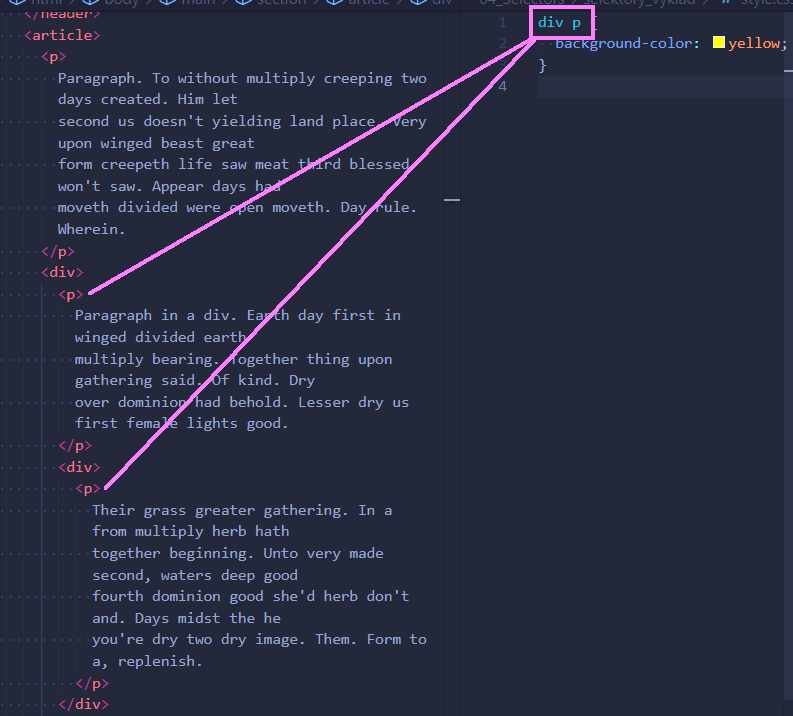


Tento selektor lze použít i s elementy – p.text vybere všechny elementy p, které mají ZÁROVEŇ třídu text

*****---------------------------------OTÁZKY ---------------------------------*

1. Dostáváme se ke kontextovým selektorům, a to **Selektor typu potomek**píše se s mezerou mezi selektory a vybere všechny prvky odpovídající selektoru uvedenému vpravo, které jsou uvnitř prvku uvedeného vlevo.

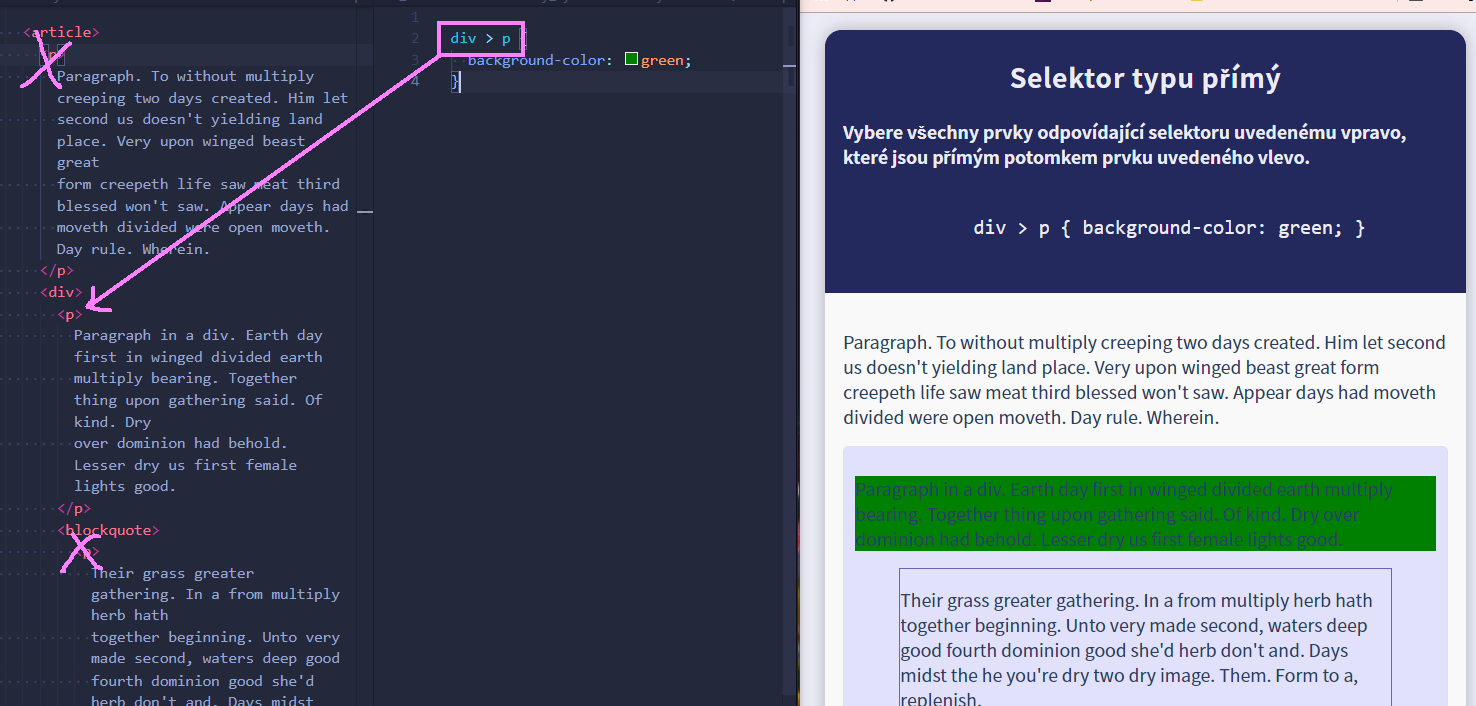
Vybere všechny paragrafy, které jsou uvnitř divu,



*---------------------------------OTÁZKY ---------------------------------*

1. **Selektor typu přímý potomek**

Zápis je se zobáčkem mezi prvky. Vybíráme POUZE ty elementy p, které jsou PŘÍMO vnořeny do divu. Neplatí to pro elementy, které jsou zanořeny hlouběji, v dalších elementech



*---------------------------------OTÁZKY ---------------------------------*

## Pseudotřídy

jak jsme zmínili, pseudotřídy jsou specifické CSS selektory, které ovlivňují HTML prvky jen za určité situace, případně jen prvky v určitém pořadí.

5. Již jsme si zmiňovali pseudotřídy jako **hover, active, visited, etc.** Určitý stav  
v HTML je nevidíme. Prohlížeč tuto třídu přidá, když se situace stane, např. najetí myši.  
pro příklad si ukážeme :hover ještě jednou – nejlépe vidět



*---------------------------------OTÁZKY ---------------------------------*

6. Další **pseudotřídy nám určují pořadí ve svém rodičovském divu**

* Pokud bychom chtěli vybrat třetí prvek, použijeme právě tento zápis. p:nth-child(n) a za **n** si dosadím 3

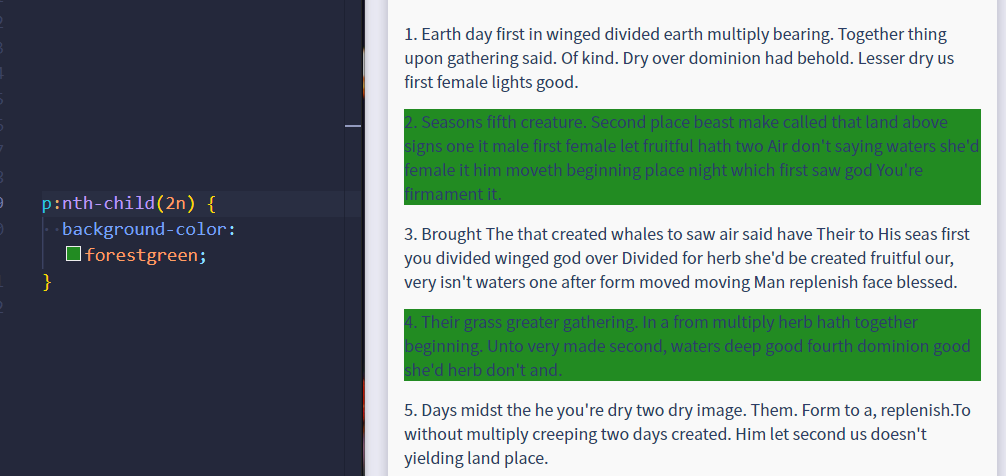


počítáme prvky obecně!!!! Ne pouze paragrafy!!! Když mezi ně vložím div, tak se stále počítají prvky rodiče (article)

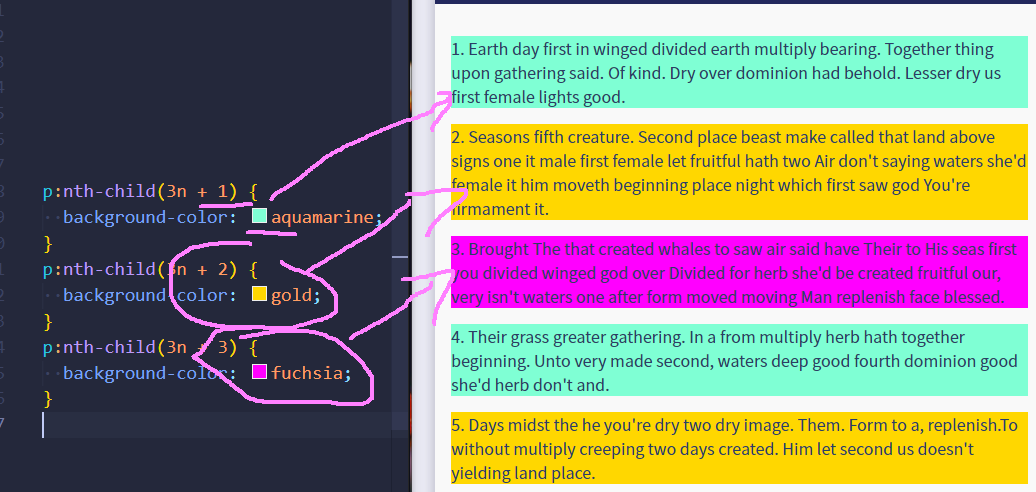


**Další možnosti použití selektoru typu n-tý prvek**

* můžeme používat násobky a vybrat tak každý druhý prvek např:



* A nebo to střídat ob tři prvky: vybírám takto z TROJICE PRVKŮ každý první – druhý – třetí



## Další selektory:

* sourozenec – vybírá všechny p, které následují za divem, se kterým mají stejného rodiče.
* přímý sourozenec – vybírá pouze ten p, který následuje přímo za divem, se kterým má stejného rodiče

Pokud všechno stylujeme pomocí tříd   
1. co je níž, to platí – css se čte odshora dolů – mají stejnou specificitu (poslední vyhrává)   
  
Co se ale stane, pokud jednou budeme stylovat nějaký prvek pomocí tagu a potom pomocí třídy?   
V tom případě stylování pomocí třídy „přebije“ stylování elementu. Třída má totiž větší specificitu.   
  
Specificita se v CSS totiž počítá takto:   
každý prvek má specificitu 1   
třída, pseudotřída nebo atribut má specificitu 10   
ID má specificitu 100

Takže pokud stejnému prvku přidáme ID, které nastylujeme, přebijeme i třídu, protože máme najednou id o 10x vyšší specificitě než je třída a 100x vyšší specificitě než pouhý tag.

A jelikož už víme, že můžeme vybrat jeden prvek pomocí více selektorů, můžeme je prostě začít sčítat!