Flexbox 1

* Relativně nová věc v CSS, dlouho tam chyběla, až ve verzi 3 se tam objevila
* Něco, co nám umožňuje dávat věci vedle sebe
* Nástroj (layout module), který přináší schopnost rozmisťovat prvky na stránce
* Např se float používal i na jiné prvky než na obtékání obrázků textem
* Css grip je ještě modernější, používá se, ale flexbox ještě nenahradil
* Odpověď na 80 % problémů je použít flexbox (na problémy s layoutem)
* Flexbox.io (videokurz od kanadského lektora)
* Css.tricks (a complete guide to flexbox)
* Interaktivní hra na procvičení flexboxu – flexbox froggy
* Nástroj pro tvorbu layoutu – rozložení prvků na stránce
* Dává věci vedle sebe (respektive podél jedné osy – buď do řádku nebo pod sebe do sloupce
* Dovnitř flexboxu (rodičovského prvku) uzavřeme potomky, když zapneme flexbox na rodiči, on dá svoje vnořené prvky vedle sebe
* Umožňuje nastavit vnořeným prvkům pravidla, prvky se budou nějak chovat – natahovat, smršťovat apod.
* Vždy máme dva typy prvků – jeden rodičovský (flex container) – například div, na tomto prvku zapínáme flexbox; všechny prvky uvnitř se stávají položkami flexboxu (flex items)
* Zapneme jej vlastností **display: flex**
* Náš úkol – když zapneme na kontejneru flex, tak flexbox dá prvky vedle sebe do řádku; zároveň se prvky smrsknou na nejmenší možnou šířku každého z nich
* Často se říká, že je jednorozměrný – rozmisťuje prvky v jednom směru
* Flexbox má dvě osy – hlavní osu, která standardně vede zleva doprava a má vedlejší osu, která je na hlavní osu kolmá a ve výchozím nastavení flexboxu vede seshora dolů
* FLEXBOX TEDY NEDÁVÁ VĚCI VEDLE SEBE, ALE VEDLE SEBE VE SMĚRU HLAVNÍ OSY; směr os lze otočit
* Můžeme i nastavit, aby hlavní osa vedla shora dolů, pak by flexbox dával položky vedle sebe ve směru šipky
* Prvky se rozmisťují podél hlavní osy a zarovnávají se podle vedlejší osy
* Ve směru hlavní osy můžeme prvky ovlivňovat určitými vlastnostmi
* Vlastnosti flexboxu se dělá na ty, které nastavujeme kontejneru X položkám flexboxu
* Rodiči říkáme: buď flex, nastavujeme display-flex
* Jiná sada vlastností se nastavuje na položky uvnitř

Velikost položek ve Flexboxu

* Umožňuje položkám uvnitř, aby se mohly natahovat, pokud zbylo volné místo nebo kdyby byly moc velké, tak aby se mohly smrskávat
* Řídí se třemi vlastnostmi:
  + **Flex-grow** (řídí, jak moc nebo jakým způsobem se může položka natahovat)
    - Nastavuje se jako číslo bez jednotky a znamená to poměr, jakým se bude položka zvětšovat
    - Standardně vypnuté
    - **Standardně je tedy flex-grow: 0; standardně se nenatahují, musíme jim to dovolit**
    - Když nastavíme jedničku, protáhla se nám položka přes celou šíři boxu
    - Když je tam volné místo, tak se provede součet flex-grow a volné místo se rozdělí na tolik částí, kolik vyšel součet a každé položce se přidělí tolik částí volného prostoru, jakou má položka flex-grow
    - K šířce položky se přidá dílek volného prostoru
    - **Nezajišťuje stejnou velikost položek, ale jejich natažení o stejný kus, respektive rozdělení díků rovnoměrně mezi položky**
  + Flex-shrink (jakým způsobem a jak moc se bude položka smrskávat)
    - Standardní nastavení je 1 – tzn. standardně se mohou smrskávat
    - Při nastavení hodnoty 0 to znamená, že se nemůže smrskávat a často to bude znamenat, že se nám vytrčí z prvku, na němž je flexbox
  + Flex-basis (nastavuje ideální šířku položky, jakou bychom si přáli, aby flexbox nastavil – flexbox ji zkusí nastavit)
    - Nastavuje výchozí, ideální velikost, jakou bychom si přáli, aby položka uvnitř flexboxu měla
    - Neznamená to, že tuto velikost mít bude, ale flexbox se bude snažit toto dodržet
    - Výchozí hodnota je auto, což znamená podle obsahu, respektive podle šířky, která byla nastavena té položce
    - Nastavuje se v délkových jednotkách (pixely, procenta, …)
    - Flex-basis: 50%
* Je výhodné mít tam to nastavení základní flex-shrink, jinak se nám ty věci dostávají mimo box, trčí z nich ven (i když někdy to můžeme potřebovat, samozřejmě
* Všechny ty vlastnosti fungují dohromady
* **Standardně se položky mohou smrskávat a nemohou se natahovat, tj. shrink 1, grow 0**
* Výchozí hodnoty:
  + **Flex-grow: 0 (nemám dovoleno se natahovat)**
  + **Flex-shrink: 1 (mám dovoleno se smrsknout)**
  + **Flex-basis: auto (standardně zabírám šířku podle obsahu)**
* **Flex: 0 1 auto; (grow, shrink, basis)**
* Ve flexboxu je možné měnit pořadí prvků
* Standardně je dá za sebe v tom pořadí, v jakém jsou v HTML (používá se například v responzivním webdesignu – zobrazení na mobilech)
* používá se k tomu vlastnost **order:** (přidáme, na jaké pozici má položka být)
  + order: 1
  + výchozí hodnota je nula, položky se na obrazovce seřadí v pořadí podle toho, jaký mají order
  + pokud jsme jedničku výše přiřadili dvojce, posune se nám na poslední místo proto, že předchozí položky mají výchozí hodnotu 0
  + když jej nastavíme, musíme počítat s tím, že všechny položky, jimž jsme order nenastavili, mají základní automatickou hodnotu 0
  + order může nabývat i záporných hodnot
  + když napíšu -1, bude dvojka na prvním místě

JE POTŘEBA VŽDYCKY MYSLET NA TO, CO SE STANE, KDYŽ BUDE TEXTU MÍŇ – PROTO BYLO V TOM CVIČENÍ POTŘEBA NASTAVIT FLEX-GROW NA 1, ABY SE V PŘÍPADĚ MALÉHO TEXTU VŽDYCKY TEN DIV ROZTÁHL NA CELOU STRÁNKU