**Vysoká škola ekonomická v Praze**

**Fakulta informatiky a statistiky**

**Katedra informačního a znalostního inženýrství**

Studijní program: Aplikovaná informatika

Obor: Aplikovaná informatika

**CSS/JS knihovna pro zjednodušení vývoje grafického rozhraní webových stránek**

**BAKALÁŘSKÁ práce**

Student : Martin Škára

Vedoucí : Ing. et Ing. Stanislav Vojíř, Ph.D.

**2018**

Prohlášení:

Prohlašuji, že jsem bakalářskou práci zpracoval samostatně a že jsem uvedl všechny použité prameny a literaturu, ze které jsem čerpal.

V Praze dne TODO . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . . .   
 Martin Škára

Poděkování

TODO

Abstrakt

TODO

Klíčová slova

TODO

Abstract

TODO

Keywords

TODO

Obsah

TODO

Úvod

Jednou z částí vývoje, kterou musí zpravidla každá webová stránka projít, je tvorba grafického rozhraní. Jelikož ale na stránkách můžeme často najít společné prvky či styly, vytvářet tak vše stále znovu by bylo velmi časově náročné a tudíž kontraproduktivní. I proto vznikají znovupoužitelné knihovny, které vývoj grafického rozhraní zjednodušují a často právě stránce definují i základní vzhled. Většina těchto knihoven je velmi obsáhlá a k některým problematikám přistupuje různě. Pro vývojáře může tak vyvstat otázka, kterou z těchto knihoven zvolit či případně nevyvinout nástroj vlastní, který bude plně vyhovovat jeho požadavkům.

Autor práce je zaměstnancem firmy Appio Digital s.r.o. (dále Appio), kde působí jako webový kodér. Věnuje se tak zejména vývoji uživatelského a grafického rozhraní. Na starost má zejména projekty menšího rozsahu, které běží na PHP frameworku Symfony (<https://symfony.com/>). Nástroje pro tvorbu grafického rozhraní si pak určují sami kodéři. Momentální situace je taková, že každý z projektů je tvořen pomocí jiné knihovny či základu a některé z komponent webových stránek (např. menu) nejsou vytvářeny jako znovupoužitelné, což brzdí vývoj zejména z časového hlediska. Na místě by tak bylo používat pouze jeden stálý nástroj pro tvorbu takových projektů. Jelikož autorovi práce (a některým jeho kolegům) nevyhovují již hotové knihovny, rozhodl se vyvinout nástroj vlastní, který bude čerpat z již vytvořených projektů a bude tak korespondovat s požadavky autora i vývoje v jeho zaměstnání. Jeho cílem nebude, na rozdíl od většiny hotových knihoven náhrada grafického rozhraní, ale měl by pomoci s vývojem rozhraní dle různorodého grafického návrhu.

V první kapitole *Existující knihovny pro řešení grafického rozhraní webových stránek* jsou rozebrány již hotová řešení (knihovny), jež se autor snaží analyzovat zejména z hlediska potřeb budoucích projektů a některými těmito řešení se nechává dále inspirovat v samotné tvorbě knihovny. Další tři kapitoly se věnují již vývoji. Jsou zde popsány technologie, kterými byla knihovna vytvořena, návrh a vlastnosti knihovny a následně i samotná realizace. V kapitole 5 se pak autor věnuje vytvoření modelových stránek a příkladů, s jejichž pomocí chce demonstrovat možnosti vytvořeného nástroje. Práce tak může sloužit jako příklad pro další vývojáře, kteří by se chtěli pustit do vývoje obdobné knihovny. Autor předpokládá, že čtenář má alespoň základní povědomí o vytváření webových stránek.

Cíle práce

Autor si tak prací stanovuje splnit následující cíle:

**Analýza stávajících řešení**

Zhodnocení vybraných stávajících řešení (knihoven) pro tvorbu grafického rozhraní webových stránek, uvedení důvodů, kvůli kterým autor vyvíjí knihovnu vlastní.

**Návrh a vytvoření knihovny zjednodušující vývoj grafického rozhraní webových stránek**

Návrh struktury a obsahu knihovny a následná realizace.

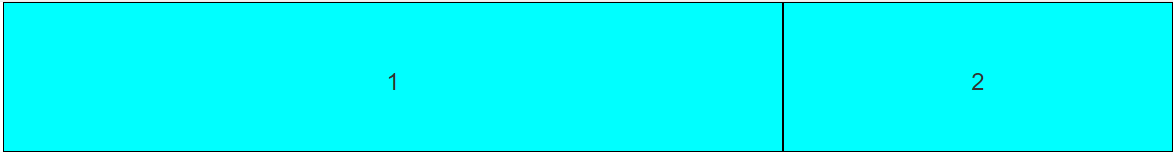
**Vytvoření modelových stránek pro demonstraci možností realizované knihovny**

Po dokončení stabilní verze knihovny vytvořit s její pomocí několik modelových stránek obsahujících většinu komponent, které knihovna nabízí.

1. Analýza existujících knihoven pro řešení grafického rozhraní webových stránek

Před vlastním vytvářením knihovny pro řešení grafického rozhraní webových stránek je namístě nejdříve analyzovat hotová řešení, z nichž se lze poučit či inspirovat.

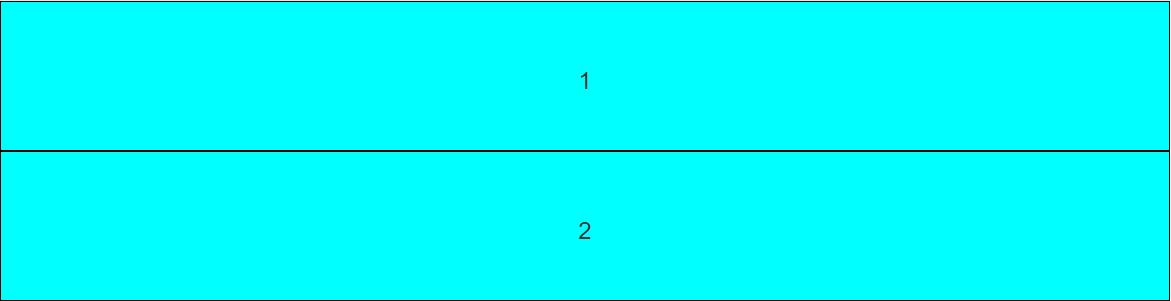
S rozmachem responzivního webdesignu, kdy se stránka navrhuje tak, aby se přizpůsobila uživatelovu zařízení, je jednou z nejpodstatnějších částí těchto knihoven grid systém, který slouží pro pozicování a zarovnávání prvků do sloupců a řádků, a tak je zcela jistě namístě vysvětlit běžné chování takového systému. Představme si prvky (sloupce), které chceme zarovnávat do řádku. První z nich zabírá dvě třetiny tohoto řádku, druhý zbylou třetinu (obrázek 1) – nicméně pouze při velkém rozlišení obrazovky. Pokud si však uživatel zobrazí webovou stránku na menším displeji (např. tabletu), chceme, aby každý z prvků zabíral půlku řádku (obrázek 2), pokud ještě na menším (např. mobilním telefonu), tak budou prvky zabírat celý řádek a budou zarovnány pod sebou (obrázek 3). Zápisy tohoto příkladu v jazyku HTML, který slouží pro zápis struktury webových stránek, jsou pomocí jednotlivých knihoven ukázány v dalších podkapitolách o jednotlivých knihovnách. [1], [2]



Obrázek 1 - Zobrazení uvažovaných prvků na velkém rozlišení obrazovky [autor]



Obrázek 2 - Zobrazení uvažovaných prvků při menším rozlišení obrazovky [autor]



Obrázek 3 - Zobrazení uvažovaných prvků na nejmenším rozlišení obrazovky (např. mobilním telefonu) [autor]

Responzivní grid tak umožňuje vývojáři manipulovat s velikostí a uspořádáním prvků v závislosti na uživatelově rozlišení obrazovky (nebo velikosti okna prohlížeče).

Mimo grid systém obsahují tyto knihovny většinou sadu hotových komponent pro webové stránky a pomocné třídy. Autor se tak v této kapitole zajímá zejména o tyto tři součásti takových knihoven a také o technologie ve kterých byly tyto knihovny vytvořeny. Pro analýzu si vybral knihovny Bootstrap (<https://getbootstrap.com/>), Foundation (<https://foundation.zurb.com/>), Pure CSS (<https://purecss.io/>) a Bulma (<https://bulma.io/>).

* 1. Bootstrap

Bootstrap byl původně vytvořen v roce 2010 vývojáři a designery ve společnosti Twitter (<https://twitter.com/>), již od začátku byl vyvíjen jako open-source software, tj. kdokoliv jej může používat, upravovat či vylepšovat. Stejně jako naprostá většina podobných knihoven je jeho aktuální verze (4.0) vyvíjena v jazyku CSS (což je jazyk sloužící k definování vzhledu HTML prvků), respektive jeho preprocesoru SASS (<https://sass-lang.com/>), jazyku postaveného nad CSS mající více možností zjednodušující vývoj, následně kompilovaného právě do jazyka CSS. Existují však i verze v jiných preprocesorech. Pro některé své části využívá též JavaScript, jazyk, který se spouští ve webovém prohlížeči uživatele a s jeho pomocí je možno manipulovat s webovým dokumentem a reagovat na akce uživatele. Ten je v rámci Boostrapu využíván pomocí knihoven jQuery (<http://jquery.com/>), díky které lze zjednodušit zápis některých javascriptových funkcí a Popper.js (<https://popper.js.org/>) pro správné fungování některých komponent. [3], [7]

Bootstrap resetuje některé defaultní styly HTML prvků (převážně těch textových) a definuje jim i vzhled bez toho, aniž by vývojář de facto chtěl (bez přiřazení příslušných CSS tříd). Toto autor práce nepovažuje za správné, jelikož vzhled těchto prvků se často liší projekt od projektu, a tak je například zbytečné definovat barvu textu odstavce. Defaultně knihovna používá nativní fonty operačních systémů (dále systémové fonty) dle článku *Using UI System Fonts In Web Design: A Quick Practical Guide* od Marcina Wicharyho. Tím je tak docíleno toho, že defaultně není potřeba využívat žádný externí font a zároveň je možno zachovat hezký defaultní vzhled na velké skupině zařízení. Zápis těchto nativních fontů v rámci CSS vlastnosti font-family, která definuje používaný font, může pak vypadat následně: [3], [8]

font-family: -apple-system, BlinkMacSystemFont,

“Segoe UI“, “Roboto“, “Oxygen“,

“Ubuntu“, “Cantarell“, “Fira Sans“,

“Droid Sans“, “Helvetica Neue“, sans-serif;

Kód 1 - Zápis nativních fontů v rámci CSS vlastnosti font-family [8]

Pro velikost fontů (i jiných rozměrů) pak Bootstrap používá jednotky rem, které se odvozují od velikosti písma prvku <html>. Bootstrap nechává tuto velikost na 100 % písma v uživatelově prohlížeči (většinou 16 pixelů, pixel přestavuje jeden svítící bod na monitoru či v obrázku), od ní pak odvozuje zbylé velikosti. [3]

Pro velikosti grid systému jsou však použity pixely. Grid systém tvoří třída container, která definuje rozměry obsahu stránky a zarovnává jej na střed. V rámci ní lze použít třídu row, což je rodičovský prvek, v němž jsou následně definovány jednotlivé sloupce. Zápis příkladu z obrázků 1-3 pomocí Bootstrapu je ukázán v kódu 2. [3]

<div class=“container“>

<div class=“row>

<div class=“col-12 col-sm-6 col-lg-8“>

1

</div>

<div class=“col-12 col-sm-6 col-lg-4“>

2

</div>

</div>

</div>

Kód 2 - Zápis uvažovaného příkladu z obrázků 1-3 pomocí knihovny Bootstrap [autor]

Sloupce tak mají povinný prefix *col-* za nímž následuje dvoumístná definice breakpointu (tj. bodu rozlišení obrazovky, kdy se chování sloupce změní). Boostrap má defaultně, ale s možností změny definované takové body 4 – sm pro 576 pixelů, md pro 768 pixelů, lg pro 992 pixelů a xl pro 1200 pixelů. Pokud se jedná o chování pod prvním breakpointem, nepíše se nic. Grid systém je mobile-first, což znamená, že defaultně se vzhled prvků definuje pro mobilní zařízení, a právě pomocí zmíněných breakpointů a CSS vlastností Media Queries k tomu sloužících se postupně definuje chování pro větší zařízení a displeje. Poslední číslo ve třídě pak znamená, kolik sloupců prvek zabírá. Pokud se číslo rovná maximálnímu počtu sloupců grid systému (v Bootstrapu defaultně nastaveno na 12, ale dá se změnit), roztáhne se prvek do šířky celého řádku. Mimo tyto třídy je ale možno použít i prostou třídu *col*, prvky si pak místo v řádku rozdělí zcela rovnoměrně. Mezery mezi sloupci jsou defaultně nastavené na 30 pixelů, pokud to vývojář nezmění. [3]

Grid však nabízí více možností – je totiž vytvořen pomocí CSS vlastností flexbox, které určují rodičovskému prvku (Flex container, flex kontejner), jak velké místo v něm budou zaujímat a jak se budou zarovnávat prvky obsažené (Flex items, flex položky). Je tak možné jednoduše měnit zarovnávání prvků do řádku či sloupce, určovat, jak velký bude určitý Flex item oproti ostatním, jak se budou moci měnit flexibilně jeho rozměry, v jakém pořadí prvky řadit apod. [15]

Boostrap tak díky flexboxu nabízí rodičovskému prvku row třídy, které následně obsažené sloupce mohou vertikálně i horizontálně zarovnávat i různorodě v rámci zmíněných breakpointů. [3]

Pro jednotlivé sloupce pak nabízí třídy pro změnu pořadí prvku, či i pro odsazení prvku, chování jde opět měnit dle rozlišení obrazovky. Tyto třídy považuje autor práce za zbytečné. Odsazení sloupce zleva či zprava není dle něj tak časté. Určité případy by se daly řešit pomocí tříd pro zarovnání v rodičovském prvku nebo vytvořením prázdného sloupce, který uživatel neuvidí. Stejně tak je to se změnou pořadí prvku, nejde o častý případ a často si lze vystačit pouze s tím, že prvek dáme na první či poslední místo. Bootstrap však nedefinuje třídy jen pro první a poslední místo v pořadí, ale pro tolik míst z kolika sloupců se skládá grid systém. I to považuje autor práce za nepřesné, jelikož i když bude grid systém definován jako dvanáctisloupcový, řádek může sloupců obsahovat více (řádek se pak zalomí).

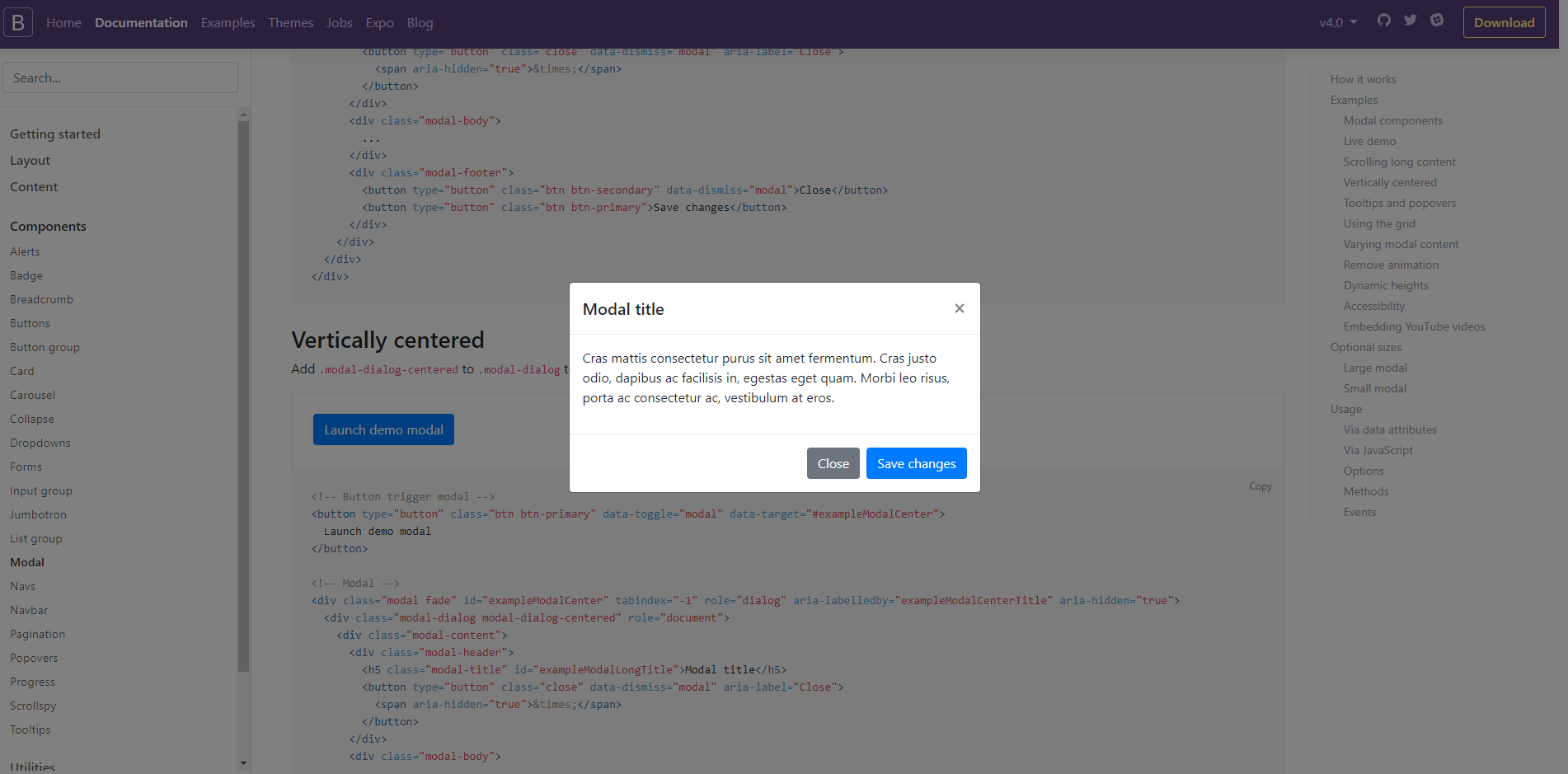
Z hlediska hotových komponent Bootstrap obsahuje většinu ze známých komponent, se kterými se lze na webových stránkách setkat. Za zmínku stojí menu a navigace, různá tlačítka, drobečková navigace (seznam odkazů pro nadřazené sekce), carousel (obrázek 4), formulářové prvky a modal okna (obrázek 5). Nevýhodou Bootstrapu je bohužel to, že i po úpravě těchto komponent může být vizuálně poznat, že vyšly právě z této knihovny – a tak je dnes v důsledku toho mnoho bootstrapových webových stránek velice podobných.

U komponenty responzivního menu autor práce nalezl chybu. Pokud ji použijeme s fixní pozicí (tj. prvek zůstává na stále stejném místě v okně prohlížeče) a zobrazíme si ji na úzkém displeji (např. mobilním telefonu v zobrazení na šířku), tak v případě, že menu obsahuje více položek uživatel některé neuvidí a nijak se k nim nedostane, což může být poněkud frustrující.

Ohledně pomocných tříd – těch nabízí Boostrap celou řadu. Co mu lze vytknout je možná zbytečnost některých těchto tříd, jelikož jejich účelu lze dosáhnout pomocí grid systému (např. třídy pro zarovnávání). Vzhledem k odlišnosti jednotlivých webových stránek pak nepovažuje autor za vhodné pomocné třídy pro barvy rámečku č pozadí – můžou se hodit pro web spoléhající pouze na samotnou knihovnu, ale nepomohou při vývoji dle zadaného grafického návrhu.



Obrázek 4 - Ukázka komponenty Carousel vytvořené pomocí knihovny Bootstrap. Carousel je všeobecně uznávaný název pro tento druh komponenty. [9]



Obrázek 5 - Ukázka komponenty Modal okna vytvořené pomocí knihovny Boostrap. Modal (či Modal okno) je všeobecně uznávaná název pro tento typ komponenty.[3]

* 1. Foundation

V roce 2008 vytvořila společnost ZURB (<https://zurb.com>), která se zabývá vývojem softwaru, nástroj pro zjednodušení své práce v jazycích HTML, CSS a JavaScript. Nazvala jej ZURB Style Guide. Tento nástroj byl dále rozvíjen a v roce 2010 byl přejmenován na Foundation. Od roku 2011 je pak dostupný open-source. Aktuální verze je Foundation 6. Stejně jako Boostrap je vyvíjen v preprocesoru SASS a využívá taktéž javascriptovou knihovnu jQuery. [4]

Pro resetování některých defaultních stylů používá Foundation Normalize.css (<https://necolas.github.io/normalize.css/>), což je soubor CSS stylů, který zařizuje vykreslování některých HTML prvků tak, aby se napříč prohlížeči zobrazovaly konzistentněji. Často obsahuje i definice stylů pro prvky, které se často tolik nevyužívají – nicméně vzhledem k množství obsáhlých komponent a stylů ve Foundation to zrovna v tomto případě vůbec nevadí. Pro rozměry využívá zejména jednotky rem, ale například breakpointy grid systému jsou definovány pomocí jednotky em, která vychází z velikosti písma rodičovského prvku, tj. pokud má rodičovský prvek velikost písma nastavenu na 10px, pak 1em = 10px. V definici breakpointů pomocí Media Queries se však tato jednotka chová stejně jako rem, tj. orientuje se podle velikosti písma prvku <html>. Vývojáři Foundation se rozhodly pro použití jednotek em v tomto případě zřejmě proto, že použití jednotek rem v Media Queries způsobovalo problémy na webových prohlížečích Safari (na nejnovějších verzích toto ale bylo opraveno). Používá taktéž systémové fonty. [4]

V rámci používání Foundation si vývojář může vybrat (alespoň u některých komponent a grid systémů), zdali chce využívat flexbox nebo ne. Tato volitelná možnost je zde zejména kvůli kompatibilitě napříč prohlížeči, která není stoprocentní, autor práce je však toho názoru, že je dnes již bezpečné jej využívat. S tím souvisí i to, že Foundation nabízí celkově tři různé grid systémy. [4]

Pokud se vývojář rozhodne flexbox nepoužívat, může tak použít *Float Grid*, grid systém využívající CSS vlastností float a procentuálních šířek. Float je původně vlastnost určující obtékání prvku, často (jako v rámci tohoto grid systému) se využívá pro zarovnání prvků vedle sebe. Defaultně má, stejně jako všechny grid systémy knihovny Foundation, 12 sloupců s možností změny. Zápis uvažovaného příkladu z obrázků 1-3 bude vypadat následně:

<div class=“row“>

<div class=“columns small-12 medium-6 large-8“>

</div>

<div class=“columns small-12 medium-6 large-4“>

</div>

</div>

Kód 3 – Zápis uvažovaného příkladu z obrázků 1-3 pomocí Float Grid/Flex Grid systémů knihovny Foundation [autor]

Defaultně má Foundation nastaveny pouze dva breakpointy, přepočítány z jednotek em vychází většinou na 640 a 1024 pixelů. Více breakpointů lze sice nastavit, ale i tak by se autor práce přikláněl k více i v defaultním nastavení. Oproti Bootstrapu si zde v kódu 3 všimnout, že zde není prvek se třídou container. Šířku obsahu totiž kontroluje přímo třída row – nabízí se tak otázka, zdali by nebylo lepší vlastnosti pro definování grid systému a šířky obsahu rozdělit do více tříd. To se autorovi práce nezamlouvá, nicméně je spokojen se zápisem tříd pro sloupce, kdy se definuje základní třída pro sloupec (v tomto případě columns) a následně se pomocí dalších tříd definuje chování v rámci breakpointů. Float Grid nabízí taktéž třídy pro odsazování prvků, změnu pořadí a centrování. Ohledně odsazování prvků, třídy nejsou tolik užitečné, jelikož většina případů by se i v tomto grid systému dala řešit pomocí prázdných sloupců či centrování. Třídy pro změny pořadí jsou užitečnější, využívají vlastností float a pozicování, v případě složitějších případů (kdy chceme měnit pořadí tří prvků a více), je pomocí nich dosáhnout kýženého cíle velice nesnadné. To stejné platí u centrování sloupců. Centrovat je pomocí obsažených tříd lze, ale centrované sloupce se zobrazují automaticky pod sebou. Nelze tak bez dalších úprav například dva menší sloupce zarovnávat vedle sebe na střed. Je to škoda, protože zarovnávání obsahu na střed je poměrně častý případ, jehož jednoduché nastavení by dle autora práce měl obsahovat každý dobrý grid systém. [4]

Pokud se vývojář rozhodne flexbox využít, může si vybrat ze dvou grid systémů, které jej používají. Jedná se o *Flex Grid* a *XY Grid*. První z jmenovaných je vlastně jen již popsaný Float Grid vylepšený o vlastnosti flexboxu. I jeho zápis vypadá na první pohled zcela stejně (kód 3). Taktéž by se dalo říci, že nabízí obdobné možnosti jako grid systém knihovny Bootstrap. Není třeba tak dál vysvětlovat jeho vlastnosti. [4]

Nejzajímavějším grid systémem Foundation (a zároveň defaultním) je tak právě XY Grid. Nejprve je vhodné ukázat zápis námi uvažovaného příkladu:

<div class=“grid-container“>

<div class=“grid-x grid-padding-x“>

<div class=“cell small-12 medium-6 large-8“>

</div>

<div class=“cell small-12 medium-6 large-2“>

</div>

</div>

<div>

Kód 4 - Zápis uvažovaného příkladu pomocí grid systému XY Grid knihovny Foundation [autor]

Jak lze vidět, zápis je poněkud složitější, než v případě systému Float Grid (resp. i Flex Grid). Třída grid-container určuje šířku obsahu. Třída grid-x je rodičovský prvek pro jednotlivé sloupce. X na konci třídy značí, že se sloupce zarovnávají vedle sebe. Y by znamenalo, že se řadí pod sebe (tzv. vertikální grid systém). Následná třída grid-padding-x udává sloupcům, že se mezi nimi mají tvořit mezery (což je u většiny grid systémů defaultně přednastaveno, např. u zbylých grid systémů Foundation je to 0,9375rem ≈ 30 pixelů) a v jakém směru (x/y). Obdobně jsou obsaženy i třídy grid-margin-x (a grid-margin-y). Vývojář si tak může určit, zdali mezery budou vytvořeny pomoci CSS vlastnosti margin (vnější okraje prvků) či padding (vnitřní okraje prvků). Následují pak definice samotných sloupců, kde se na rozdíl od zbylých Foundation grid systémů nazývá definující buňka cell, nikoliv columns. Kromě definicí velikostí lze používat i třídy shrink a auto. První definuje sloupec široký pouze pro potřebu jeho obsahu a druhá sloupce, které se rovnoměrně rozprostřou do zbývajícího prostoru řádku (obrázek 6). [4]



Obrázek 6 - Ukázka sloupců s třídami shrink a auto. Sloupec s textem Shrink má šířku postačujícím rozměrům jeho obsahu a druhý sloupec se rozprostřel do zbytku prostoru řádku [4]

Samozřejmě lze využívat i flexboxové třídy pro horizontální i vertikální zarovnávání prvků či změnu pořadí. Poslední vlastností XY Grid systému je třída grid-frame, čerpající z příbuzného frameworku Foundation for Apps (<https://foundation.zurb.com/apps.html>). Slouží převážně k vytváření celkového layoutu (rozložení) stránky pomocí flexboxu. Celkově je tak XY Grid jedním z nejkomplexnějších grid systémů vůbec (nebo co se alespoň týče systémů popsaných v této práci), dle autora práce je však příliš složitý a všechny jeho možnosti by sám zřejmě nevyužil. [4]

Co se týče již hotových komponent, rozděluje je Foundation do několika kategorií – controls (ovládací prvky), navigations (menu, stránkování, drobečková navigace aj.), containers (rodičovské prvky dalšího obsahu, např. rozklikávací záložky, karty, modal okna, tabulky) a media (štítky, videa, progress bary). Oproti Bootstrapu je jejich defaultní grafický styl více odlehčený, což může být užitečné zejména v případě, že hotovou komponentu chceme použít při vývoji dle různorodého grafického návrhu.

Pomocné třídy jsou pak v knihovně velmi důležité, zejména pak ty týkající se vlastností flexboxu, jelikož se využívají i v rámci grid systémů. Jinak zahrnují většinu vlastností CSS a pro velkou část z nich existují i jejich responzivní ekvivalenty (tj. s prefixem dle velikosti obrazovky, small-, medium-, large-), což ale poněkud zvětšuje velikost výsledného CSS souboru.

* 1. Pure CSS

Pure CSS je knihovna vznikající od roku 2013. Již defaultně je tvořena jako skupina několika balíků tříd – base, grids, forms, buttons, tables, menus. Její největší výhoda je její opravdu malá velikost výsledného CSS souboru. Je vyvíjena v samotném jazyku CSS a nepoužívá, na rozdíl od větších knihoven, JavaScript (ani jQuery). [5]

Pro resetování stylů využívá knihovna, stejně jako Foundation známý soubor Normalize.css a taktéž systémové fonty. Pro většinu velikostí používá pak jednotky em, případně px (pro definici breakpointů, některých horizontálních mezer aj.). [5]

Grid systém, obsažen v balíku grids, má v základním nastavení definované 4 breakpointy (sm 568 pixelů, md 768 pixelů, lg 1024 pixelů a xl 1280 pixelů) a je mobile-first. Zápis uvažovaného příkladu z obrázků 1-3 v něm vypadá takto:

<div class=“pure-g“>

<div class=“pure-u-5 pure-u-md-12-24 pure-u-lg-18-24“>

</div>

<div class=“pure-u-5 pure-u-md-12-24 pure-u-lg-6-24“>

</div>

</div>

Kód 5 – Zápis uvažovaného příkladu z obrázků 1-3 v grid systému knihovny Pure CSS [autor]

Třída pure-g je rodičovský prvek pro jednotlivé sloupce, na rozdíl od jiných systémů neurčuje šířku obsahu (taková třída v Pure CSS není). Jednotlivé sloupce určují třídy s prefixem *pure-u.* Následně se dvěma písmeny definuje breakpoint (či se vynechává v případě definice sloupců pod 568 pixelů) a poté dvě čísla. První znamená počet sloupců, které prvek bude zabírat a druhé z kolika sloupců. Pure CSS totiž nabízí naráz třídy pro pětisloupcový i čtyřiadvacetisloupcový grid systém. Tyto třídy se dají naráz kombinovat u jednoho prvku. Na rozdíl od jiných grid systémů nenabízí žádné pomocné třídy pro zarovnávání obsahu, odsazování, změnu pořadí sloupců atd., z nichž některé by autorovi práce podstatně chyběly. [5]

Hotové komponenty lze nalézt v balíkách forms, buttons, tables a menus. Z jejich názvu lze vyčíst, jaké komponenty obsahují. Jejich vzhled lze snadno upravit, ale bohužel zde nelze nalézt složitější komponenty – k nim by totiž zřejmě bylo potřeba použít JavaScript, který v knihovně není obsažen. Z pomocných tříd obsahuje Pure CSS pouze třídy pro skrývání prvků a zobrazování obrázků. [5]

* 1. Bulma

Bulma je CSS knihovna vyvíjecí se jako open-source od roku 2016. Jedná se tak, vzhledem k vzrůstající popularitě této knihovny, o jednu z nejrychleji se rozvíjejících knihoven. Pro srovnání například Foundation, dostupné veřejně od roku 2011, má k dnešnímu datu (24. 2. 2018) na serveru Github (<https://github.com>) přidáno v oblíbených cca 27 000 uživatelů a Bulma je na tom podobně – skoro 25000. Knihovna je vyvíjena v preprocesoru SASS. [6]

Pro resetování některých defaultních stylů používá Bulma příbuzný soubor minireset.css. Jeho autorem je vlastně autor samotné knihovny Bulma, Jeremy Thomas. Používá v něm, stejně jako naprostá většina ostatních knihoven, systémové fonty. Jako jednotky využívá knihovna dle potřeby jak em, tak rem i pixely. [6]

Grid systém knihovny Bulma má v základním nastavení 5 breakpointů – tablet 769px, desktop 1024px, widescreen 1216px a FullHD 1408px. Zápis uvažovaného příkladu z obrázků 1-3 vypadá pak takto:

<div class=“container“>

<div class=“columns“>

<div class=“column is-12-mobile is-6-tablet is-8-desktop“>

</div>¨

<div class=“column is-12-mobile is-6-tablet is-4-desktop“>

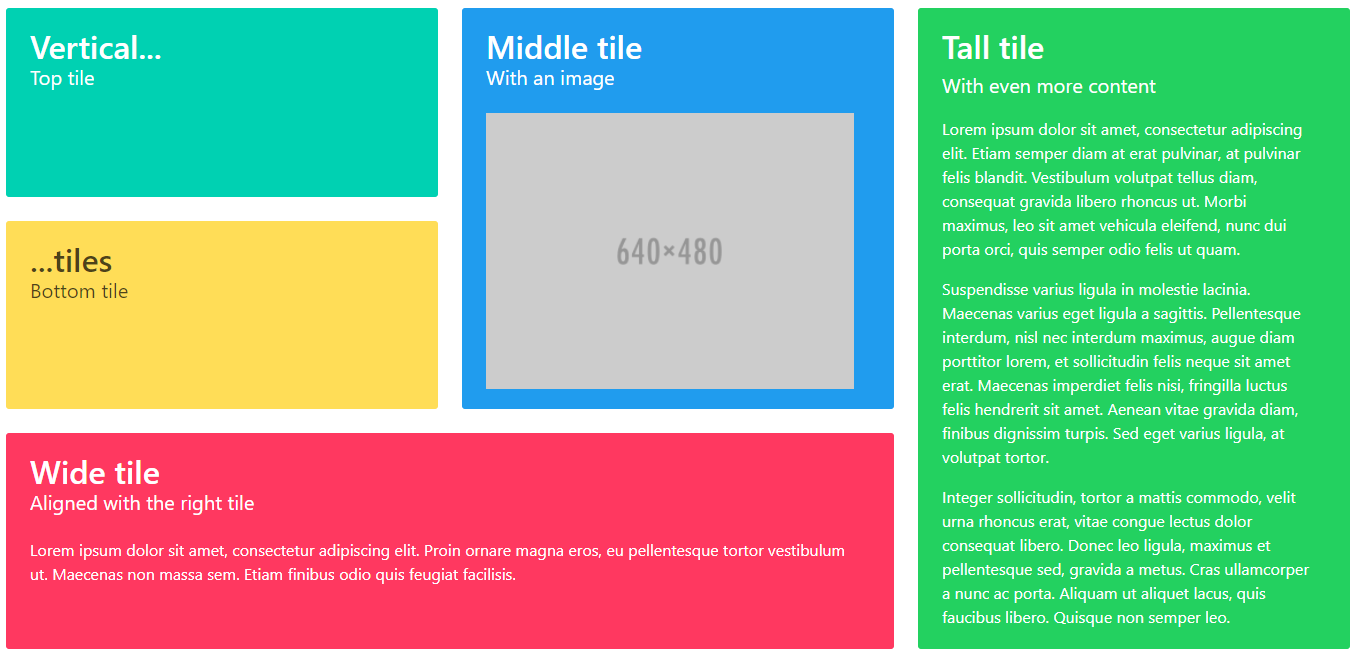
</div>

</div>

</div>

Kód 6 - Zápis uvažovaného příkladu z obrázků 1-3 v grid systému knihovny Bulma [autor]

Třída container určuje šířku obsahu, columns je rodičovská třída pro následné sloupce. Jejich třídy připomínají poměrně třídy knihovny Foundation. Krom tříd ve tvaru is-(číslo)-(breakpoint) nabízí Bulma třídy jako např. is-three-quaters-(breakpoint), is-two-thirds-(breakpoint), is-half-(breakpoint) a podobně. Jedná se o třídy definující anglicky poměr, jaký bude sloupec v řádku zabírat. Autorovi práce tyto třídy přijdou zbytečné – stačily by dle něj třídy ve tvaru ukázaném v kódu 6. Bez těchto tvarů pak třída column funguje tak, že si rozdělí zbylý prostor v řádku. Stejně tak lze sloupci přidat třídu is-narrow, která funguje stejně jako třída shrink v knihovně Foundation (viz obrázek 6). Mimo to grid systém nabízí opět třídy pro centrování a odsazování prvků. Zvláštní je, že v samotné dokumentaci je psáno, že odsazování sloupců může být řešeno použitím prázdných sloupců, avšak stejně jsou třídy odsazení obsaženy. Použití tříd je sice pro vývojáře pohodlnější, avšak navyšuje velikost výsledného CSS souboru. Mimo, dá se říci tradiční grid systém, nabízí Bulma i tzv. tile grid systém, který se skládá z dlaždic, které je schopen distribuovat jak v horizontálním, tak ve vertikálním směru. Příklad, který demonstruje co lze s jeho pomocí vytvořit, může vypadat takto:



Obrázek 7 - Příklad vytvořený pomocí tile grid systému knihovny Bulma [6]

Z hotových komponent obsahuje Bulma menu, patičku, formulářové prvky, různé karty, tlačítka, progress bary, tabulky, modal okna, rozklikávací záložky i textové prvky. Pojem hotové komponenty v tomto případě ale není úplně přesný. Bulma neobsahuje žádné soubory JavaScriptu, které by v normálním případě byly ke správnému fungování některých komponent (například modal okna) nutné. Komponenty jsou tak hotové pouze co se týče vzhledové stránky, takže některé z nich nemůže použít bez dalších úprav či dodělávek. Pokud vývojář tak vyvíjí stránku dle grafického návrhu, nejsou mu tak komponenty tolik užitečné, jelikož by ocenil spíše hotové chování komponent a ne vzhled. [6]

Bulma obsahuje pomocné třídy pro základní vzhledové vlastnosti některých komponent, zarovnávání i pozicování, zobrazování i textové vlastnosti. Třídy jsou poměrně užitečné, jelikož pokrývají nejčastější vlastnosti, které vývojáři používají. Jediné, co by autor práce vytknul jsou třídy určující barvu textů, tlačítek a dalších komponent. Často se nestane, že by je vývojář použil v případě vývoje dle grafického návrhu a přesto, že je lze v SASS souborech měnit, stále by tyto vlastnosti raději vynechal. [6]

* 1. Závěry z provedené analýzy

Autor v předešlých kapitolách provedl analýzu čtyř stávajících knihoven z hlediska použitých technologií, grid systému, hotových komponent a pomocných tříd. V této kapitole by rád uvedl, jak jednotlivá hlediska ovlivní vývoj jím vytvořené knihovny. Je jasné, že se knihovnami, které měl tu možnost analyzovat, mohl nechat v jednotlivých částech inspirovat.

**Použité technologie**

Nejlepším řešením bude dle autora knihovnu vyvinout v preprocesoru SASS. Preprocesor umožňuje knihovnu vyvinout rychleji a dává možnost případným vývojářům knihovnu lépe upravit svým požadavkům díky proměnným a funkci import, pomocí které snadno může vývojář určit, jaké části knihovny (ne)použije. I přes existenci více preprocesorů bude použit SASS, zejména proto, že většina projektů ve firmě Appio jej využívá. Pro správné fungování knihovny pak bude nutno použít jazyk JavaScript (případně jeho knihovny), jelikož autor chce, aby vytvořené komponenty byly plně funkční, jak tomu je u knihoven Bootstrap a Foundation. [3],[4]

Pro resetování některých stylů nepoužije autor žádný již vytvořený soubor, ale případně se nechá inspirovat jednotlivými CSS pravidly, které obsahují. Nebude taktéž udávat základní vzhled (jako barvu, velikost písma apod.) textovým prvkům, bude jim však možno přiřadit některé třídy komponent. Nerad by zahrnoval resetování stylů pro prvky, u kterých není předpokládáno použití tříd samotné knihovny. To ušetří velikost výsledného CSS souboru. Je možné, že s postupným vývojem knihovny a použitím na reálných projektech se však výsledné styly resetování budou upravovat a zvětšovat dle potřeby vývojářů. Defaultně, stejně jako všechny analyzované knihovny použije nativní fonty operačních systémů, jelikož nechce, aby nutil vývojářům načítat určitý font. Fonty se ve vyvíjených projektech často liší, a tak by to nebylo vhodné. Pro rozměry písma budou použity jednotky rem, pro horizontální rozměry a definice breakpointů pak pixely, stejně jako u knihovny Bootstrap. [3]

**Grid systém**

Grid systém by měl obsahovat třídu, která bude definovat šířku obsahu a následnou rodičovskou třídu, ve které budou definovány jednotlivé sloupce. Zápis sloupců se bude inspirovat knihovnou Foundation, tj. bude zde základní třída definující jednotlivé sloupce a dalšími třídami se bude deklarovat další chování. Grid bude defaultně dvanáctisloupcový s možností úpravy a bude využívat flexbox. Samozřejmostí tak bude obdoba tříd shrink a auto právě z knihovny Foundation. Zápis breakpointů ve třídách se však bude inspirovat Bootstrapem, jelikož na jeho zápis breakpointů je skupina vývojářů ve firmě Appio zvyklá. Grid systém bude obsahovat pomocné třídy vlastností flexboxu, ale nebude obsahovat třídy pro odsazování prvků a další, dle autora nepotřebné třídy zmíněné v předešlých kapitolách. [4]

**Hotové komponenty**

Z hlediska hotových komponent se autor příliš hotovými řešeními inspirovat nenechá, spíše bude vycházet z jichž vytvořených projektů. Obsahově by však nechtěl vytvářet přespříliš velké množství komponent, ale zároveň by jich mělo být více než v analyzované knihovně Pure CSS. Právě u komponent bude muset deklarovat prvkům i některý základní vzhled. Měl by být však co nejvíce decentní, aby jej šlo rychle upravit. To se dle jeho názoru například Bootstrapu v některých případech nedaří.

**Pomocné třídy**

U pomocných tříd bude nutno zhodnotit, zdali je vlastnost, kterou dané třídy budou pokrývat, natolik používaná, že ji má smysl zahrnovat do knihovny. Autor při vývoji zřejmě projde i pomocné třídy analyzovaných knihoven, aby posoudil, zda obdobné třídy nezahrne do knihovny jím vytvářené.

1. Použité technologie

V této kapitole budou popsány technologie, které budou použity při vytváření samotné knihovny. Mezi ně patří CSS a jeho preprocesor SASS, JavaScript a jeho knihovna jQuery a nástroj pro zkompilování knihovny – Gulp.

* 1. CSS

CSS (Cascading Style Sheets) je jazyk, který ovlivňuje vlastnosti zobrazení prvků v souborech vytvořených pomocí jazyka HTML a dalších. Jeho hlavním smyslem je oddělení vzhledových vlastností od obsahu jednotlivých prvků. Jeho současná verze je CSS3, starší verze nabízeli daleko méně možností a vlastností. Již od první verze CSS byla zavedena syntaxe vybírání prvků pomocí selektorů a nastavování jejich hodnot. Nastavovat šly vlastnosti písma a textu, barvy textu i pozadí, rozměry elementů a jejich pozice apod. Zápis CSS může vypadat například takto:

.text { /\*selektor třídy\*/

color: black;

font-size: 12px;

font-weight: bold;

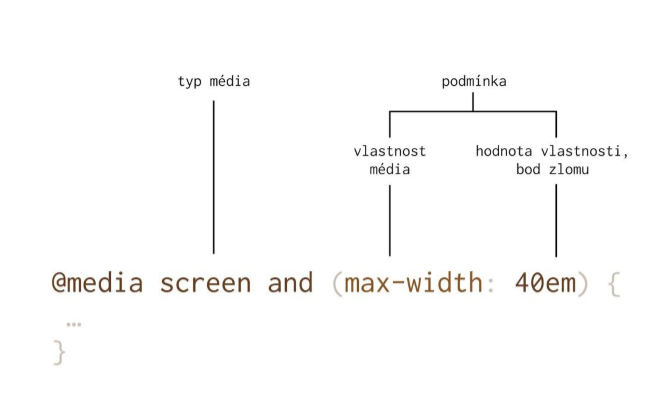
}

Kód 7 - Ukázka zápisu třídy v jazyce CSS [autor]

Důležitou funkcí CSS je v dnešní době tvorba layoutu stránky. V minulosti se pro tvorbu layoutu stránky používaly rámy či tabulky. V dnešní době se pro tvorbu layoutu se využívají vlastnosti CSS, HTML slouží pouze k znázornění struktury obsahu. Postupem času se využívalo zejména proměnlivých rozměrů prvků, float a display inline-block, absolutního pozicování či display table. Pomocí vlastností float a display inline-block jsou často tvořeny i grid systémy. S jejich pomocí lze zarovnávat prvky vedle sebe, což je základ pro tvorbu grid systému. Žádná z těchto vlastností však nebyla původně navržena pro tvorbu layoutu stránky, tudíž se dnes čím dál více přistupuje k novějším CSS modulům. Těmi jsou konkrétně flexbox a CSS Grid (modul jazyka CSS, nikoliv grid systém). Pro tvorbu responzivního layoutu jsou také důležitá pravidla Media Queries. Více v následujících kapitolách. [10], [11], [13], [15]

* + 1. Media Queries

Specifikace Media Queries udává v rámci jazyka podmínky, které umožní aplikovat určité CSS vlastnosti v určitých situacích. Dnes slouží zejména pro definování CSS dle velikosti okna prohlížeče a orientace zařízení. Jsou jedním z nejdůležitějších nástrojů pro tvorbu responzivního layoutu. Celkově lze nimi ale definovat i styly pro monitory s určitým rozlišením a poměrem, pro tisk a pro prohlížeče a zařízení se specifickými vlastnostmi. Typickou zjednodušenou strukturu Media Query lze vidět na obrázku níže. [14]

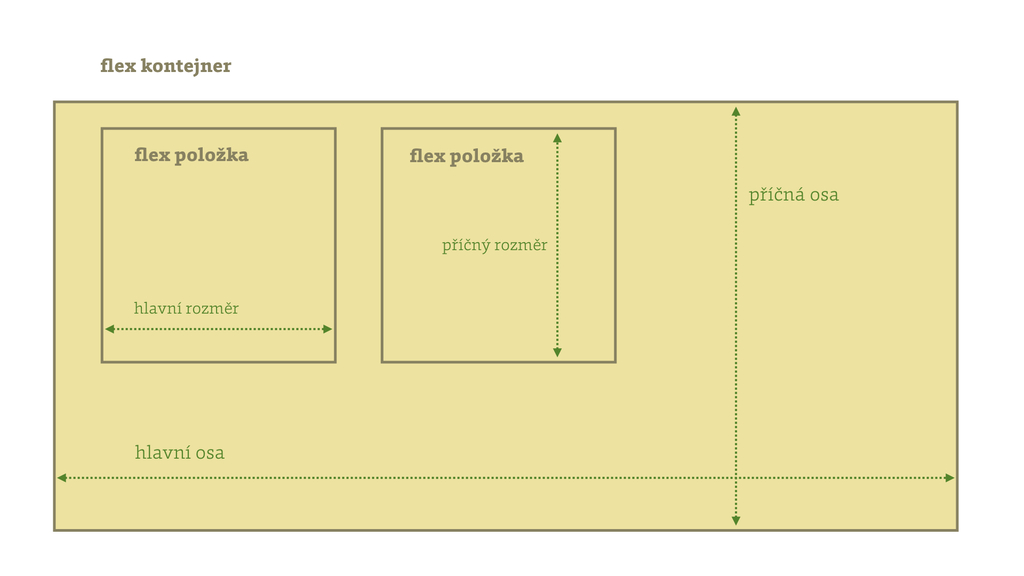
* + 1. Flexbox

Obrázek 8 - Zjednodušená struktura Media Query zápisu. [14]

Flexbox je modul CSS, jehož vlastnosti určují rodičovskému prvku (Flex container, flex kontejner), jak velké místo v něm budou zaujímat a jak se budou zarovnávat prvky obsažené (Flex items, flex položky). Schéma lze vidět na obrázku 9 dále. V defaultním nastavení manipuluje s prvky tak, že se snaží přizpůsobit jejich vzájemné rozměry jejich potřebnému obsahu. Prvky jsou tak flexibilní. [1]

Flex kontejneru lze nastavit následující vlastnosti: [1]

* **Flex-direction** – směr v jakém se uvnitř kontejneru budou řadit položky – definuje hlavní osu. Defaultně zleva do prava.
* **Flex-wrap** – zalamování řádků / sloupců položek, pokud jejich rozměry přesáhnou rozměr kontejneru. Defaultně se nezalamují.
* **Justify-content** – zarovnávání prvků ve směru hlavní osy. Defaultně se zarovnávají na začátek.
* **Align-items** – zarovnávání prvků v rámci příčné osy (tj. osy kolmé k ose hlavní). Defaultně se položky v kontejneru chovají tak, aby jejich příčný rozměr byl stejný jako příčný rozměr kontejneru.
* **Align-content** – zarovnávání řádků (či sloupců v případě, že je hlavní osou vertikální osa) v rámci kontejneru. Defaultně si řádky rozměry rovnoměrně vyplní tak, aby zaplnily celý příčný rozměr kontejneru.

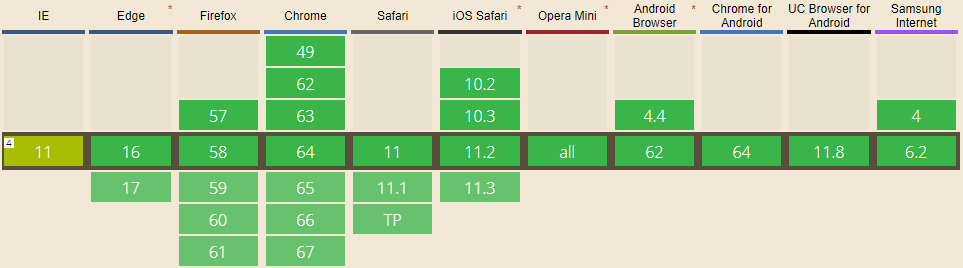


Obrázek 9- Schéma použití flexboxu [1]

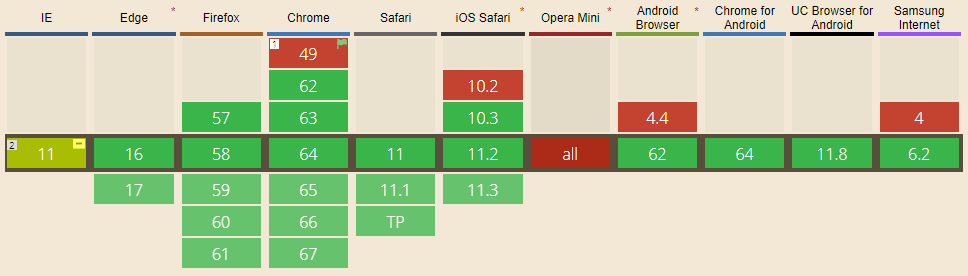
Jednotlivým flex položkám pak lze nastavit následující vlastnosti: [1]

* **Order** – určuje pořadí položky v rámci flex kontejneru
* **Flex-grow** – určuje schopnost položky zvětšit svůj rozměr oproti ostatním položkám, pokud je to nutné
* **Flex-shrink** – určuje schopnost položky zmenšit svůj rozměr oproti ostatním položkám, pokud je to nutné
* **Flex-basis** – Defaultní rozměr položky, předtím, než si položky rozdělí místo v kontejneru dle vlastností flex-grow a flex-shrink
* **Align-self –** Přetěžuje vlastnost align-items v kontejneru, ale pouze právě u jedné položky.

I přes jednoduché použití vyčtených vlastností, se stále k flexboxu mezi vývojáři přistupuje s jistou obezřetností. Důvodem jsou problémy a podpora flexboxu v různých prohlížečích. Autor práce je však toho názoru, že obavy již dnes nejsou na místě. Podpora flexboxu v prohlížečích dle serveru Can I use (<https://caniuse.com>) ukázána na následujícím obrázku 10. Pro srovnání je pak na obrázku 11 prohlížečová podpora moderního modulu CSS Grid, který však vzhledem k nižší podpoře a menší vhodnosti pro tvorbu layoutu komponent (více se hodí ke tvorbě layoutu celé stránky), autor práce nevyužil.



Obrázek 10 - Podpora flexboxu ve webových prohlížečích [20]



Obrázek 11 - Podpora CSS Grid ve webových prohlížečích [20]

Na obrázcích 10 a 11 lze vidět podporu prohlížečů, které využívá alespoň 0,5 % uživatelů či případně novějších verzí takových prohlížečů. Aktuální verze jsou vyznačeny černým rámečkem. Zelená barva znamená plnou podporu daného modulu, červená žádnou. Můžeme tak říci, že dnes používanými prohlížeči je flexbox již podporován. Jediný problém je s prohlížečem Internet Explorer. V obrázku 10 je vyznačen žlutou barvou, jelikož flexbox sice podporuje, ale s chybami. Modul CSS Grid pak již i některé dnes více používané prohlížeče nepodporují, což je problém. Starší verze iOS Safari, Opera Mini, Android Browser 4.4 a Samsung 4 nepodporují tento modul vůbec. Internet Explorer pak podporuje pouze starší syntaxi tohoto modulu a v prohlížeč Google Chrome dokáže sice tento modul podporovat, ale pouze pokud se upraví jeho nastavení. To ale běžný uživatel téměř nikdy neudělá. CSS Grid tak na rozdíl od flexboxu nemá dnes tak spolehlivou podporu. [20]

* 1. Preprocesory CSS

Preprocesory jsou jazyky postavené nad CSS, které je následně nutné zkompilovat právě do CSS souborů. Využívány jsou proto, že na rozdíl od jazyka CSS nabízejí více funkcí a zjednodušují díky tomu práci. Mezi nejznámější preprocesory patří již zmiňovaný SASS, mimo něj pak LESS (<http://lesscss.org/>) a Stylus (<http://stylus-lang.com/>). I přesto, že se jednotlivé preprocesory mohou lišit syntaxí a odlišností některých funkcí, lze říci, že téměř každý nabízí výsledné funkce:

* Ukládání hodnot do proměnných
* Zanořování selektorů
* Skládání více souborů dohromady
* Mixiny (skupiny vlastností s parametry, které lze následně používat v rámci celého kódu)
* Dědění tříd a vlastností
* Další funkce (práce s barvami, operátory atd.)

Lze namítnout, že pro některé z těchto funkcí existuje i specifikace v CSS, a tak by nebylo nutné používat preprocesor, například proměnné (<https://www.w3.org/TR/css-variables-1/>), skládání souborů, práce s operátory v rámci calc funkcí. Avšak s funkcemi preprocesorů lze daleko lépe pracovat a v některých případech (jako právě u již zmiňovaných proměnných v CSS, které například nefungují v prohlížeči Internet Explorer ad.) fungují po kompilaci na všech prohlížečích. [1], [17]

* + 1. Syntactically awesome style sheets

Preprocesor Syntactically awesome style sheets (SASS) vznikl již roku 2006. Postupem času se přidali další vývojáři – SASS je totiž vyvíjeno jako open-source. Jeho součástí jsou všechny funkce jmenované v kapitole 2.2. SASS se samozřejmě neustále vyvíjí, největší změnou a novinkou, kterou však lze zmínit je vyvinutí jeho syntaxe. Na počátku mělo SASS imperativní (tj. odpovídající více běžným programovacím jazykům) syntaxi. Ta je samotná dnes nazývaná jako SASS (stejně jako celý preprocesor). Stále ji lze používat, avšak primárně doporučovaná již není a většina moderních knihoven v ní již není dále psána. Syntaxe SASS vypadá takto:

.trida

color: #fff

width: 200px

height: 200px

Kód 8 - Ukázka syntaxe SASS preprocesoru SASS [autor]

S verzí 3 pak nabídlo SASS (preprocesor) i syntaxi zvanou SCSS. Pokud, jako ve výše uvedené ukázce zápisu nepoužijeme žádné funkce preprocesorů, vypadá pak jako normální CSS zápis.

.trida {

color: #fff;

width: 200px;

height: 200px;

}

Kód 9 - Ukázka syntaxe SCSS preprocesoru SASS [autor]

V zásadě je ale jedno, jaká syntaxe je použita, jelikož se tím nijak nemění možnost funkcí. Pro tvorbu knihovny autor použil syntaxi SCSS. [17]

* 1. JavaScript

JavaScript je jednoduchý skriptovací jazyk, jehož syntaxe vychází z jazyků C a Java. Jeho kód se spouští v prohlížeči uživatele a umožňuje s webovou stránkou manipulovat a reagovat na jeho akce. V rámci vytvořené knihovny je použit zejména kvůli manipulaci se třídami elementů a pro vytvoření dynamických komponent knihovny, např. carouselu či modal okna. Pro zjednodušení je použita knihovna jQuery. [10]

* + 1. jQuery

jQuery je javascriptová knihovna, která usnadňuje práci s HTML dokumentem. S její pomocí lze mnoha věcí, které by normálně psal vývojář v čistém JavaScriptu, dosáhnout daleko jednodušeji a přehledněji. Aktuální verze je 3.3.1, ta je v rámci práce taktéž použita, avšak v tzv. slim build, tj. odlehčené verzi knihovny, která neobsahuje některé funkce a moduly. [18]

* 1. Nástroje pro zkompilování knihovny

Jelikož knihovna má být k dispozici k použití i v rámci jednoho CSS a jednoho JS souboru (pokud vývojář použije některé z komponent využívající JavaScript/jQuery), je nutné celou knihovnu zkompilovat. Mimo utvoření a sestavení těchto souborů proběhne během kompilace ještě několik operací:

* Kompilace SASS (resp. SCSS) souborů do validního CSS
* K některým problematičtějším vlastnostem CSS se přidají potřebné prefixy pomocí pluginu Autoprefixer (<https://github.com/postcss/autoprefixer>), což tak zlepší kompatibilitu napříč prohlížeči
* Výsledné CSS a JS soubory se minifikují

Kompilace knihovny by mohla být provedena mnoha způsoby, autor v rámci knihovny uvede příklad kompilace v nástroji Gulp -> potřebné nástroje a pluginy budou nainstalovány pomocí balíčkovacího systému Yarn.

* + 1. Yarn

Yarn je balíčkovací systém Node.js (<https://nodejs.org>), který čerpá balíčky a moduly ze stejného rozhraní, jako starší systém NPM (<https://www.npmjs.com>). Balíčky se instalují pomocí příkazů v příkazovém řádků a jejich struktura je také zapsána v souboru package.json. Yarn autor použil namísto NPM zejména proto, že instalace balíčků pomocí něj je rychlejší (ale nabízí i další výhody). [19]

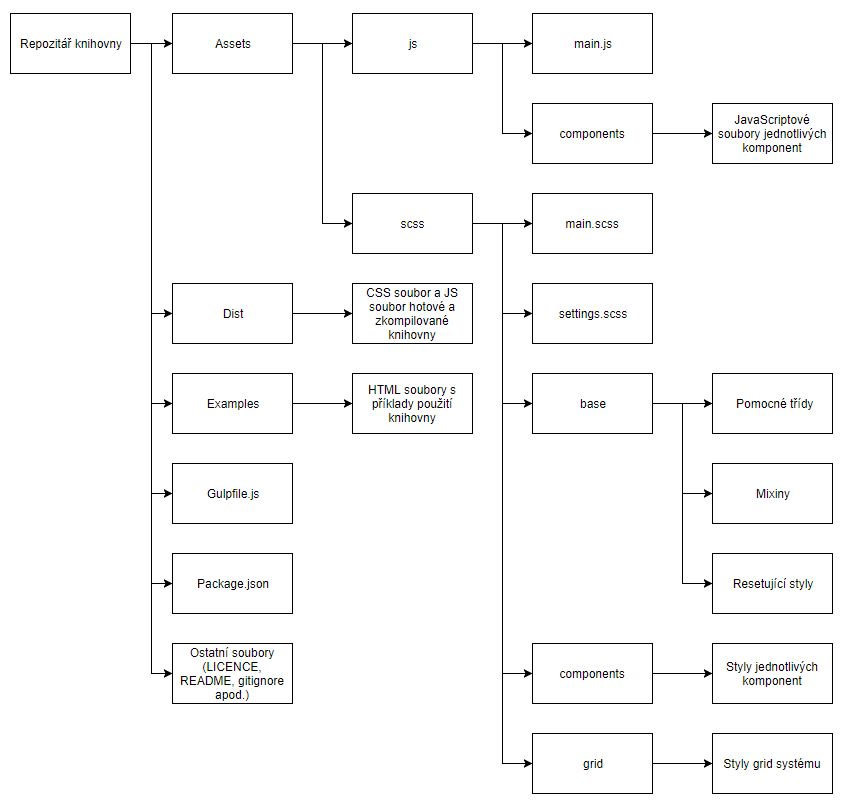
* + 1. Gulp

Gulp je nástroj napsaný v JavaScriptu sloužící pro automatizaci úloh. Je dostupný i jako balíček, takže jej lze nainstalovat pomocí systému Yarn. Úlohy, které chce vývojář automatizovat, se zapisují do souboru gulpfile.js, samotný Gulp se následně spouští z příkazové řádky.

1. Struktura a vize knihovny

Tato kapitola má za úkol popsat plánovanou strukturu knihovny a její vizi.

Repozitář knihovny se bude skládat ze složky assets, kde bude knihovna dostupná ve formátu před kompilací (tj. v preprocesoru SASS a v neminifikovaném JavaScriptu), složky dist se zkompilovaným CSS a JS souborem samotné knihovny a ze složky examples, která bude obsahovat HTML soubory se příklady použití jednotlivých částí knihovny. Mimo zmíněných složek jsou důležité ještě soubory gulpfile.js a package.json, kvůli samotné kompilaci knihovny. Podrobnější strukturu zobrazuje následující schéma:



Obrázek 12 - Schéma struktury knihovny [autor]

Co na schématu najít nelze, je výčet komponent, které knihovna bude obsahovat. To lze najít v tabulce níže. U jednotlivých komponent pak je ještě zmíněno, zdali plánují vyžívat javascriptovou část knihovny.

|  |  |
| --- | --- |
| Komponenty | Využívá javascriptovou část knihovny |
| Textové prvky | Ne |
| Responzivní obrázky, videa a iframe prvky | Ne |
| Karty | Ne |
| Horizontální menu | Ano |
| Vertikální menu | Ano |
| Carousel | Ano |
| Patička | Ne |
| Vizuály | Ne |
| Formulářové prvky | Ne |
| Drobečková navigace | Ne |
| Stránkování | Ne |
| Progress bar | Ne |
| Modal okno | Ano |
| Rozklikávatelné záložky | Ano |

Tabulka 1 - výčet plánovaných komponent [autor]

Z hlediska kompatibility se autor bude zabývat prohlížeči, které využívá více než 0,5 % uživatelů. Jedná se tak o verze desktopových prohlížečů Internet Explorer 11, Edge 16+, Mozilla Firefox 57+, Google Chrome 49+, Safari 11+ a verze mobilních prohlížečů Safari 10.2+, Opera Mini, Android Browser 4.4+, Google Chrome 64+, UC Browser for Android 11.8+ a Samsung Internet 4+. Knihovna samozřejmě může fungovat i na jiných prohlížečích, nicméně autor práce to nebude testovat. Je také možné, že některé vlastnosti nebudou fungovat ve všech prohlížečích, to však autor případně zmíní v dokumentaci knihovny a navrhne alternativní řešení.

Sama knihovna bude vyvíjena pod názvem *skar-is* (zkratka pro Skara Interface Solution) a bude umístěna na adrese <https://github.com/skaryys/skar-is>. Knihovna tak bude dostupná open-source. Vzhledem k tomu, že autor k ní zamýšlí napsat i dokumentaci, ve které by měly být obsáhlé HTML příklady, bude taková dokumentace dostupná na zvláštní webové stránce na doméně <http://skaris.skaramart.in>. Knihovna tak bude dostupná ze tří zdrojů – této domény, Github repozitáře a taktéž jej bude možno nainstalovat pomocí balíčkovacích systémů Yarn a NPM.

Realizace knihovny

TODO

realizace, kompilace, použití + dokumentace

Vytvoření modelových stránek pro demonstraci možností implementované knihovny

TODO

Závěr

TODO

TODO seznamy obrázků, tabulek, příloh

Terminologický slovník

|  |  |
| --- | --- |
| Termín | Význam (zdroj) |
| Autoprefixer | Nástroj pro automatické přidání prefixů pro potřebné vlastnosti CSS. Přidávání prefixů probíhá proto, aby problematické vlastnosti CSS fungovaly i na požadovaných prohlížečích, které vývojář nastaví v nastavení autoprefixeru. |
| Breakpoint | Bod rozlišení, ve kterém se změní chování či styly určitých prvků na stránce. |
| Carousel | Komponenta používaná na webových stránkách, ve které se mění obsah ať již automaticky či pomocí ovládacích prvků. (Obrázek 4) |
| CSS | Cascading Style Sheets – jazyk ovlivňující vlastnosti zobrazení prvků na webových stránkách. Jeho hlavním smyslem je oddělení vzhledových vlastností od obsahu jednotlivých prvků. [9] |
| CSS Grid | Modul CSS, sloužící pro jednoduché vytváření komplexních layoutů celé stránky. [15] |
| CSS knihovna | CSS knihovna je soubor tříd, pravidel a vlastností CSS které mají za úkol pomoct vývojáři při vývoji grafického rozhraní, určitých komponent a jiných řešení. Často pro své fungování využívá i JavaScript. Větší CSS knihovny bývají též často nazývány CSS frameworky. |
| CSS preprocesor | CSS preprocesory jsou jazyky postavené nad CSS, které je následně nutné zkompilovat právě do CSS souborů. Využívány jsou proto, že na rozdíl od jazyka CSS nabízejí více funkcí a zjednodušují díky tomu práci. [1] |
| Drobečková navigace | Anglicky nazývána breadcrumbs, je navigace znázorňující hierarchii webu. Obsahuje cestu přes nadřazené sekce k sekci momentálně zobrazené. |
| Fixní pozicování | Pokud je prvek pozicován fixně, znamená to, že zůstává vždy na stejném místě v okně prohlížeče. |
| Flexbox | Vlastnosti CSS, které určují rodičovskému prvku (Flex container, flex kontejner), jak velké místo v něm budou zaujímat a jak se budou zarovnávat prvky obsažené (Flex items, flex položky). Je tak možné jednoduše měnit zarovnávání prvků do řádku či sloupce, určovat, jak velký bude určitý Flex item oproti ostatním, jak se budou moci měnit flexibilně jeho rozměry, v jakém pořadí prvky řadit apod. [1] |
| Float | Vlastnost CSS pro obtékání, často se používá i pro zarovnávání prvků vedle sebe. |
| Font-family | CSS vlastnost definující jaký font se použije pro vykreslení písma. |
| Grid systém | Struktura, podle které jsou pozicovány jednotlivé prvky a lze díky tomu tak snadno řadit prvky do řádků či sloupců. Pokud se jedná o responzivní systém, manipuluje s umístěním těchto prvků dle okna uživatelova prohlížeče (či zařízení). V případě, že se prvky řadí vedle sebe, mluvíme o horizontálním grid systému, pokud pod sebe, mluvíme o vertikálním grid systému. [2] |
| HTML | HyperText Markup Language – značkovací jazyk využívající se pro tvorbu webových stránek, udává strukturu stránky. Dále s ním lze manipulovat pomocí jazyku JavaScript či udávat prvkům vzhled pomocí jazyka CSS. [9] |
| JavaScript | Jednoduchý skriptovací jazyk, jehož kód se spouští ve webovém prohlížeči a umožňuje s webovou stránkou dále manipulovat a reagovat na akce uživatele. [9] |
| jQuery | Javascriptová knihovna, která usnadňuje práci s HTML dokumentem a zjednodušuje zápis některých javascriptových funkcí. [17] |
| Layout stránky | Rozložení stránky do několika částí s různým významem (např. hlavička, menu, obsahová část, patička, postranní panel apod.). |
| Margin | CSS vlastnost určující šířku vnějšího okraje prvku. |
| Media Queries | CSS vlastnosti Media Queries udávají v rámci jazyka podmínky, které umožní aplikovat určité CSS vlastnosti v určitých situacích. Nejčastěji se používají pro definování stylů pro určité rozlišení displeje. [13] |
| Mobile-first | Mobile-first je metoda tvorby (či návrhu) webových stránek, kde se primárně stránka vytvoří pro mobilní zařízení a následně se upraví pro uživatele zařízení s větší obrazovkou a desktopů. Přesně opačný přístup je pak nazýván *desktop-first*. |
| Modal okno | Komponenta používaná na webových stránkách, jedná se o okno, které se uživateli otevře v popředí stránky. Často se zbytek stránky překryje průhlednou barvou. (Obrázek 5) |
| Open-source software | Open-source software je software jehož zdrojový kód mlže kdokoliv používat, upravovat či vylepšovat. [7] |
| Padding | CSS vlastnost určující vnitřní okraj prvku. |
| Pixel, px | Pixel přestavuje jeden svítící bod na monitoru či jeden bod v obrázku a tisku. Pixel nemá pevné rozměry, odvozuje se podle rozlišení. |
| Rem | Jednotka, která se odvozuje od velikosti písma prvku <html>. Velikost písma v tomto prvku se rovná 1 rem. |
| Responzivita | Responzivní web je takový, který se správně zobrazuje a přizpůsobuje zařízením, které uživatel používá. |
| SASS | Syntactically awesome style sheets, jeden z nejznámějších a nejpoužívanějších CSS preprocesorů. Jako SASS může být nazývána i jedna z jeho syntaxí. [17] |
| SCSS | Jedna ze syntaxí preprocesoru SASS, velice podobná zápisu jazyka CSS. [17] |
| Systémové fonty | Nativní fonty operačních systémů, které jsou v nich automaticky předinstalované. Vývojář se tak může při jejich použití spolehnout, že se uživateli na daném operačním systému načtou. |
| Yarn | Yarn je balíčkovací systém Node.js, který stejně jako známější a starší NPM čerpá balíčky a moduly z rozhraní dostupného na adrese [www.npmjs.org](http://www.npmjs.org). Balíčky se instalují pomocí příkazů v příkazovém řádků a jejich struktura je také zapsána v souboru package.json. [19] |

Seznam citací (TODO po dopsání seřadit podle abecedy)

1. Vzhůru do CSS3. Martin Michálek, 2015. ISBN 978-80-260-8440-2.
2. AMRAN, Adam. Úvod do grid systémů. In: Meebio [online]. Meebio, 01.11.2011 [cit. 2018-01-26]. Dostupné z: <http://blog.meebio.cz/clanek/158/uvod-do-grid-systemu/>
3. Bootstrap · The most popular HTML, CSS, and JS library in the world. [online]. Mark Otto, Jacob Thornton, and Bootstrap contributors [cit. 2018-01-26]. Dostupné z: <https://getbootstrap.com>
4. Foundation [online]. Campbell, California: ZURB [cit. 2018-01-26]. Dostupné z: <https://foundation.zurb.com/>
5. *Pure* [online]. 2018 [cit. 2018-02-17]. Dostupné z: <https://purecss.io/>
6. *Bulma: a modern CSS framework based on Flexbox* [online]. 2018 [cit. 2018-02-17]. Dostupné z: <https://bulma.io/>
7. What is open source? In: *Opensource.com*[online]. Red Hat [cit. 2018-01-27]. Dostupné z: <https://opensource.com/resources/what-open-source>
8. WICHARY, Marcin. Using UI System Fonts In Web Design: A Quick Practical Guide. In: *Smashing magazine* [online]. Smashing magazine, 13 November 2015 [cit. 2018-02-18]. Dostupné z: <https://www.smashingmagazine.com/2015/11/using-system-ui-fonts-practical-guide/>
9. *W3Schools Online Web Tutorials* [online]. W3Schools, 2018 [cit. 2018-02-20]. Dostupné z: <https://www.w3schools.com/>
10. SANDEEP, Panda, Tiffany B. BROWN a Kerry BUTTERS. *HTML5 Okamžitě*. Computer Press, 2015. ISBN 978-80-251-4355-1.
11. УСАЧЕВ, Максим a fantasai. Evolution of CSS Layout: 1990s to the Future. In: *Fantasai: home* [online]. 2012 [cit. 2018-01-31]. Dostupné z: <http://fantasai.inkedblade.net/weblog/2012/css-layout-evolution/>
12. SCHAFER, Steven M. *HTML, XHTML a CSS: bible [pro tvorbu WWW stránek] : 4. vydání*. Praha: Grada, 2009. Průvodce (Grada). ISBN 978-80-247-2850-6.
13. BŘÍZA, Petr. Tvorba layoutu webu – teoretický úvod. In: *Interval.cz* [online]. 2004 [cit. 2018-02-01]. Dostupné z: <https://www.interval.cz/clanky/tvorba-layoutu-webu-teoreticky-uvod/>
14. MICHÁLEK, Martin. *Vzhůru do (responzivního) webdesignu*. Verze 1.1. Praha: vlastním nákladem autora, 2017. ISBN 978-80-88253-00-6.
15. ANDREW, Rachel. *CSS3 Layout Modules* [online]. 2nd edition. United Kingdom: edgeofmyseat.com, 2014 [cit. 2018-02-07].
16. CSS Grid Layout. *W3schools.com* [online]. [cit. 2018-02-07]. Dostupné z: <https://www.w3schools.com/css/css_grid.asp>
17. *Sass: Syntactically Awesome Style Sheets* [online]. Hampton Catlin, Natalie Weizenbaum, Chris Eppstein, and numerous contributors, 2017 [cit. 2018-02-07]. Dostupné z: <https://sass-lang.com/>
18. *JQuery* [online]. The jQuery Foundation, 2018 [cit. 2018-02-08]. Dostupné z: <http://jquery.com/>
19. *Yarn* [online]. [cit. 2018-02-09]. Dostupné z: <https://yarnpkg.com>
20. *Can I use... Support tables for HTML5, CSS3, etc* [online]. [cit. 2018-02-25]. Dostupné z: <https://caniuse.com>

TODO přílohy?