

Sem vložte zadání Vaší práce.

ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
KATEDRA SOFTWAREVÉHO INŽENÝRSTVÍ



Bakalářská práce

Analýza CSS preprocesorů a frameworků usnadňujících stavbu webového rozhraní

Pavlína Ostrá

Vedoucí práce: Ing. Jiří Pavelka

16. května 2013

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval samostatně a že jsem uvedl veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů, zejména skutečnost, že České vysoké učení technické v Praze má právo na uzavření licenční smlouvy o užití této práce jako školního díla podle § 60 odst. 1 autorského zákona.

V Praze dne 16. května 2013

.....

České vysoké učení technické v Praze
Fakulta informačních technologií

© 2013 Pavlína Ostrá. Všechna práva vyhrazena.

Tato práce vznikla jako školní dílo na Českém vysokém učení technickém v Praze, Fakultě informačních technologií. Práce je chráněna právními předpisy a mezinárodními úmluvami o právu autorském a právech souvisejících s právem autorským. K jejímu užití, s výjimkou bezúplatných zákonných licencí, je nezbytný souhlas autora.

Odkaz na tuto práci

Ostrá, Pavlína. *Analýza CSS preprocesorů a frameworků usnadňujících stavbu webového rozhraní*. Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2013.

Abstract

The bachelor's thesis is concerned with usage of CSS frameworks and preprocessors on production of web interface. It introduces how to work a graphical environment of web site with useful tools and offers a comparison of them in terms of usability. The work describes an origin of a component in frameworks in the basis of retrieved knowledges. In the last part it presents the integration of preprocessors into existing Content Management System.

Keywords CSS, framework, preprocesor, web, rozhraní

Abstrakt

Bakalářská práce se zabývá využitím CSS frameworků a CSS preprocesorů při tvorbě webového rozhraní. Vývojářům představuje, jak zpracovat grafické prostředí webové stránky pomocí užitečných nástrojů, jež frameworky a preprocesory obsahují a nabízí jejich srovnání na základě použitelnosti pro jednotlivé druhy webových rozhraní. Práce popisuje vytvoření komponenty ve frameworkcích na základě získaných poznatků při testování. Na závěr prezentuje integraci preprocesorů do daného redakčního systému.

Klíčová slova CSS, framework, preprocessor, web, interface

Obsah

Úvod	1
1 Vymezení pojmů	3
1.1 CSS framework	3
1.2 CSS preprocesor	4
2 Motivace	5
2.1 Současný stav problematiky	5
2.2 Problém volby	6
3 Kritéria a hlediska testování	7
3.1 Layout	7
3.2 Responzivní web design	10
3.3 Typografie	12
3.4 Ovládací prvky a formuláře	13
3.5 Soulad s použitelností webu	14
4 Testování frameworků	15
4.1 Rozdělení frameworků podle zaměření	15
4.2 Použitelnost CSS frameworků v praxi	16
4.3 Návrh a realizace komponenty do CSS frameworků	28
5 CSS preprocesory	31
5.1 Úvod do problematiky preprocesorů	31
5.2 Kompilace a použití	34
5.3 Hodnocení CSS preprocesorů	35
5.4 Integrace CSS preprocesorů do CMS systému	38

Závěr	43
Literatura	45
A Obrázky	47
B Seznam použitých zkratk	57
C Obsah přiloženého CD	59

Seznam obrázků

3.1	Ukázka fixního layoutu[8]	9
3.2	Ukázka fluidního layoutu [8]	9
3.3	Výsledky výzkumu o přístupu na internet z různých zařízení[17]	11
3.4	Podpora media queries	12
4.1	Sloupce typu „123“[6]	19
4.2	Alloy UI komponenta	23
4.3	Myšlenka účaří	24
4.4	Neuplatněná myšlenka účaří	25
5.1	Ukázka upozornění preprocesoru LESS	36
5.2	Ukázka upozornění preprocesoru Stylus	36
5.3	Ukázka upozornění preprocesoru Sass	36
5.4	Interaktivní mód preprocesoru Stylus	38
A.1	Nedokonalá typografie u frameworku YAML	47
A.2	Responzivní web pomocí frameworku YAML	48
A.3	Poznátky týkající se frameworku YAML	49
A.4	Ukázka použitých komponent Bootstrapu	50
A.5	Flexibilní obrázek ve frameworku Bootstrap	51
A.6	Responzivita frameworku Bootstrap	52
A.7	Bootstrap: Navigace v nákupním košíku podle předlohy Alza.cz	53
A.8	Ukázka zobrazení mřížky na stránce v Baseline	53
A.9	Baseline: Rozhozená koncepce účaří	53
A.10	Předloha z RollingStone.com pro vytvoření 3 sloupců v jednom	54
A.11	Využití 960 Grid System	54
A.12	Horizontální banner	55
A.13	Komponenta: Reklamní bannery fixně podél layoutu vlevo a vpravo.	55

A.14 Ukázka responzivní reklamy	56
---	----

Seznam tabulek

3.1	Významné (hraniční) šířky	8
4.1	Hodnocení YAMLu	20
4.2	Hodnocení Bootstrapu	22
4.3	Hodnocení Baseline	26
5.1	Hodnocení preprocesorů	38

Úvod

Bakalářská práce má za cíl informovat čtenáře o pokročilých technologiích v oblasti kódování webového rozhraní. Téma reaguje na její aktuální situaci a podrobně se zabývá souvisejícími problémy. Vzniklo za účelem ujasnění si již získaných zkušeností a jejich dalšího rozvoje.

Práce by měla začínající vývojáře nasměrovat na cestu, na níž se naučí využívat možností nových nástrojů tak, aby jejich činnost byla rychlá a efektivní a výsledek vynaloženého úsilí byl kvalitní. Pokročilé naopak upozorňuje, na které aspekty při využívání moderních technologií by si měli dát pozor a jak z nich vytěžit maximum. Obě skupiny čtenářů by však měli mít minimálně nějaké povědomí o tvorbě webových stránek a aplikací, především pak o problematice HTML a CSS.

Čtenářům jsou předloženy výsledky reálného využití CSS frameworků a preprocesorů v praxi a tlumočeny hodnotné poznatky. Čtenáři zjistí, které frameworky je vhodné využít pro tvorbu velkých projektů, malých osobních stránek nebo zda se vyplatí použít specializovaný framework k jednomu konkrétnímu účelu. Potenciální či aktivní uživatelé frameworků se dozvědí, jak složité je si vybraný framework upravit podle svých potřeb, pokud si chtějí vytvořit vlastní komponentu.

Poslední část práce je zaměřena na vývojářskou obec CSS preprocesorů. Preprocesory jsou porovnány z hlediska syntaxe, způsobu nasazení a kompilace. Práce ukazuje, že preprocesory se využívají nejen při vývoji na svém vlastním počítači, ale lze s nimi pracovat i v samotných aplikacích. Na závěr je předvedeno řešení integrace preprocesoru do konkrétního redakčního systému.

Vymezení pojmů

K celistvému pochopení tématu je nezbytné vysvětlit určité termíny tak, aby se čtenář v tématu neztrácel. Tyto klíčové pojmy začínajícím vývojářům nemusí být zcela jasné.

1.1 CSS framework

Framework obecně slouží jako konceptuální struktura pro budování něčeho užitečného. Jinými slovy se jedná o univerzální základ, na němž lze stavět složitější prvky nebo vyíjet svůj další vlastní framework.

CSS¹ frameworkem se dnes rozumí knihovna CSS souborů, které se používají k vývoji webových stránek založených na HTML² a CSS. CSS framework obvykle poskytuje CSS styly a vlastnosti pro:

- typografii³,
- layout⁴ – typicky v tzv. mřížkovém systému,
- resetování vlastností prohlížečů⁵.

Kodér webových stránek tak nemusí od samého základu vytvářet webové rozhraní, jednoduše využije standardní styly např. pro zaoblené rohy objektů.

¹CSS (Cascading Style Sheets). Jazyk navržený W3C organizací představuje jednoduchý mechanismus přidávání stylů (např. fontů, barev, mezer) do webových dokumentů.

²HTML (HyperText Markup Language) Značkový jazyk používaný pro tvorbu webových stránek.

³Obor zabývající se písmem, jeho výběrem, použitím a sazbou.

⁴Označení pro schéma či rozmístění objektů, v případě webových stránek jde o prvky na stránce.

1.2 CSS preprocesor

Jedná se o dynamický jazyk DSL⁵. Umožňuje dynamické programování v samotném CSS. DSL se blíží více programování, takže programátoři toto ulehčení velice vítají.

Výhoda dynamického jazyka tkví v přínosu objektivně orientovaného zápisu kódu (kódu rozděleného na funkce, s využitím proměnných, dokonce i dědičnosti). Lze používat také matematické binární operace (sčítání, odčítání, násobení, dělení) mezi různými jednotkami i v šestnáctkové soustavě a lze používat i uzávorkování.

Př.: Kód napsaný v LESS⁶ využívající proměnné, zanořování, barevné funkce a matematické operace.

```
@the-border: 1px;
@base-color: #111;
@red: #842210;

#header {
  color: (@base-color * 3);
  border-right: (@the-border * 2);
  img {
    border: 1px solid black;
  }
}
#footer {
  color: (@base-color + #003300);
  border-color: desaturate(@red, 10%);
}
```

⁵DSL (Dynamic Stylesheet language) Dynamický jazyk.

⁶Webová stránka preprocesoru LESS <http://lesscss.org/>

Motivace

Produkce internetových stránek rapidně stoupá a webové technologie se vyvíjejí každým dnem. Díky tomu vzniká potřeba co nejrychleji a nejkvalitněji vytvářet uživatelsky přívětivá a přístupná webová rozhraní. S ohledem na existenci rozmanitých prohlížečů a zařízení není stylování stránek v dnešní době snadné. Díky těmto zařízením a také díky růstu četnosti uživatelů se zároveň na tvorbu webu kladou čím dál větší nároky.

Stylování webového rozhraní je neodmyslitelným denním chlebem kódérů a webdesignerů (dále jen vývojáři). Jedná o složitý vývoj, pokud vše vývojář musí vytvářet od samého počátku. Přitom často může jít o opakovanou činnost. Vývojáři se nemohou nadále zdržovat sestavováním layoutů od samých základů. Proto podobně jako v programovacích jazycích vznikají kvůli opakovaným činnostem frameworky pro stylování stránek. Ty vývoj značně usnadňují a urychlují a tak mohou své úsilí a svůj čas věnovat opravdu náročným nebo nevšedním problémům.

2.1 Současný stav problematiky

Účelem práce je srovnání CSS frameworků na hlubší úrovni než jednoduché tabulky na stránkách Wikipedie. V tabulkách tohoto typu je shrnuto pouze, co framework obsahuje či ne. Další tabulky ukazují, který framework funguje ve kterém prohlížeči, zdali mají podporu nějakého z preprocesorů a pod jakou jsou licenci⁷. Také si lze najít, jaké nabízejí frameworky komponenty. Tyto tabulky bezpochyby pomůžou utřídit si informace k tomu, aby výběr byl zúžen. Např. vyžaduje-li zadavatel podporu prohlížeče IE⁸ ve

⁷Ke shlédnutí např. na <http://usablica.github.com/front-end-frameworks/compare.html>.

⁸IE (Internet Explorer) Internetový prohlížeč.

verzi 6, má vývojář na výběr ze dvou frameworků - YAML⁹ a Elasticss.

Výhodou tabulkového srovnání je rychlá orientace na poli frameworků. Bohužel už nenabídnou srovnání, jak frameworky obstávají v konkrétních případech. Během vývoje webového rozhraní tak vývojář zjistí, že frameworku chybí podstatné součásti, které by sám očekával. Nebo mohou obsahovat chyby či nedokonalá řešení.

Existují jisté zvyky a způsoby, podle kterých se uživatelské rozhraní navrhuje v souvislosti s použitelností. Pak teprve přichází na řadu stylování samotných grafických prvků, jež dávají stránkám origiální vzhled a díky nimž jsou jedinečné. Vývojář, pokud chce používat CSS framework nebo preprocesor, potřebuje znát, co je pro jeho konkrétní případ nejvíce užitečné. Jiné funkce bude upřednostňovat vývojář např. pro e-shop a jiné zase pro osobní webovou stránku.

2.2 Problém volby

V poslední době se působnost CSS frameworků rozrostla natolik, že není jednoduché si vybrat ten správný pro své potřeby. Zároveň vývojář nezná úskalí jednotlivých a neví, co od nich může očekávat. S tím souvisí i použití CSS preprocesorů, díky nimž získává kódování stylů nový rozměr.

Volba správného frameworku bude záviset na konkrétních faktorech (podpora responzivního designu, ovládacích prvků, multimediálního obsahu, ...). Ve výsledku ušetří mnoho času. Lze upozorovat, že určité webové aplikace mají podobně řešený layout. Zaměření tedy padne na výběr frameworku, který konkrétní typ zvládne. Podpora širšího spektra prvků se zvolí, pokud aplikace bude obsahovat příliš mnoho funkcí. Vývojář se vyhne zbytečným překážkám a lehce stvoří kvalitní webové rozhraní. V neposlední řadě je důležité zhodnotit, zdali při použití frameworku bude po dlouhém čase CSS kód stále čitelný. Autor se sám s odstupem času v něm musí umět orientovat.

⁹YAML (Yet Another Multicolumn Layout) CSS framework.

Kritéria a hlediska testování

Než dojde k prezentaci výsledků testování samotných frameworků, zde je pár důležitých pravidel, podle kterých je použitelnost frameworků hodnocena. Tato pravidla vycházejí ze zvyků a standardních postupů stavby webového rozhraní a posloužila jako kritéria pro hodnocení.

3.1 Layout

Layout je základnou či stavebním kamenem celého webového rozhraní. Na základě layoutu (obsahujícím nabídku menu, záhlaví, zápatí, obsahovou část a další) se člověk na stránce orientuje. Pokud není layout logicky sestaven a nemá ty správné proporce, web se stává velmi nečitelným a pro uživatele nepohodlným¹⁰.

Nejprve je nutné si uvědomit, pro jaké zařízení je layout navrhován. Už to není pouze monitor o jednom či dvou rozlišeních, ale nepřeberné množství obrazovek s různými rozlišeními (vizte kapitolu 3.2 Responzivní design). V tabulce 3.1 je přehled hraničních šířek, které jsou ve webovém světě významné.

Vzhledem k těmto rozlišením je vhodné layout přizpůsobit. Jinak bude vypadat rozestavění na mobilním zařízení, jinak na tabletu a desktopu. I když existují již dané šablony layoutů¹¹, podle kterých design lze přizpůsobit, vždy je lepší mít nějaký základ, který se naopak bude schopný přizpůsobit designu, jenž byl pro konkrétní účel navrhnut. U kvalitních webových projektů nejprve vzniká layout vzhledem ke konkrétnímu účelu.

¹⁰Více o použitelnost webového rozhraní na <http://www.nngroup.com/articles/usability-101-introduction-to-usability/>.

¹¹Široký výběr lze nalézt na <http://coding.smashingmagazine.com/2007/01/12/free-css-layouts-and-templates/>.

3. KRITÉRIA A HLEDISKA TESTOVÁNÍ

Tabulka 3.1: Významné (hraniční) šířky obrazovek[10]

320px	Malá rozlišení, obrazovka na výšku.
480px	Malá zařízení, obrazovka na šířku.
600px	Menší tablety, Amazon Kindle (600 x 800), obrazovka na výšku.
768px	Tablety, iPad 1 a 2, obrazovka na výšku.
1024px	Notebooky, stolní monitory, tablety při obrazovce na šířku.
1200px	Širokoúhlé zařízení.

3.1.1 Rozdělení layoutů podle chování

Velice důležitým faktorem ve stylování layoutů je volba takového, který bude sloužit konkrétnímu účelu. Dnes je užitečné flexibilní chování k danému zařízení, ale i fixní layouty najdou své využití. Každý z layoutů má své pro a proti. Pro účely bakalářské práce jsou v potaz brány layouty:

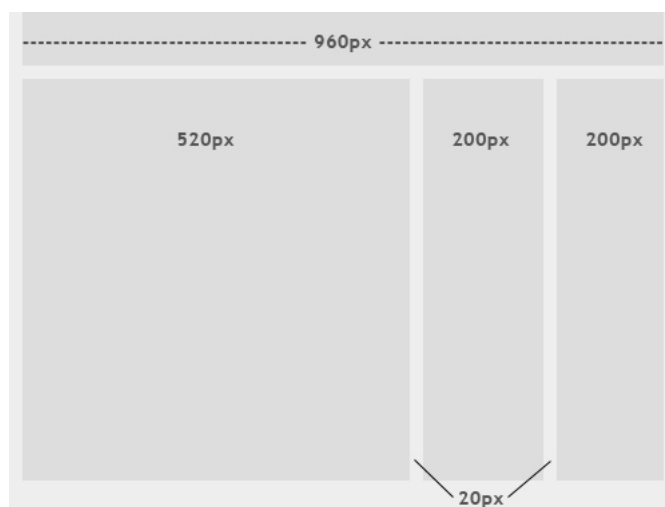
- fixní,
- fluidní,
- adaptabilní.

Fixní layout má danou šířku, která je neměnná a všechny další prvky v něm mohou mít šířku jak fixní tak procentuelní. Výhodou takového layoutu je, že co vidí kodér na svém zařízení, to samé uvidí i uživatelé na dalších jiných zařízeních[8]. Celý layout “sedí” na své pozici a nehýbe se. Ukázka fixního layoutu je na obrázku 3.1. Nejčastější šířka fixního layoutu bývá 960 px, neboť drtivá většina uživatelů vlastní obrazovku s rozlišením 1024 x 768 a vyšší¹² (více o šířce 960px v kapitole 4.2.6 960 Grid System).

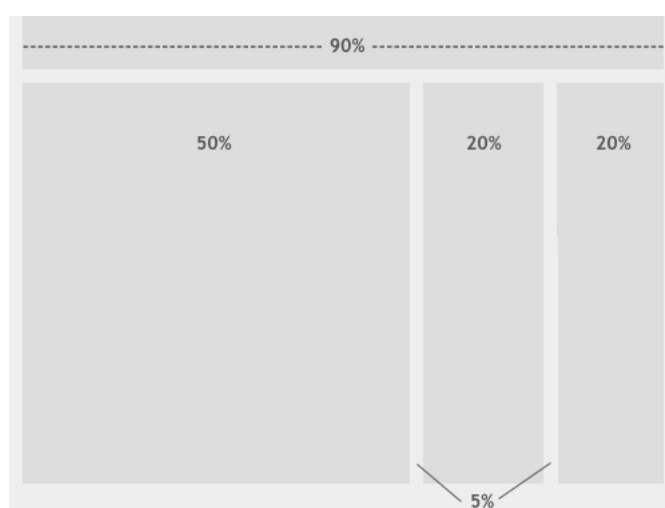
Fluidní layout neboli proměnlivý či tekutý layout (obr. 3.2) se typicky používá k dosažení responzivního designu (vizte kapitola 3.2 Responzivní design). Jeho šířka se přizpůsobuje rozlišení zařízení. Elementy se nastavují v procentech nebo *em* jednotkách. Vnitřní a vnější okraje (*padding* a *margin*) se můžou nastavit fixně v pixelech ale i v procentech. Častěji se využívá fixní šířka, neboť při velkém rozlišení při procentuálním vyjádření vznikají příliš velké mezery a na malém zařízení jsou zase příliš úzké nebo obsah je těžce čitelný[8].

Výhoda fluidního layoutu tkví ve flexibilitě vůči nastavení uživatele včetně velikostí fontů. Kodér má ale menší kontrolu nad tím, co se uživa-

¹²Graf s nejpoužívanějšími rozlišeními na adrese <http://gs.statcounter.com/#resolution-ww-monthly-201204-201303>



Obrázek 3.1: Ukázka fixního layoutu[8]



Obrázek 3.2: Ukázka fluidního layoutu [8]

teli zobrazí konkrétně a v některých rozlišeních může být vzniklý vzhled nežádoucí.

Adaptabilní layout řeší nedostatky fluidního layoutu, které se projevují při příliš nízkém či naopak vysokém rozlišení. Layout se adaptuje na tyto hraniční rozlišení díky využití *media queries* (vizte následující kapitolu 3.2) nebo javascriptu. Při nízkém rozlišení navržený design vypadá

příliš roztahaně a naopak při extrémně velikém se kolem objevuje mnoho nevyužitého místa. Proto se ideálně navrhuje např. tři základní rozložení - pro velmi malé displeje, pro klasické rozlišení obrazovek (např. u notebooků) a pro extrémně velké displeje[7].

3.2 Responzivní web design

S pojmem **responzivní web design** přišel první Ethan Marcotte v roce 2010 ve stejnojmenném článku na A List Apart. Vysvětluje, že při dnešní vysoké produkci všech možných zařízení (chytré telefony, tablety, čtečky knih), odkud lidé přistupují na webové stránky, je nezbytné, aby se web tomuto vývoji přizpůsobil. V dalších letech mobilní zařízení předstihnou desktopy a najednou se nevyplatí vytvářet web zvlášť pro desktop a navíc např. pro iPhone[10]. Jeho předpověď se vyplnila velmi brzy. Dokazuje to statistický výzkum Googlu z roku 2011. Na obrázku 3.3 lze vidět, že lidé přistupují nejčastěji na internet ze svých přenosných zařízení. Jeví se to logicky, neboť uživatelé mají své zařízení stále u sebe, takže mohou na internet přistoupit doslova kdykoliv.

A tady dostává své místo design přizpůsobující se rozlišení zobrazovacího zařízení neboli prohlížeči: *responzivní web design*. Úzce souvisí právě s tvorbou layoutů. Jedná se o kombinaci fluidního a adaptabilního layoutu. Základním pilířem responzivního designu je tzv. media query - specifikace CSS3, díky níž se definují rozdílné styly pro různá rozlišení. Použití v praxi vypadá následovně:

```
@media all and (min-width:500px) { ... }  
@media (min-width:500px) { ... }
```

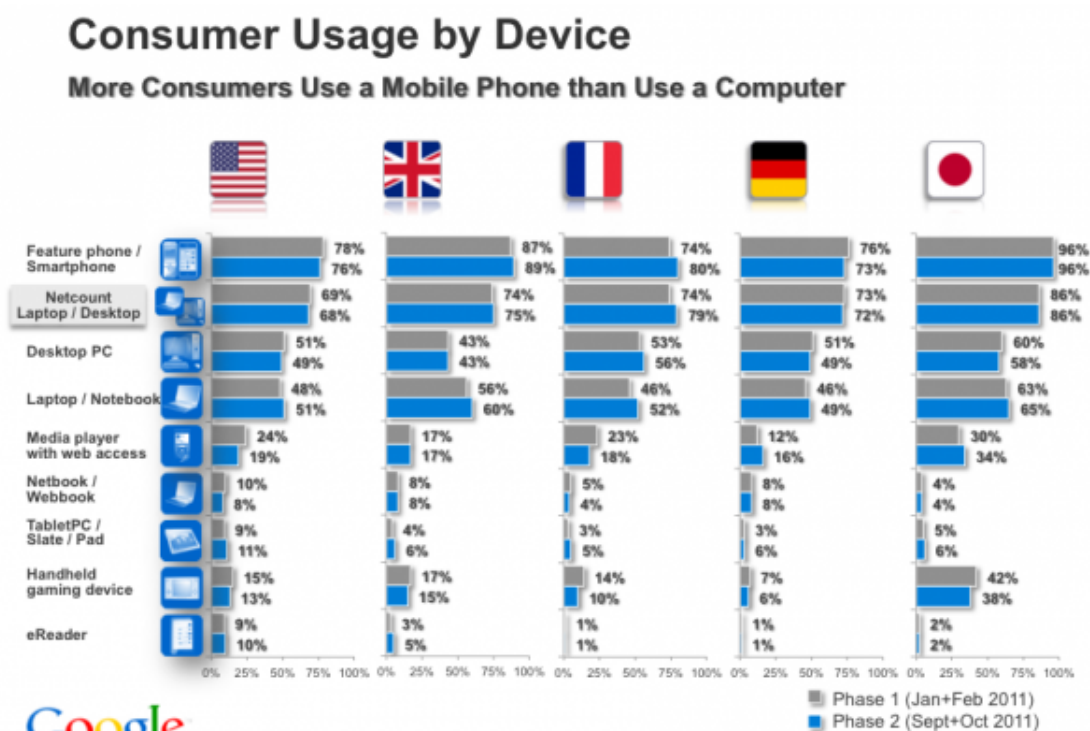
Uvnitř bloku vymezeného složenými závorkami se definují vlastnosti, které se aplikují po splnění podmínky – konkrétně minimální šířky prohlížeče na příslušném zařízení. Styly je možné různě propojovat a strukturovat. Lze toho dosáhnout přímo v HTML dokumentu:

```
<link rel="stylesheet" type="text/css"  
media="screen and (max-device-width: 480px)"  
href="style480.css" />
```

nebo v CSS souborech[20]

```
@import url(style480.css) (min-width:480px);
```

Responzivní design by měl také z velké části být doménou samotných webových grafiků. Ti by měli být schopni navrhnout takový design, aby se při změně šířky vhodně adaptoval. Kodér naopak by měl znát jeho technické úskalí a umět responzivitě reálně aplikovat. K tomu znatelně pomáhají



Obrázek 3.3: Výsledky výzkumu o přístupu na internet z různých zařízení[17]

právě CSS frameworky, zejména pak mřížkové (grid) systémy, které bez zohlednění responzivity jsou prakticky nepoužitelné[17][13].

Otázka také je, zdali dnešní prohlížeče responzivní design vůbec podporují - tedy techniku CSS3 *media queries*. Podle obrázku 3.4 je to překvapivě vysoké procento a v podstatě jediné prohlížeče, které působí problém, jsou staré verze IE[18].

3. KRITÉRIA A HLEDISKA TESTOVÁNÍ

Compatibility table for support of CSS3 Media Queries in desktop and mobile browsers.

Legend: ■ = Supported ■ = Not supported ■ = Partially supported ■ = Support unknown

CSS3 Media Queries - Recommendation

Method of applying styles based on media information. Includes things like page and device dimensions

Resources: [IE demo page with information](#) [Media Queries tutorial](#) [Polyfill for IE](#)

Global user stats*:

Support:	85.38%
Partial support:	0.01%
Total:	85.39%

	IE	Firefox	Chrome	Safari	Opera	iOS Safari	Opera Mini	Android Browser	BlackBerry Browser	Opera Mobile	Chrome for Android	Firefox for Android
22 versions back			4.0									
21 versions back			5.0									
20 versions back			6.0									
19 versions back		2.0	7.0									
18 versions back		3.0	8.0									
17 versions back		3.5	9.0									
16 versions back		3.6	10.0									
15 versions back		4.0	11.0									
14 versions back		5.0	12.0									
13 versions back		6.0	13.0									
12 versions back		7.0	14.0									
11 versions back		8.0	15.0									
10 versions back		9.0	16.0		9.0							
9 versions back		10.0	17.0		9.5-9.6							
8 versions back		11.0	18.0		10.0-10.1							
7 versions back		12.0	19.0		10.5							
6 versions back		13.0	20.0		10.6			2.1				
5 versions back	5.5	14.0	21.0	3.1	11.0			2.2		10.0		
4 versions back	6.0	15.0	22.0	3.2	11.1	3.2		2.3		11.0		
3 versions back	7.0	16.0	23.0	4.0	11.5	4.0-4.1		3.0		11.1		
2 versions back	8.0	17.0	24.0	5.0	11.6	4.2-4.3		4.0		11.5		
Previous version	9.0	18.0	25.0	5.1	12.0	5.0-5.1		4.1		12.0		
Current	10.0	19.0	26.0	6.0	12.1	6.0	5.0-7.0	4.2	7.0	12.1	25.0	19.0
Near future		20.0	27.0						10.0			
Farther future		21.0	28.0									

Obrázek 3.4: Podpora media queries

3.3 Typografie

Typografie je neodmyslitelnou složkou celého webdesignu. Struktura textu má na výsledný dojem stejný podíl jako layout. Framework by měl vývojáři práci s typografií usnadnit, ať už má nebo nemá vyloženě grafickou předlohu.

Typografická pravidla, kterých na webu lze dosáhnout, jsou zobrazení prvků jako uvozovky, pomlčky, rozdělovníky, mezery mezi navazujícími odstavci, vertikální mezery mezi nadpisy a odstavci, odsazení prvního řádku. Některá nejsou možná zařídit kaskádovými styly jako např. odstranění jednoznakových předložek z konců řádků. Ty se řeší až na úrovni samotného textu. Zbytek pravidel by však neměl být opomíjen.

Testování klade důraz na správné formátování odstavců a nadpisů. Velikost fontu je možné relativně zvětšovat u sémanticky významných prvků (nadpisů). Ovšem zmenšovat jej v rámci běžných textů pod preference uží-

vatelů nedoporučuje. Plyne to z doporučení W3C¹³ o přístupnosti webu.[21]

Jiný případ představuje řádkování. Řádky by od sebe měly mít nějaký odstup, který také není vhodný definovat fixně. Vůči nastavení uživatele to může být nedostačující nebo přehnané. Kdežto relativní jednotky zajistí formátování řádků vhodné přesně pro konkrétní nastavení.

Mezi jednotlivými odstavci a nadpisy by mělo být volné místo, aby záchytné body byly na první pohled patrné. Ostatní pravidla typografie ovlivní až vydavatel, který bude texty psát¹⁴.

3.4 Ovládací prvky a formuláře

Podpora ovládacích prvků je důležitá z prostého důvodu – málokterá webová stránka se bez nich obejde. Jedná se o prvky, které uživatele nějakým způsobem navigují nebo mohou sami na stránce něco ovládat. Považovány jsou za ně tlačítka, různé druhy navigací, panel nástrojů, kontextová nápověda atd. Jejich stylování je přitom také časově náročná záležitost.

Podpora všech těchto ovládacích prvků může být mnohdy nadstandardní ale pro někoho naopak nedostačující. Technicky je možné spojit více frameworků, pokud vývojář na stránce potřebuje širší škálu komponent. Tomuto účelu může posloužit např. knihovna Alloy UI, která je podrobněji popsána v kapitole 4.2.4. Předpoklad testování je tedy zhodnocení základních ovládacích prvků, které jsou důležité a potřebné téměř při každé stavbě webu:

- navigace (i místní nabídka),
- tabulky,
- seznamy.

Samostatným kritériem jsou formuláře, jež se dají považovat za komponentu nadřazenou některým ovládacím prvkům. Jedná se o textové oblasti (*text area*), tlačítko pro odeslání (*submit*), zaškrtačací pole (*check box*), přepínače (*radio button*), rozbalovací nabídka (*select box*) a případně další. Formuláře jsou nedílnou součástí objevující se na webu a proto se předpokládá jejich podpora v CSS frameworkcích.

¹³W3C (World Wide Web Consortium) Organizace vyvíjející webové standardy.

¹⁴V dnešní době se o některá pravidla starají textové editory zabudované do redakčních systémů – např. CKEditor.

3.5 Soulad s použitelností webu

V poslední řadě nelze opomenout použitelnost a přístupnost webu, jak předkládá W3C konzorcium. Z velké části je to záležitost HTML kódu, ale o některé vlastnosti se zasluhují také kaskádové styly. CSS frameworky by tuto použitelnost neměly porušovat.

CSS frameworky by měly obsahovat, pokud se jimi zabývají, následující vlastnosti:

- zřetelnou a konzistentní navigaci (jednotná myšlenka designu),
- elementy rozlišné barvy by měly svou informaci sdělovat i bez barevného rozlišení,
- možnost zobrazit obsah různými způsoby (např. jednodušší layout) bez ztráty informací nebo struktury.[21]

Testování frameworků

Následující kapitoly rozšiřují jednoduché srovnání o podrobnější poznatky než jsou shrnutí vlastností frameworků na internetu¹⁵. Také rozdělují frameworky do skupin podle jejich zaměření. Kapitoly dále představí:

- Použitelnost jednotlivých CSS frameworků tak aby práce na webovém rozhraní byla:
 - jednoduchá a pohodlná,
 - rychlá a efektivní,
 - důsledná a bez chybných zobrazení.
- Jak si frameworky poradí s jednotlivými kritérii a jak jsou celkově použitelné pro:
 - e-shop (obsahují mnoho ikonek a formulářů, mívají složitý layout),
 - webový portál (vyžadují komplexní řešení, důležitá je typografie).

4.1 Rozdělení frameworků podle zaměření

CSS frameworků se v poslední době objevuje nepřehledné množství. Aby jim vývojáři lépe porozuměli, lze je rozdělit do typů podle jejich specializace:

¹⁵Široké spektrum frameworků a co podporují lze nalézt např. na webu *Social Compare* ze srpna 2012 <http://socialcompare.com/en/comparison/css-grids-and-responsive-frameworks-comparison>.

- mřížkové systémy,
- specializované typografické knihovny,
- komplexní frameworky.

Každou kategorii v této práci zastupuje minimálně jeden CSS framework. Pro komplexní frameworky je výzva shrnout všechny dobré praktiky používané na webu do jednoho celku. Ostatní zaměření frameworků mohou vývojářům velmi dobře posloužit ke konkrétním účelům. Jaké to jsou a které frameworky jim odpovídají včetně zdůvodnění, proč byly zvoleny právě tyto, je uvedeno v následující kapitole.

4.2 Použitelnost CSS frameworků v praxi

CSS frameworky, které jsou testovány v této bakalářské práci, jsou následující:

- Twitter Bootstrap,
- YAML CSS framework,
- 960 Grid System – (specilizovaný na mřížkový systém),
- Baseline – (specializovaný na typografii).

Komplexní framework Twitter Bootstrap představuje zajímavý objekt k testování z důvodu velké oblíbenosti mezi vývojáři. Kdo se alespoň okrajově setkal s kódováním webových stránek, minimálně se o něm doslechl. Přestože žádný framework nedosahuje takové popularity jako Bootstrap, je příhodné srovnat alespoň podobně vypracovaný framework a zjistit, do jaké míry se mu svým charakterem může vyrovnat. Tento předpoklad splnil YAML CSS framework.

Frameworky, které jsou zaměřené na konkrétní oblast, se uplatní v případě, kdy někteří vývojáři mají už své vlastní frameworky šité na míru např. specifickým potřebám ve své firmě. Komplexnost řešení dané problematiky ve frameworku by měla odpovídat vyšší úrovni než dosahují obecné frameworky. Předpokladem u takových frameworků je také menší velikost zdrojových souborů a s tím související menší náročnost při načítání stránky.

Předlohu pro testování frameworků YAML a Bootstrap představuje internetový obchod Alza.cz, který poskytuje dostatek potřebných prvků k otestování a jejich složitější rozestavení. Internetový obchod Alza.cz tak poskytl

výborné prostředí pro zhodnocení, jak se určitý framework dokáže vypořádat s takto komplexní aplikací.

Jako předloha pro typograficky zaměřený Baseline a pro 960 Grid System posloužil internetový hudební zpravodaj RollingStone.com. Na zpravodajských stránkách je dobrá typografie vyžadována více než na e-shopu (několik tisíc uživatelů čte denně články). Zpravodaj RollingStone.com navíc obsahuje rozmanitý layout, který je skvělý k otestování frameworku 960 Grid System.

4.2.1 Postup práce

S použitím konkrétního frameworku je cílem dosáhnout základního funkčního vzhledu zvoleného webu bez jediného zásahu do stylů. Znamená to, že bude vytvořeno jen HTML s třídami, jež framework nabízí. Z toho vyplynou silné a slabé stránky celého frameworku.

Doporučení *Při kódování webových stránek je dobré využít styly, které upozorňují na některé nedostatky kódu. Lze použít již existující nástroj Diagnostic CSS¹⁶. Díky nim vývojář na první pohled uvidí:*

- nevyplněné atributy href, alt a title,
- HTML značky bez obsahu (div, span, li, p, td, th),
- nevyplněné atributy class a id.

Pokud není z těchto vlastností nějaká v pořádku, prvek se na stránce zvýrazní odlišnou barvou nebo se ohraničí. Ukazuje jednoduchý způsob, jak si ohlídat i jiné vlastnosti na stránce. Tento nástroj se ukáže užitečný především v době, kdy nám obsah generuje např. nějaký redakční systém a uživatelé jej publikují pomocí editoru.

4.2.2 Způsob hodnocení frameworků

Na závěr každého testování bude daný framework ohodnocen body podle zvolených kritérií. Hodnocení je shrnuto do tabulky následujícím způsobem:

- 2 body: řešení je ideální a bezchybné,
- 1 bod: kritérium je splněno, ale řešení není ideální nebo bezchybné,

¹⁶Ke stažení na <http://meyerweb.com/eric/tools/css/diagnostics/>.

- 0 bodů: framework neobsahuje požadované kritérium.

Čím větší získá framework ohodnocení, tím lépe splňuje kritéria a je tedy z hlediska testovaných kritérií kvalitnější.

4.2.3 YAML CSS framework

Přestože YAML je zkratkou pro Yet Another Multicolumn Layout, tedy „ještě jiný vícesloupcový layout“, dokáže konkurovat svou propracovaností i frameworkům, které oficiálně nabízejí širší spektrum předpřipravených prvků. Tvůrce YAMLu *Dirk Jesse* si správně uvědomil, že když už uživatel tvoří webové rozhraní, objeví se na něm i formuláře, různé typy navigací a tabulky.

YAML má dobře vyřešenou podporu prohlížeče IE. Veškeré tyto tzv. „IE hacky“¹⁷ ukládá do samostatného souboru. Výhodu to představuje především pro vývojáře, kteří se rozhodnou cílit na skupinu používající moderní prohlížeče a přejít si, aby zůstal web validní podle W3C konzorcia.

Responzivní web design YAML si ve smyslu responzivity vede velmi dobře a celková práce s layoutem je více než pohodlná. Obsahuje jak fluidní, tak adaptabilní layout (obr. A.2). Myšlenka adaptace layoutu je postavena na tzv. progresivní linearizaci. Znamená to, že není nutné vytvářet styly pro každé rozlišení zařízení zvlášť. Stačí určit hraniční šířku, od které layout vypadá nepoužitelně a upravit třídy tvořící mřížku, aby se přizpůsobovaly aktuální šířce okna.

Layout Ve frameworku YAML je možné volně kombinovat sloupcový systém s mřížkovým systémem. Sloupcový systém poskytuje 1-3-2 nastavení. Sloupce 1 a 2 jsou definovány jako plovoucí prvky a třetí představuje statický kontejner (obr. 4.1). Zastánci layoutu s fixní šířkou 960 px (více v kapitole 4.2.6) najdou ve frameworku také styly pro tento systém.

V mřížkovém systému vývojář intuitivně použije správnou třídu pro šířku sloupce, kterou potřebuje, neboť její název vyznačuje určité procento. Pokud si vývojář přeje použít šířku jinou než YAML nabízí, musí si takovou třídu vytvořit ve svém vlastním CSS souboru.

¹⁷Staré verze prohlížeče Internet Explorer jsou známé svým nedodržováním webových standardů. Odlišné zobrazení designu vývojáři řeší vkládáním identifikátorů před styly v CSS souborech, kterým rozumí pouze daná verze IE.

Obrázek 4.1: Sloupce typu „123“^[6]

Typografie Jak si YAML vyhrál s grid a column systémem, o to méně si dal záležet na typografických zásadách. Všimnout si toho můžeme např. u nadpisů. Na obrázku A.1 nadpis přes více řádků postrádá mezi řádky mezeru, aby byl nadpis lépe čitelný. V tomto případě se řádky téměř slévají do sebe. Samozřejmě jde o drobný nedostatek, který lze jednoduše zpravit ve svém vlastním CSS kódem přidáním většího `line-height`.

Když ale nahlédneme do zdrojových souborů, zjistíme, že výška řádků u jednotlivých nadpisů je rozdílná a ve většině případů nedostačující. U nadpisu první úrovně je velikost písma 350 % a výška řádku je pouze 0.857 em. Celkově není jasné, podle jakých pravidel byly velikosti písem a řádků stanoveny, každopádně čitelnosti neprospěly.

Ovládací prvky Dalším nedostatkem je malá flexibilita formulářových prvků. Celkově nelze dosáhnout klasicky používaných formulářových vzhledů, taktéž zaškrtačací pole (*checkboxy*) se správně neuspořádaly (obr. A.3).

YAML poskytuje volby zarovnání, či nastavení obrázků, videí či dalších volitelných prvků i jejich responzivní řešení. V ohledu responzivity se jedná o důležitý faktor, pro který by mělo existovat řešení, pokud je cílem práce vývojářům ulehčit. Za to patří YAMLu velké plus.

S ostatními ovládacími prvky nenastal žádný problém. U tabulek je možné vybírat ze tří stylů, což pro základní stavbu webu naprosto stačí. Dále nabízí navíc styly pro upozornění a chybové hlášky plus další drobné prvky¹⁸.

Potenciál využití S ohledem na různé skupiny vývojářů bylo shledáno, že framework YAML se skvěle hodí pro vývojáře, kteří webové stránky nikdy příliš nekódovali nebo chtějí rychle vytvořit funkční základ pro svou osobní prezentaci či jiný web a nechtějí najímat drahé firmy. V podrobné

¹⁸Všechny si lze prohlédnout v online dokumentaci <http://www.yaml.de/docs/index.html>.

dokumentaci vývojář najde vše, co nabízí s popisem, jak co použít včetně ukázkových HTML kódů.

Zároveň framework může výborně posloužit i větším webovým rozhraním, pokud se člověk chce zaměřit na details sám od začátku. Všechny základní principy použití jsou dostatečně vyřešeny a dá se na nich stavět.

Překážkou v použití může být neexistující minimalizovaný soubor, který by měl obsah všech CSS souborů sjednotit. Dá se to vyřešit sice celkem snadno různými online minifikátory¹⁹, ovšem hotový soubor ve frameworku by tento čas ušetřil. Další možností je použití preprocesoru Sass²⁰ nebo Stylus, které jedním příkazem zkompilují celý adresář a zároveň soubory zminimalizují.

Hodnocení Kritéria hodnocení uvedená v kapitole 3 Kritéria a hlediska testování byla pro názornost rozdělena do více podskupin. Dohromady získal YAML 10 bodů z 12. Obsahoval všechny požadované styly. Ve dvou případech nebylo řešení úplně bezchybné.

Tabulka 4.1: Hodnocení frameworku YAML

Kritérium	Body
Layout	2
Responsivní design	2
Typografie	1
Formuláře	1
Ovládací prvky	2
Media (obrázky, videa,.. i vzhledem k RD)	2
Celkem	10

4.2.4 Twitter Bootstrap

Bootstrap se oproti ostatním frameworkům stal robustním projektem, který si za dobu své krátké existence (necelé dva roky) získal širokou komunitu pomáhající s vývojem. Cílovou skupinou tohoto frameworku se stali weboví vývojáři z důvodu široké škály funkcí (existuje i mnoho forků²¹ a komponent), jež Bootstrap obsahuje.

¹⁹Program, který zminimalizuje zdrojový kód např. do jednoho řádku. Jeden z oblíbených se nachází na adrese <http://refresh-sf.com/yui/>

²⁰Sass (Syntactically Awesome Stylesheets) CSS preprocessor.

²¹Ve vývoji softwaru se forkem označuje případ, kdy vývojář vezme kompletní kód již existujícího programu a začne na něm nezávisle vytvářet jiný program.

Propracovanost frameworku Bootstrap může být občas nevýhodou. Jelikož je v něm vše vyřešeno, vývojáři se nechtějí dále zabývat grafickou stránkou projektu, a tak weby ztrácejí svou jedinečnost. Aby weby byly od sebe rozpoznatelné, dají se na ně aplikovat **témata** na tomto frameworku postavená. Vývojář může vybírat ze zdarma open-source²² témat např. na stránce <http://bootswatch.com>. Mnohem lepší varianta se jeví zakoupit nějaké téma z komerčních na stránce <http://wrapbootstrap.com/>, které tvoří profesionálové a jejich výběr je pak daleko širší.

Bootstrap obsahuje ve výsledku dva nezminimalizované soubory s CSS styly. V prvním souboru jsou uvedeny všechny styly v jednom logickém celku, ve druhém se nacházejí styly pro vše, co souvisí s responzivním designem. Obsahuje tedy i adaptabilní layout i řešení responzivity obrázků, jak je viditelné na obrázku A.5 (pro obrázky existuje také několik dalších stylů).

Zdrojové soubory si lze stáhnout také rozdělené podle společných vlastností v jednotlivých souborech, ale jsou napsány v preprocesoru LESS. CSS rozdělené soubory neobsahuje.

Layout Bootstrap užívá 12 sloupců, kde si vývojář volí mezi fixním a fluidním layoutem. Poskytuje zanořování, takže nenastal případ, kdy by mezi jednotlivými elementy byly nežádoucí nebo chybějící mezery.

Responzivní web design Layout se správně seskupil podle velikosti malého zařízení. Uživatel má nejdříve k dispozici menu, aby se mohl přemístit, kam potřebuje (obr. A.6). Obrázky se automaticky zmenšují už na úrovni, jež zahrnuje responzivní design.

Ovládací prvky Už na elementární úrovni člověk pozná, že se jedná o komplexní framework. Obsahuje grafické rozhraní často pro nezbytné malíčkosti jako je drobečková navigace nebo stránkovač (obr. A.4).

Všechny ostatní prvky, které bylo nutné použít k postavení webového rozhraní Alzy, Bootstrap obsahoval v daleko širším rozpětí, než bylo potřeba. (Bootstrap má podobně komplexní dokumentaci jako YAML²³) Pro oblasti formuláře, ovládací prvky i media nabízel výběr hned mezi několika volbami – např. navigace horizontální, vertikální, rozbalovací, přepí-

²²Obrat open-source se používá v souvislosti s produkty, které mají veřejně dostupný zdrojový kód, podmínky užití se pak řídí danou licencí.

²³Více informací o všech podporovaných prvcích na oficiálním webu <http://twitter.github.io/bootstrap/>.

nací záložky. V kombinaci dvou voleb byla jednoduše vytvořena i navigace v e-shopu (obr. A.7).

Potenciál využití Bootstrap těžko využije webový grafik nebo obyčejný uživatel pro svůj blog. Nebude rozumět syntaxi preprocesoru LESS, v němž je Bootstrap původně nakódován, a v konečném souboru obsahujícím všechny styly pohromadě se bude ztrácet. Na druhou stranu lze s využitím různých komponent bez jediného zásahu do zdrojových souborů s ním postavit celý web.

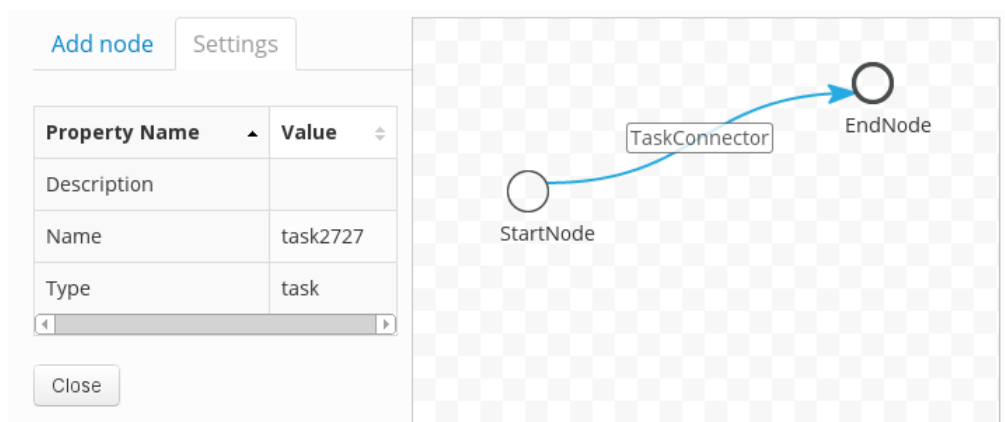
Hodnocení Ve funkčním vzhledu není mezi Bootstrapem a YAMLelem příliš znatelný rozdíl. I YAML je až na zmíněné nedostatky velmi dobře propracovaný. Jediný velký rozdíl mezi těmito frameworky je velikost komunity vyvíjející a používající framework a to především z důvodu odlišné orientace každého frameworku.

Bootstrap se silně snaží o dosažení dokonalosti a není třeba zapírat, že se mu to dobře daří. Ve všech požadovaných kritériích dosahuje nejvyššího hodnocení. Znamená to, že pokud vývojář hledá framework, na který se může spolehnout ve všech důležitých i méně používaných aspektech stránky, Bootstrap je pro něj tou nejlepší variantou.

Tabulka 4.2: Hodnocení frameworku Bootstrap

Kritérium	Body
Layout	2
Responsivní design	2
Typografie	2
Formuláře	2
Ovládací prvky	2
Media (obrázky, videa,.. i vzhledem k RD)	2
Celkem	12

V souvislosti s Bootstrapem stojí za zmínku ještě tzv. Alloy UI knihovna, jenž na něm staví a obsahuje detailní komponenty jako např. hlasování pomocí hvězdiček, progressbar a stromovou strukturu menu. Jedná se ovšem z větší části o javascriptovou knihovnu (obr. 4.2).



Obrázek 4.2: Jednou z nejnovějších komponent v Alloy UI je např. Diagram Builder.

4.2.5 Baseline

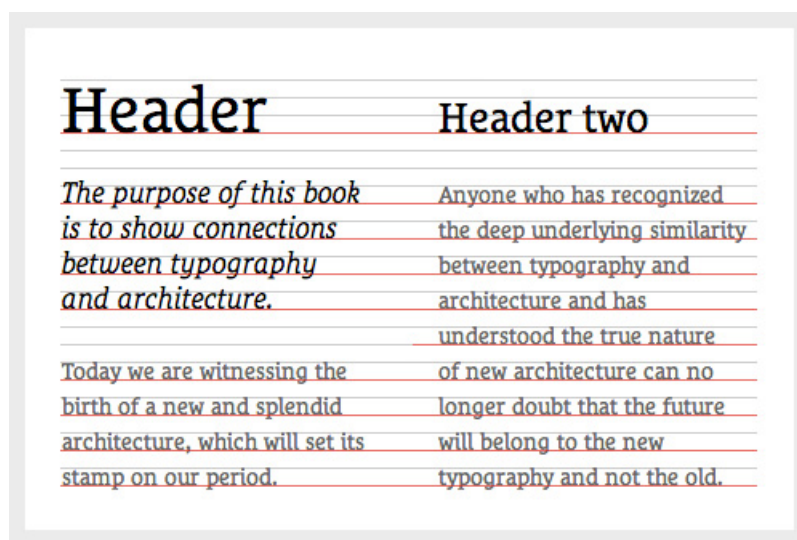
Baseline je framework postavený na myšlence účaří²⁴ v typografii. Jde o jakýsi “vertikální rytmus”, který uživateli v jeho podvědomí napomáhá se na stránce orientovat. Baseline by měl zaručit, že ve všech sloupcích budou všechny velikosti fontů sedět na jednotné lince. Názorně je to předvedeno na obrázcích 4.3 a 4.4.

Myšlenka účaří se ve webovém prostředí uplatňuje [11]. Nutno však zmínit, že jde o složitý vývoj vzhledem ke každému webu a tudíž ji je vhodné použít spíše na menší weby, kde nebude příliš mnoho boxů. Pokud je typografie dobře nakódovaná, nic se nezmění ani při výměně fontu. Chtěl-li by ale někdo jiné proporce písma, musel by vytvořit nový koncept celého Baseline.

K hodnocení specializovaného frameworku jako je tento se musí přistupovat individuálním způsobem. Už na začátku jsme seznámeni s tím, že v rámci tohoto frameworku nejsou zhotoveny žádné navigační prvky, zvláštní ikonky a tlačítka. Jeho hodnota by měla být někde jinde. Přesto obsahuje grid systém, formuláře a tabulky. Ze zvolených kritérií k hodnocení bude ubrána položka responzivního designu.

Layout Baseline je omezující z pohledu stavby layoutu. Při testování je nutné přizpůsobit frameworku výběr webu. Na webu RollingStone.com tedy

²⁴Místo na které umísťujeme písmo např. v sešitě je to linka.



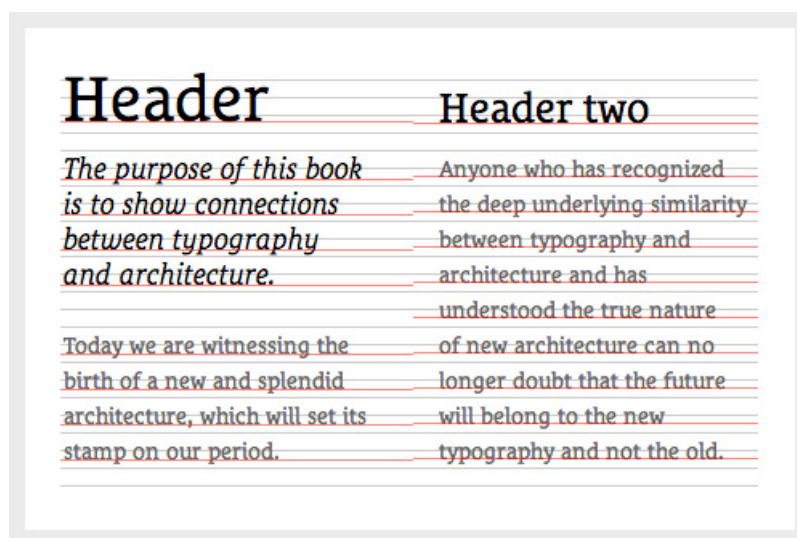
Obrázek 4.3: Všechn text zarovnán na jednu linku[2]

nebyla brána v úvahu úvodní strana, ale klasický dvousloupcový layout. Ten je typický právě pro zpravodajské weby a blogy.

Při tvorbě webu si lze pomocí CSS zobrazit mřížku A.8, aby vývojář viděl a kontroloval si, zdali je koncept účarí funkční a něco jej nenarušuje. Kromě ní obsahuje framework ještě styly pro mobilní zařízení. Přestože nepočítá s responzivním designem, je plusem, že myslí alespoň na mobilní zařízení.

Typografie Problém nastane, když vložíme do stránek obrázky. Styly pro obrázky v Baseline nejsou vůbec nijak předpřipraveny. Je to chybou, neboť rozhodí text tak, že přestane sedat na linky A.9. V tento okamžik přestává framework sloužit účelu, za jakým byl vytvořen. Přidání vlastnosti `float: left` nebo `right` vše napraví. Potenciálnímu vývojáři frameworku, který nemá s kódováním příliš zkušenosti, to nemusí být ihned jasné.

Baseline bohužel zmenšuje základní velikost písma, což odporuje zásadám použitelnosti podle W3C standardu[21]. Typografie pokulhává i v případě nadpisů a odstavců. Pro odstavce používá knižní odsazení a za odstavci nekládá mezery. Odsazení je v pořádku, ale nutno podotknout, že na webu se zvlášť při malé velikosti písma čte text bez mezery mezi odstavci hůře. Navíc u nadpisů 5. a 6. úrovně nejsou mezery vůbec - ani nad ani pod nadpisem. Uživatel bude těžko tyto záchytné body považovat za nadpisy. Když zaměření padne jen na nadpis 6. úrovně, není rozeznatelný od textu



Obrázek 4.4: Text ve druhém sloupci odskakuje[2]

označeného tučně tagem ``. Jsou to nedostatky, jež lze zpravit dodáním vlastního CSS, jenomže tak může být lehce koncepce učaří rozbita.

Ovládací prvky Po zaměření na formulářové prvky nebyly shledány žádné nedostatky. Třídy používající se ke stavbě layoutu na různé šířky sloupců se univerzálně dají použít také na inputy, textarey, tabulky apod. Vývojář si tedy nemusí zapamatovávat hromadu názvů tříd.

Potenciál využití Framework má vysoký potenciál být rozšířenějším mezi vývojáři, kdyby věnoval svou pozornost více mřížkovému systému. Vytvoření více tříd, díky nimž nebude stavba tolik omezující jako nyní, by mu výrazně zvedlo preference.

Baseline najde své využití u vývojářů, kteří nerozumí typografii nebo neumí její pravidla v praxi aplikovat. Výrazně ulehčí práci také lidem, jež mají nějaké zkušenosti s kódováním a nepotřebují vytvářet složité webové rozhraní. Musí si ale dát pozor na to, aby nerozhodili koncept učaří, takže je dobré vědět, co jej ovlivní. Další skupinou lidí pro něž Baseline bude užitečný jsou vývojáři, kteří chtějí použít princip učaří jen s jinými proporcemi písma. Najdou v něm návod, jak správně tuto myšlenku realizovat.

Hodnocení Framework z 10 možných bodů (responzivní design nebrán v úvahu) získal 6. Znamená to, že má v lecčems hodně co dohánět. Autoři frameworku by si také měli ujasnit, na jakou skupinu vývojářů se chtějí zaměřit. Framework je postaven na opravdu dobré filozofii, která by měla být dotažena do nejmenších detailů.

Tabulka 4.3: Hodnocení frameworku Baseline

Kritérium	Body
Layout	1
Responsivní design	-
Typografie	1
Formuláře	2
Ovládací prvky	2
Media (obrázky, videa,.. i v0zhledem k RD)	0
Celkem	6

4.2.6 960 Grid System

Jak už název sám napovídá, jedná se o framework obsahující styly pouze pro mřížkový (*grid*) systém. Nezapomněl ale na typografické minimum a nabízí také resetovací styly. Z definovaných kritérií se testují pouze *layout*, *responsivní design* a *typografie*. Vývojáři tento framework rádi a hojně používají díky vlastnosti čísla 960, proto byl výborný adept k testování.

Koncept 960 px šířky vznikl primárně kvůli možnosti dělení čísla 960 mnoha čísly (2, 3, 4, 5, 6, 8, 10, 12, 15, 16, 20, 24, 30, 32, 40, 48, 60, 64, 80, 96, 120, 160, 192, 240, 320 a 480). Díky tomu je mřížka široká 960 px vysoce flexibilní vůči prvkům, které se na stránce vyskytují. Jak se k tomu staví v dnešní době tento systém, když rozlišení obrazovek stoupá, je popsáno níže.

Layout Framework ve své 960px šířce nabízí hned tři varianty – 12, 16 a 24 sloupcový layout. Testovanému portálu RollingStone.com sedla lépe 12 sloupcová varianta. Kategorie Music vyžaduje dělení layoutu po 3 sloupcích (které jsou dále děleny opět na tři) nebo klasicky na jeden široký a druhý užší. V 16 sloupcové variantě je problém rozdělit již třetinu layoutu na další čitelné třetiny (vizte obr. A.10 a A.11).

960 Grid System obsahuje ve stylech i prefixy a sufixy, jimiž si vývojář nadefinuje, zdali chce mít před nebo za sloupcem místo a kolik přesně. Mezi

jednotlivými řádky lze vložit tag `<div>` s třídou `.clear`, tak aby mezi boxy vznikla větší mezera.

Responzivní web design Jak bylo uvedeno v kapitole o responzivním designu, vývojáři čelí zařízením s mnoha rozlišeními. V této souvislosti je na místě položit si otázku, zdali legendární šířka 960 px bude pro web jako takový dostatečná. Závisí to na každém zvlášť, co je jeho preferencí. Chce-li však někdo používat pouze dobrý grid systém, 960px šířka nemusí být překážkou.

Framework nabízí řešení pro adaptivní layout²⁵. Detekci šířky prohlížeče provádí pomocí javascriptu a přiřazuje jim po té CSS šité na míru. Šírky dělí na:

- mobilní pod 760 px (definuje pouze vnější okraje)
- 720 px u rozlišení 760 px do 980 px
- 960 px u rozlišení 980 px do 1280 px
- 1200 px u rozlišení 1280 px do 1600 px
- 1560 px u rozlišení 1600 px do 1940 px
- 1920 px u rozlišení 1940 px do 2540 px
- 2520 px u rozlišení nad 2540 px

Kouzlo tkví v návaznosti na samotný grid systém. Stačí do hlavičky HTML stránky přidat pár řádků a okamžitě je stránka responzivní. Je to také příklad javascriptového řešení pro definování šířek k docílení responzivního designu.

```
<meta name="viewport" content="width=device-width,
initial-scale=1, minimum-scale=1, maximum-scale=1" />
```

```
<script>
var ADAPT_CONFIG = {
  // Cesta k-CSS
  path: 'nathansmith-adapt-37a83aa/assets/css/',

  // false = Spustí se pouze jednou při načtení stránky
  // true = Mění se s-velikostí okna
  dynamic: true,

  range: [
    '0px to 760px = mobile.min.css',
```

²⁵Na své hlavní stránce se odkazuje na <http://adapt.960.gs/>.

4. TESTOVÁNÍ FRAMEWORKŮ

```
'760px to 980px = 720.min.css',  
'980px to 1280px = 960.min.css',  
'1280px to 1600px = 1200.min.css',  
'1600px to 1940px = 1560.min.css',  
'1940px to 2540px = 1920.min.css',  
'2540px          = 2520.min.css'  
]  
};  
</script>  
<script src="nathansmith-adapt-37a83aa/assets/js/adapt.min.js"></script>
```

Typografie Pokud se vývojář rozhodně používat čistě grid systém, má k tomu důvod, a proto není potřeba vkládat do frameworku více, než je nezbytně nutné k postavení slušného základu. 960 Grid System sice neobsahuje velké množství stylů pro typografii, ale to by nebyl ten hlavní problém. Tyto styly pro typografii odporují zásadám použitelnosti webu podle kapitoly 3.5, neboť výška písma je definována fixně. Uživatelé mohou zmenšit nastavení na jeho zařízení a zmenšit do takové velikosti, až jej špatně uvidí.

Potenciál využití Framework 960 Grid System se hodí pro tři skupiny lidí. První jsou weboví vývojáři, kteří si potřebují jasně a rychle navrhnout sloupcový systém a pro ostatní prvky na webové stránky mají už vlastní framework. Nebo využívají či chtějí vyvíjet v rámci své firmy framework pro své vlastní specifické potřeby a potřebují kvalitní základnu v podobě grid systému. Spíše ale než pro stavbu rozhraní by framework posloužil lépe webdesignerům.

Druzí jsou grafici webových stránek, kteří spolupracují s vývojáři a dohromady se potřebují shodnout na jednotném smysluplném layoutu. 960 Grid System totiž poskytuje šablony pro grafické programy, které pomohou při návrhu layoutu. Patří mezi ně např. Adobe Photoshop a InDesign, Gimp, Inkscape a další.

Třetí skupinou lidí mohou být začátečníci ve vývoji webového rozhraní. Díky frameworku se naučí používané návyky ve stavbě layoutu.

4.3 Návrh a realizace komponenty do CSS frameworků

Vývoj webových technologií jde neustále kupředu a pokud se přestanou vyvíjet, pro vývojáře přestanou být atraktivní a užitečné. Nehledě na to, že potřebují mít jistotu, že technologie není zastaralá a bude stále podporována. V současných frameworkcích, zejména pak ve frameworku Twitter

Bootstrap, existuje mnoho komponent a přibývají další. Při používání frameworků uživatel snadno narazí na něco, co mu schází a co by mu usnadnilo práci.

4.3.1 Návrh

Při tvorbě stránek podle předlohy Alza.cz s použitím frameworku YAML a Bootstrap vznikla potřeba nakódování reklamy kolem layoutu. Nic, co by usnadnilo reklamu vytvořit ani jeden z frameworků nepodporuje. V době, kdy vládne marketing a firmy cítí nutnost propagovat své produkty na webových stránkách, je reklama často využívaným prvkem. Cílem bylo do těchto frameworků v rámci bakalářské práce komponentu vytvořit a zjistit, jak je to v jeho rámci složité. Protože se nejedná o specifický grafický prvek, nebyl problém s designem daného frameworku.

Oba frameworky poskytují responzivní design a pokud má být reklama cílená na všechny uživatele, je nutné, aby také odpovídala na aktuální šířku prohlížeče. Do frameworků byly navrženy celkem tři druhy reklamních bannerů (proužků). Vkládání bannerů se předpokládá do HTML jako obrázek pomocí tagu ``, který může být obalený libovolnými značkami. Třídy se styly k reklamě se nacházejí v souboru *adver.css*.

4.3.2 Použití v praxi

Při tvorbě stylů byly v úvahu brány konvence pojmenovávání daných frameworků. Každý element ve frameworku YAML má prefix „ym-“, v Bootstrapu výrazná omezení v pojmenování nejsou.

Základem je horizontální banner, který může být umístěn kdekoliv na stránce (obr. A.12). Pro zajištění správného chování v případě řádkových (*inline*) prvků, je potřeba tagu obsahující banner přidat třídu `.ym-ad-hor` v YAMLU a v Bootstrapu `.ad-hor`. Jelikož YAML poskytuje pro media třídu `.flexible`, aby se banner přizpůsoboval šířce okna, stačí mu tuto třídu přiřadit. V Bootstrapu toto není potřeba, neboť ten obrázky přizpůsobuje automaticky za každých podmínek.

Dalším využívaným prvkem k upoutání pozornosti potenciálního zákazníka bývají bannery umístěné na bocích layoutu s jejich fixní pozicí (obr. 3.1). Uživatel při skrolování stránky má banner stále na očích. Tagům obsahující tyto prvky se přiřadí třída `.ym-ad-fix` nebo `.ad-fix`. Přesné umístění banneru si uživatel musí definovat ale už sám podle rozmístění svého layoutu.

Tento způsob umístění reklamy má význam pouze u velkých rozlišení, kdy kolem layoutu existuje velká část prázdného místa. V určitém okamžiku

se okno prohlížeče nebo zařízení dostane do stavu, kdy samozřejmě boční bannery nebudou viditelné. Tehdy by se reklama měla umístit dovnitř layoutu (obr. A.14), stejně jako to řeší samotná Alza.cz. Pro tento případ se použije horizontální varianta obrázku. V HTML dostane třídu `.ym-ad-res` (`.ad-res`), pro kterou je definován bod, v němž layout nemá kolem sebe již dostatek místa na zobrazení bočních bannerů. Obrázek u YAMLu by měl opět obsahovat třídu `.flexible`.

4.3.3 Zhodnocení

Při porovnávání tvorby komponent pro každý framework zvlášť nebyly shledány výrazné rozdíly. Ve frameworku Bootstrap bylo jednodušší vytvořit HTML kostru, aniž by vývojář musel myslet na více tříd, které prvky potřebují mít přiřazené. Větší pozornosti a znalost daného frameworku by byly potřeba, pokud by do něj vývojář chtěl více zasahovat. Platí podobná pravidla jako při vývoji softwaru. Svým zásahem lze nabourat navrženou koncepci.

Celkově vytvořit komponentu však nebylo nijak náročné. Lze předpokládat, že závislost na zdrojových souborech bude minimální i při dalších možných komponentách.

CSS preprocessory

Přestože první preprocessory vznikly už před několika lety, jejich možnost využití se do většího podvědomí dostává kolem roku 2010. Internetové diskuse se plní debatami, zda použít CSS preprocesor a pokud ano, tak jaký.

Důraz se klade na možnosti a vhodnost nasazení z hlediska syntaxe preprocesorového jazyka a způsoby konverze do CSS, případně další rozšiřující volby. Hodnotí se funkčnost a varianty možného provázání do aplikací. Nakonec je popsána implementace napojení preprocesorů na redakční systém.

5.1 Úvod do problematiky preprocesorů

V momentu, kdy několik vývojářů pracuje na velkém projektu (pro představu např. Google nebo Facebook), okamžitě vývojář pozná sílu preprocesorů. Stejně jako v programování se dnes používají zejména objektově orientované jazyky, přichází s tímto i komunita využívající CSS. V několika desítkách CSS souborů už opravdu nastává obrovský chaos. CSS preprocessory jsou pro něj jediným řešením.

Průkopníkem CSS preprocesorů se stal jazyk **Sass** vyvíjený od roku 2006. Z něj si vzal inspiraci **LESS**, na němž se stále usilovně pracuje. S novým a vlastním neotřelým řešením přichází **Stylus**. Vytváří si vlastní konvence, s nimiž může být obtížné se sžít, pokud jde o konzervativní programátory.

Existují i další CSS preprocessory, ale už nejsou tak dokonalé, nevyvíjejí se (např. xCSS²⁶) anebo postrádají jakoukoliv oficiální dokumentaci či ro-

²⁶Tento preprocesor má kompilaci napsanou pouze v PHP²⁷ <http://xcss.antpaw.org/>.

zumného průvodce. Kvůli uvedeným důvodům si u vývojářů zatím nezískali tak vysokou popularitu.

Populární webový portál *CSS Tricks* zabývající se CSS vylepšeními v roce 2012 uskutečnil anketu²⁸, kde zjišťoval, zdali návštěvníci využívají preprocesory a pokud ano, tak jaké preferují. Odpovědělo téměř 13 000 návštěvníků. Mezi nejčastějšími odpověďmi se objevily právě tyto preprocesory. Přestože Stylus v porovnání se Sass a LESS získal nízké procento úspěšnosti, nástroj vypadá dostatečně zajímavě na to, aby mu byla věnována pozornost. Před započatím prezentace poznatků je potřeba krátce uvést a vysvětlit, jak se používají.

5.1.1 Sass

Preprocessor Sass je postaven na Ruby²⁹. Proto vývojáři, kteří jsou webové stránky zvyklí programovat v Ruby on Rails³⁰, nebudou mít problém s nasazením, neboť se nemusí zabývat novými technikami.

Jako běžná syntaxe se používá „SCSS“ s rozšířením `.scss` a je nadmnožinou CSS3 syntaxe. Znamená to, že každé validní CSS3 styly jsou i validní SCSS. Existuje i starší syntaxe s rozšířením `.sass`. Hlavním rozdílem oproti SCSS je zjednodušený zápis – není třeba používat závorky, středníky; místo toho se kód odsazuje a zarovnává do bloků. Obě syntaxe jsou plně podporované.

Kód napsaný v SCSS:

```
#main {
  color: blue;
  font-size: 0.3em;
}
```

Kód napsaný v Sass:

```
#main
  color: blue
  font-size: 0.3em
```

K instalaci kompilátoru je nutné mít nainstalované Ruby, samotné stažení balíku a instalace se provede příkazem:

```
gem install sass
```

Překlad souborů se provede příkazem:

```
sass style.scss style.css
```

(O kompilaci více v kapitole 5.2 Kompilace a použití.)

²⁸Výsledky ke shlédnutí na <http://css-tricks.com/poll-results-popularity-of-css-preprocessors/>.

²⁹Objektově orientovaný skriptovací programovací jazyk.

³⁰Framework nad jazykem Ruby používaný k programování webových aplikací.

5.1.2 LESS

LESS funguje jak s kompilováním na straně serveru, tak s javascriptem na straně klienta v prohlížeči. Syntaxe je opět mezi LESS a CSS soubory přenositelná, ale na druhé straně vývojář nesmí žádné značení opomíjet. Může se to stát překážkou pro ty, kteří očekávají od preprocesoru ulehčení ve všech směrech včetně syntaxe. Na druhou stranu nedovolí vývojáři tvořit nečitelný kód.

Instalace se provede příkazem:

```
npm install -g less
```

Kompilace pak:

```
lessc style.less style.css
```

případně nalinkováním souboru less.js dostupném na oficiálním webu nebo Githubu³¹ projektu. Ukázka zavedení javascriptového překladu LESS souborů v HTML:

```
<link rel="stylesheet/less" type="text/css" href="styles.less">
<script src="less-min.js" type="text/javascript"></script>
```

5.1.3 Stylus

Stylus používá zjednodušenou syntaxi podobně jako Sass s rozdílem, že Stylus opomíjí všechny dělicí znaky kromě čárek při výčtu hodnot:

```
body
  font 12px Helvetica, Arial, sans-serif

a.button
  -webkit-border-radius 5px
  -moz-border-radius 5px
  border-radius 5px
```

Nevýhodou této syntaxe je odsazování vlastností, podle kterého pozná i zařazené elementy. Ovšem je nutné dbát, aby v celém dokumentu byly buď jen tabulátory nebo mezery. V opačeném případě kompilace nebude fungovat. Pro vývojáře, kteří jsou tak zvyklí používat právě tabulátory k odsazování, bude milejší pravděpodobně používání závorek.

V preprocesorovém jazyce Stylus je klasická synaxe také možná. Lze používat jak závorky tak středníky a dvojtečky. I když nebude dodržovaná, Stylus kompilaci zvládne. Ovšem otázkou je, jestli považovat za správné míchání syntaxe. Svádí potom ke kombinování a opomíjení pravidel, což zhoršuje čitelnost kódu. Zjednodušený zápis kódu bez závorek se také hůře čte, navíc editory bez speciálních pluginů nezvládnou zvýrazňování syntaxe.

³¹Jedná se o síť pro softwarový vývoj projektů jak open-source, tak komerčních.

Vyznat se v takovém zápisu není jednoduché. Ale to také závisí především na osobních preferencích.

Stylus se instaluje se pomocí Node, podobně jako u LESS:

```
npm install -g stylus
```

Kompiluje se následujícím způsobem:

```
stylus style.styl style.css
```

5.2 Kompilace a použití

Všechny jazyky je třeba dostat do takové podoby, aby mu rozuměl sám prohlížeč. Ten neumí preprocesorový jazyk interpretovat a vykreslit, neboť mu nerozumí. Proto zdrojové soubory musí být zkompileovány a toho se dá docílit třemi způsoby:

- javascriptem,
- kompilací pomocí PHP,
- kompilátorem daného jazyka.

Používat kompilaci javascriptem v produkčním režimu na webu není šťastná volba. Při každé změně souboru se okamžitě na straně klienta změní i obsah. Vhodné využití najde, když není žádoucí stále spouštět příkaz na kompilaci. V případě LESSu nahradí chybějící volbu `--watch`, jenž obsahuje Sass a Stylus. Nicméně by bylo rozumnější tuto funkčnost minimálně zavést k příkazu `lessc` oficiálně. Vývojář by si mohl vybrat, zdali chce používat javascript nebo bude využívat raději příkazovou řádku.

PHP kompilace je výhodná u PHP aplikací a jejich frameworků, které nabízejí nějakou správu assetů³². Vývojáři pak nemusí používat nějaké další prostředí pro kompilaci preprocesorů a vše se tak může dít automaticky. Díky PHP parsování mohou být CSS preprocesory integrovány do redakčních systémů či do PHP frameworků, což je ukázáno v kapitole 5.4 Integrace preprocesorů do redakčního systému.

Klasická kompilace na příkazové řádce se používá při vývoji na localhostu³³, kde má uživatel preprocesor nainstalovaný. Pokud vyvíjí webové rozhraní, použití této varianty je bezesporu nejvýhodnější. Samotný kompilátor obsahuje totiž několik užitečných nástrojů vývoj značně zpříjemňujících.

³²Statické soubory, které souvisejí s tvorbou webové stránky jako např. obrázky, css soubory apod.

³³Localhost odkazuje na místní počítač, kde běží program. Např. pokud je spuštěn webový prohlížeč na daném počítači, tento počítač je považován za localhost.

5.3 Hodnocení CSS preprocesorů

Z důvodu markantnějších rozdílů mezi nástroji preprocesorů než mezi jejich samotnými funkcemi práce srovnává schopnosti jejich **nástrojů** vývoj ulehčit. U každého nástroje je preprocesor zhodnocen 1 - 3 body vzhledem k ostatním (3 je nejvíce) následujícím způsobem:

- Pokud nástroj obsahuje každý z preprocesorů, dostanou body podle jejich vhodnosti řešení (3,2 nebo 1).
- Neobsahuje-li jeden preprocesor některý nástroj, ohodnocen bude 0 body, ostatní se mezi sebou rozdělí o 3 body. Pokud jsou nástroje rovnocenné, dostanou každý po 1 bodu.
- Poskytuje-li jen jeden preprocesor určitý nástroj a ostatní ne, bude ohodnocen 3 body.

Formát výstupu U preprocesoru Sass si vývojář může zvolit formát výstupního souboru hned ze čtyř možných variant. Kromě komprese kódu do jednoho řádku lze vybrat formát s viditelnou hierarchickou strukturou

```
.ym-vlist {
  color: #00275a; }
.ym-vlist ul {
  border-bottom: 0;
  border-top: 0; },
```

s elementy rozdělenými novým řádkem

```
.ym-vlist { color: #00275a; }
.ym-vlist ul { border-bottom: 0; border-top: 0; }
```

nebo klasické rozmístění s každou vlastností na samostatném řádku

```
.ym-vlist {
  color: #00275a; }
.ym-vlist ul {
  border-bottom: 0;
  border-top: 0; }.
```

Naopak LESS nabízí kompresi YUI³⁴ či svou vlastní zjednodušenou kompresi. Stylus obsahuje volbu pouze zjednodušené komprese.

Sass: 3, LESS: 2, Stylus: 1

Odladování chyb Pohodlné řešení debugovacího nástroje poskytuje LESS. V konzoli při chybě překladu přímo zobrazí zvýrazněnou část kódu včetně čísla řádku, v němž se nachází chyba i s podáním vysvětlení.

³⁴Yahoo! User Interface knihovna pro stavbu webových stránek. Mimo jiné poskytuje kompresor pro javascript a css soubory <http://yui.github.io/yuicompressor/>.

5. CSS PREPROCESSORY

```
ParseError: Syntax Error on line 6 in /data/less/style.less:6:12
5 body
6     font-family: @font;
7     color: @col;
```

Obrázek 5.1: Ukázka upozornění preprocesoru LESS

Upozornění podobně řeší Stylus, občas ale nedokáže specifikovat jasně daný problém nebo také chybně, tak jako na obrázku. Za vlastností `font-weight` chyběla hodnota, Stylus ale upozornil na špatné formátování.

```
/home/pave/node_modules/stylus/bin/stylus:516
    throw err;
    ^
ParseError: stylus.styl:27
23|
24| .btn
25|   font-weight:
26|
> 27| .ym-wrapper
   28|   max-width: 90em
   29|
   30| .ym-wbox
expected "indent", got "outdent"
```

Obrázek 5.2: Ukázka upozornění preprocesoru Stylus

Kdežto Sass vypíše pouze vysvětlení chyby, která se nachází okolo určitého řádku, ale vždy to přesně nedokáže specifikovat. Znamená to, že se splete o několik řádků. Všechny tři preprocesory pak obsahují debugovací nástroje pro prohlížeč Firefox – *FireSass*, *FireLESS* a *FireStylus*.

LESS: 3, Stylus: 2, Sass: 1

```
Syntax error: Invalid CSS after "
margin-top:": expected pseudoclass or pseudoelement, was " -10px;"
on line 23 of pokus.scss
Use --trace for backtrace.
```

Obrázek 5.3: Ukázka upozornění preprocesoru Sass

Hromadná kompilace a sledování změn Sass a Stylus obecně nabízí širší škálu nástrojů. Navíc obsahují např. hromadnou kompilaci css souborů v určitém adresáři nebo sledování změny souboru, kdy se příkaz na kompilaci spustí jen jednou:

```
sass --watch style.scss:style.css
stylus --watch style.styl:style.css
```

Sass: 1, Stylus: 1, LESS: 0

Zaokrouhlování Při matematických operacích Sass poskytuje volbu přesnosti zaokrouhlení u desetinných míst. Toto se užitečné jeví především při použití fluidního layoutu.

Sass: 3, LESS: 0, Stylus:0

Konverze do vlastní syntaxe Všechny tři preprocesory umožňují dokonce konverzi CSS do své syntaxe, ale žádný nenahradí opakované hodnoty např. proměnnými. Stylus nezanořuje bloky do sebe.

```
stylus --css < test.css > test.styl
```

LESS tento nástroj zvaný LESSify³⁵ oficiálně nepodporuje. Jde zatím o vývojovou verzi, ale oproti preprocesoru Stylus dokáže rozpoznat bloky, které k sobě patří a zanořit je. To už je mnohem použitelnější, potřebuje-li vývojář přenášet různé styly např. z dřívějších projektů. Sass umí také zanořovat, pokud CSS soubor obsahuje komentáře, kompilátor se s ním bezchybně nepopere. Sice vše zanoří správně, ovšem způsobí zakomentování celého kódu od bloku, kde se poprvé vyskytl. Bohužel také konvertuje do syntaxe bez závorek a středníků, což může být pro někoho překážka.

```
sass-convert test.css test.scss
```

LESS:3, Sass: 2, Stylus:1

Interaktivní režim Za zmínku stojí interaktivní mód, jenž nabízí Stylus. Na příkazové řádce lze na výstupu dostat, jak bude např. vypadat určitá barva:

Stylus: 3, Sass: 0, LESS: 0

Hodnocení Z výsledného hodnocení plynou dvě věci. Za prvé, jde vidět, že Sass se vývoji věnuje již několik let a obsahuje oproti ostatním nejvíce nástrojů z hodnocených. Za druhé, LESS dokazuje, že i když má nástrojů oproti konkurentům méně, jsou použitelnější a uživatelsky přívětivější. Stylus má ve kvalitě co dohánět, ale je na slibné cestě (tab. 5.1).

³⁵Na internetu je dostupný online konvertor <http://leafo.net/lessphp/lessify/>.

```
[. @localhost bp]# stylus -i
> color = white
=> #fff
> rgba(color, 0.5)
=> rgba(255,255,255,0.5)
> color -= rgb(200,50,0)
=> #37cdff
> █
```

Obrázek 5.4: Interaktivní mód preprocesoru Stylus

Tabulka 5.1: Hodnocení preprocesorů

Kritérium	Sass	LESS	Stylus
Formát výstupu	3	2	1
Odladování chyb	1	3	2
Hromadná kompilace a sledování změn	1	0	1
Zaokrouhlení	3	0	0
Konverze do vlastní syntaxe	2	3	1
Interaktivní režim	0	0	3
Celkem	10	8	8

5.4 Integrace CSS preprocesorů do CMS systému

Na poli redakčních systémů neboli CMS³⁶ existují implementace kompilátoru CSS preprocesorů uvnitř systémů, ale lze je najít i v samotných PHP frameworkcích (např. Symfony). Pro účely této bakalářské práce byl vybrán open source redakční systém Venne:CMS³⁷ (dále jen Venne). Tento redakční systém postavený na Nette frameworku se vyznačuje vysokou modularitou a cílením jak na uživatele, tak samotné vývojáře.

5.4.1 Motivace

Integrovaný CSS preprocesor umožní psát styly v dynamickém jazyce pro šablony přímo v CMS tak, aby kompilace mohla probíhat přímo online na daném hostingu. Je možné tedy měnit styly pro webové rozhraní přímo ve vybraném preprocesorovém jazyce.

³⁶CMS (Content Management System) Systém, který slouží pro správu webového obsahu.

³⁷Celý projekt i s moduly je volně ke stažení na <http://github.com/venne>.

Jako užitečná pomůcka se to projeví zejména ve chvíli, kdy správce webu nemá u sebe svou vývojovou verzi a je nutné rychle zasáhnout a opravit nějakou chybu či nedostatek. V opačném případě na localhostu usnadní vývojáři čas tím, že nemusí stále spouštět příkaz na kompilaci souborů při každé malé změně. Modul to vyřeší za něj.

5.4.2 Realizace

K vytvoření jednotlivých modulů, které nám zajistí kompilaci potřebujeme PHP kompilátor pro daný preprocesor. Tyto kompilátory dnes pro Sass i LESS existují. U LESS je to *lessphp* a u Sass *phpsass*. Pro Stylus zatím php kompilátor neexistuje.

Základem modulu ovládající kompilaci je vytvoření makra fungujícího v šablonovacím systému CMS. Syntaxe makra byla zvolena obdobná, jako u maker *css* a *js*, které jsou standardně dostupné:

```
{pojmenovani_makra @NazevModul/cesta_k_souboru/nazev_souboru.pripona}
```

Označením `@NazevModul` v tomto případě rozumíme modul pro konkrétní webovou stránku, ve kterém se nachází zejména šablona a dále všechny potřebné soubory k zobrazení layoutu a rozhraní stránky - tzn. obrázky, styly, fonty a nyní i soubory preprocesorového jazyka. Cesta k souboru se určuje absolutně od tohoto modulu až k samotnému souboru[9].

Aby byly dodrženy určité dobré zvyky a důslednost v daném redakčním systému, do implementace bylo zahrnuto vytváření adresáře v mezিপaměti, kam se následně zkompileovaný soubor ukládá. Není totiž žádoucí, aby se soubor kompiloval při každém načítání stránky, ale pouze při kompilaci šablony, kde je umístěné zmíněné makro. (Uvažujeme-li však nastavení CMS v debugovacím módu, makro rozliší tuto skutečnost a zkompileje less soubory vždy při změně CSS souboru.)

Část názvu souboru se vygeneruje jako hash cesty k souboru a data poslední změny. Toto nám zaručí unikátní název souboru. Nestane se, aby se do adresáře umístily dva shodně nazvané soubory a jelikož celá cesta k souboru může být příliš dlouhá, vyřeší toto hash. Hash se vygeneruje vždy stejně dlouhá.

V debugovacím módu při změně souboru je starý nepotřebný soubor se styly odstraněn díky druhé hashi v názvu. Aby vývojář sám byl schopen poznat o který soubor se jedná, na začátku názvu je umístěn původní název souboru - např. *style.less*.

5.4.3 Modul LESS

38

V případě makra pro less soubory bude zápis v šabloně vypadat následovně:

```
{less @NazevModul/cesta_k_souboru/style.less}
```

Po kompilaci less souboru a zpracování makra dostaneme v HTML stránce nalinkovaný css soubor:

```
<link rel="stylesheet"
href="/www/cache/less/styl.less-hash1-hash2.css" type="text/css" />
```

Implementace LESS makra pak závisí na dvou nejdůležitějších krocích:

- vytvoření instance less kompilátoru
- příkazu zařizujícím samotnou kompilaci na základě vstupního souboru

```
//vytvoření instance kompilátoru
$less = new \lessc();
//metoda kompilující soubor $in do souboru $out,
//kde $in a~$out jsou absolutními cestami souborů
$less->compileFile($in, $out);
```

Jelikož se redakční systém Venne dá ovládat z příkazové řádky (má v sobě zabudovanou konzoli z frameworku Symfony), do modulu byl implementován také příkaz, který kompilaci zajistí bez nutnosti instalace kompilátoru LESS do uživatelského počítače³⁸. Implementace je poněkud jednodušší, kompilaci je však nutné ošetřit vyhozením výjimky a informováním uživatele o případné chybě:

```
$less = new \lessc();
try {
    $less->compileFile($in, $out);
} catch (\Exception $e) {
    // příkaz, který na konzoli vypíše, kde se v~souboru stala chyba
    $output->writeln("<error>fatal error: {$e->getMessage()}</error>");
}
```

V případě makra se toto řešit nemusí, neboť Nette vyhazuje výjimky automaticky v nástroji zvaném Tracy (dříve Laděnka).

³⁸Volně dostupný ke stažení na <http://github.com/venne/less-module>.

³⁹Vytvořeno podle manuálu <https://github.com/Venne/venne-docs/blob/master/framework/commands.md>

5.4.4 Modul Sass

Logika zpracování souboru s obsahem preprocesorového jazyka bude při zavedení podpory každého dalšího procesoru stejná. Lišit se bude v kompilovacím příkazu a drobnostech, které implementace jednotlivého parseru odlišuje. U Sass makra použijeme následující:

```
//metoda toCss vrací string výsledného css obsahu
$css = $sass->toCss($path);
//css obsah vložíme do souboru nacházející se na absolutní adrese uložené
//v proměnné $out
file_put_contents($target, $css);
```

Implementace příkazu kompilující scss nebo sass soubory bude postavena na kombinaci logiky v LESS modulu a výše uvedených řádků:

```
$sass = new \SassParser(); //vytvoření instance kompilátoru
try {
    $css = $sass->toCss($in);
    file_put_contents($out, $css);
} catch (\Exception $e) {
    $output->writeln("<error>fatal error: {$e->getMessage()}</error>");
}
```

Uživatelé redakčního systému, kteří si přejí používat k vývoji CSS preprocesor mají nyní na výběr z LESS nebo Sass jazyka podle jejich vlastních preferencí. Oba z modulů jsou dostupné na Githubu⁴⁰. Do budoucna se určitě počítá s vytvořením i modulu pro Stylus, jakmile se pro něj objeví PHP kompilátor.

⁴⁰LessModule: <http://github.com/venne/less-module>,
SassModule: <http://github.com/venne/sass-module>

Závěr

Čtyři různé CSS frameworky mi rozšířily obzory a poskytly znalosti, na kterých bude snadné stavět další. Zjistila jsem, že na počátku každého projektu automaticky vzniká potřeba ujasnit si kritéria, která webové rozhraní má splňovat a která očekávám od použitých nástrojů.

Technologie se navzájem doplňují a podporují jedna druhou. Proto i v této práci byla snaha zaznamenat souvislosti mezi CSS preprocesory a frameworky. Čím více jsou technologie mezi sebou podporované, tím více vývojáři usnadní práci. Vývojáři často argumentují výtkami, jimiž obhajují, proč nepoužívají modernější nástroje. Tato práce však většinu dokázala vyvrátit pouhými zjištěnými poznatky, což považuji za úspěch.

Bakalářská práce je pouhým odrazem v dané problematice. Preprocesory a frameworky mi přišla natolik zajímavá, že v jejím zkoumání neustanu, budu otestované sledovat a další zkoušet a používat.

Literatura

- [1] Bacci, M.: Basic Typography Rules for a Well Designed Website. [cit. 2013-05-05]. Dostupné z: <<http://www.grokdotcom.com/2012/11/07/basic-typography-rules-for-a-well-designed-website/>>
- [2] Brunborg, E.: CSS Baseline: The Good, The Bad And The Ugly. 2012, [cit. 2013-05-02]. Dostupné z: <<http://coding.smashingmagazine.com/2012/12/17/css-baseline-the-good-the-bad-and-the-ugly/>>
- [3] Catlin, H.; Weizenbaum, N.; Eppstein, C.: *Syntactically Awesome Stylesheets*. Dostupné z: <<http://sass-lang.com/>>
- [4] Curzi, S.: *Baseline - a designer framework by Stephanecurzi.me*. 2011. Dostupné z: <<http://baselinecss.com/>>
- [5] IAC/InterActiveCorp: *Definition: CSS Framework*. [cit. 2013-03-24]. Dostupné z: <http://webdesign.about.com/od/cssglossary/g/css_framework.htm>
- [6] Jesse, D.: *YAML CSS framework*. Dostupné z: <<http://www.yaml.de/>>
- [7] Knight, K.: Adaptive CSS-Layouts: New Era In Fluid Layouts? 2009, [cit. 2013-03-24]. Dostupné z: <<http://coding.smashingmagazine.com/2009/06/09/smart-fixes-for-fluid-layouts/>>
- [8] Knight, K.: Fixed vs. Fluid vs. Elastic Layout: What's The Right One For You? 2009, [cit. 2013-03-24]. Dostupné z:

- <<http://coding.smashingmagazine.com/2009/06/02/fixed-vs-fluid-vs-elastic-layout-whats-the-right-one-for-you/>>
- [9] Kříž, J.: *Vlastní Latte makra (macros)*. 2013. Dostupné z: <<https://github.com/Venne/venne-docs/blob/master/framework/macros.md>>
 - [10] Marcotte, E.: *Responsive Web Design*. Jeffrey Zeldman, 2011, ISBN 978-0-9844425-7-7, [cit. 2013-03-24].
 - [11] Miner, W.: Setting Type on the Web to a Baseline Grid. 2007, [cit. 2013-05-10]. Dostupné z: <<http://alistapart.com/article/settingtypeontheweb>>
 - [12] Otto, M.; Chewing, F.: *Bootstrap*. 2012. Dostupné z: <<http://twitter.github.io/bootstrap/>>
 - [13] Russell, J.: Mobile now accounts for 10that of 2010: report. [cit. 2013-04-28]. Dostupné z: <<http://thenextweb.com/mobile/2012/05/09/mobile-now-accounts-for-10-of-internet-usage-worldwide-double-that-of-2010-report/>>
 - [14] Sellier, A.: *The Dynamic Stylesheet language*. Dostupné z: <<http://lesscss.org/>>
 - [15] Smith, N.: *960 Grid System*. Dostupné z: <<http://960.gs/>>
 - [16] Smith, N.: *Stylus*. Dostupné z: <<http://learnboost.github.io/stylus/>>
 - [17] Sterling, G.: Study: 69 Percent Access The Mobile Internet Daily. [cit. 2013-04-22]. Dostupné z: <<http://marketingland.com/study-69-percent-access-the-mobile-internet-daily-4371>>
 - [18] Sumner, G.: How many web browsers support responsive design? [cit. 2013-04-12]. Dostupné z: <<http://gabesumner.com/how-many-web-browsers-support-responsive-design>>
 - [19] WWW Consortium: *Cascading Style Sheets*. [cit. 2013-03-24]. Dostupné z: <<http://www.w3.org/Style/CSS/Overview.en.htm>>
 - [20] WWW Consortium: *Media Queries*. [cit. 2013-03-24]. Dostupné z: <<http://www.w3.org/TR/css3-mediaqueries/>>
 - [21] WWW Consortium: *Web Content Accessibility Guidelines (WCAG) 2.0*. [cit. 2013-05-15]. Dostupné z: <<http://www.w3.org/TR/WCAG20/>>

Obrázky

SPEED LINK Gravity Thunder 2.1 Subwoofer System



Reproduktory 35W RMS, harmonický moderní de
dálkového ovládání, aktivní subwoofer

Obrázek A.1: Nedokonalá typografie u frameworku YAML: Mezera mezi řádky nadpisu je téměř neznatelná.

Média CD, DVD
Flashdisky
Ostatní
Komponenty
Elektronika
Domácí elektro
Herní zóna
Software a e-knihy
Hračky a další
Akce



[Nakupte společně a ušetřete](#)

Hračky.cz
Kleopatra.cz
Kosmetika Avon

- [Moje objednávka](#)
- [Kontaktujte nás](#)

Reproduktory 2.1

Nejprodávanější

	Genius GX Gaming SW-G2.1 1250 černé - Skladem > 5 ks	743,- s DPH
	Reproduktory 38W (20W RMS subwoofer + 2x 9W RMS satelity)	899,-
	SPEED LINK Gravity Thunder 2.1 Subwoofer System - Skladem > 5 ks	749,- s DPH
	Reproduktory 35W RMS, harmonický moderní design, dálkového ovládání, aktivní subwoofer	906,-

Obrázek A.2: Responzivní web pomocí frameworku YAML

1. Košík

2. Doprava a platba

3. Dodací údaje

4. Souhrn

1.

Email (login):*

Heslo:*

Potvrzení hesla:*

Fakturační údaje

Jméno a příjmení (název firmy):*

Ulice:*

Město:*

PSC:*

Telefon:*

Odběr novinek

Chci odebírat akční slevy a novinky a souhlasím se zasíláním informací na email:

☐ Alza.cz

☐ Kleopatra.cz

☐ Speciál pro ženy

☐ Hračky.cz

☐ Kosmetika Avon

Ostatní

Odkud jste se o nás dověděli

Please choose

Zpět

Pokračovat




Obrázek A.3: Poznatky týkající se frameworku YAML: 1. Přestože je YAML primárně grid systém, obsahuje i pohodlné javascriptové přepínání záložek. 2. Bohužel YAML nabízí pouze tři základní délky inputů ve formuláři. Není tedy příliš přizpůsobivý potřebám uživatele. 3. Přestože do CSS nebylo nijak zasahováno a HTML standardně použito podle předpokladů YAML, checkboxy se rozmístily do nového sloupce křivě.

A. OBRÁZKY

[Úvod](#) / [PC doplňky](#) / [Reproduktory](#)

Reproduktory pro PC

Nejprodávanější

	Genius GX Gaming SW-G2.1 1250 černě - Skladem > 5 ks Reproduktory 38W (20W RMS subwoofer + 2x 9W RMS satelity)	743,- s DPH 899,-
	SPEED LINK Gravity Thunder 2.1 Subwoofer System - Skladem > 5 ks Reproduktory 35W RMS, harmonický moderní design, dálkového ovládání, aktivní subwoofer	749,- s DPH 906,-
	Genius GX Gaming SW-G2.1 3000 černě - Skladem > 5 ks Reproduktory 70W (40W RMS subwoofer + 2x 15W RMS satelity)	1 323,- s DPH 1 601,-

Top

[Nejprodávanější](#) [Od nejdražšího](#) [Od nejlevnějšího](#)

[«](#) [1](#) [2](#) [3](#) [»](#)

Genius SW-HF 2.1 1205 barva dřeva Reproduktory 32W (14W RMS subwoofer + 2x 9W RMS satelity), dřevěné	Genius GX Gaming SW-G2.1 3000 černě Reproduktory 70W (40W RMS subwoofer + 2x 15W RMS satelity)	Creative GigaWorks T3 Reproduktory 2.1 set, subwoofer 50W + 2x15W RMS	Genius GX Gaming SW-G2.1 3000 černě Reproduktory 70W (40W RMS subwoofer + 2x 15W RMS satelity)
---	---	--	---

Obrázek A.4: Ukázka použitých komponent Bootstrapu

SPEED LINK Gravity Thunder 2.1 Subwoofer System



Reproduktory 35W RMS, harmonický moderní design, dálkového ovládání, aktivní subwoofer

Cena bez DPH	749,-
Cena s DPH	906,-
Běžná cena	2 517,-
Ušetříte	64% / 1 611,-

Skladem > 5 ks

[Koupit ▾](#)

Obrázek A.5: Flexibilní obrázek ve frameworku Bootstrap: screen obrazovky v originální velikosti s přizpůsobeným obrázkem.

Nakupte společně a ušetřete

[Hračky.cz](#)

[Kleopatra.cz](#)

[Kosmetika Avon](#)

- [Moje objednávka](#)
- [Kontaktujte nás](#)

[Úvod](#) / [PC doplňky](#) / [Reproduktory](#)

Reproduktory pro PC

Nejprodávanější



[Genius GX Gaming SW-G2.1 1250 černé](#) - Skladem > 5 ks
Reproduktory 38W (20W RMS subwoofer + 2x 9W RMS satelity)

743,-
s DPH
899,-



[SPEED LINK Gravity Thunder 2.1 Subwoofer System](#) - Skladem > 5 ks
Reproduktory 35W RMS, harmonický moderní design, dálkového ovládání, aktivní subwoofer

749,-
s DPH
906,-



[Genius GX Gaming SW-G2.1 3000 černé](#) - Skladem > 5 ks
Reproduktory 70W (40W RMS subwoofer + 2x 15W RMS satelity)

1 323,-
s DPH 1
601,-

Top

[Nejprodávanější](#)

[Od nejdražšího](#)

[Od nejlevnějšího](#)

« 1 2 3 »

[Genius SW-HF 2.1 1205 barva dřeva](#)


Reproduktory 32W (14W RMS subwoofer + 2x 9W RMS satelity), dřevěné




Obrázek A.6: Responzivita frameworku Bootstrap

1. Košík 2. Doprava a platba 3. Dodací údaje 4. Souhrn

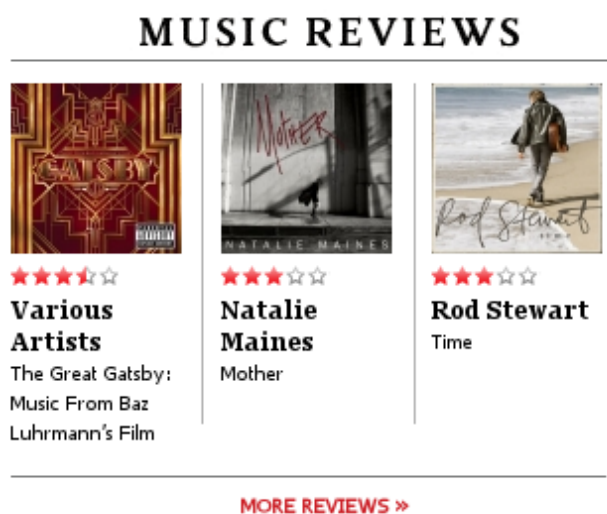
Obrázek A.7: Bootstrap: Navigace v nákupním košíku podle předlohy Alza.cz

Pete Townshend Settles Down	Most Popular
In the conclusion of Pete Townshend's 1968 Rolling Stone Interview, the Who guitarist ruminates on the state of rock and rollk	<ul style="list-style-type: none">• Music• Politics• Photos• Videos 

Obrázek A.8: Ukázka zobrazení mřížky na stránce v Baseline

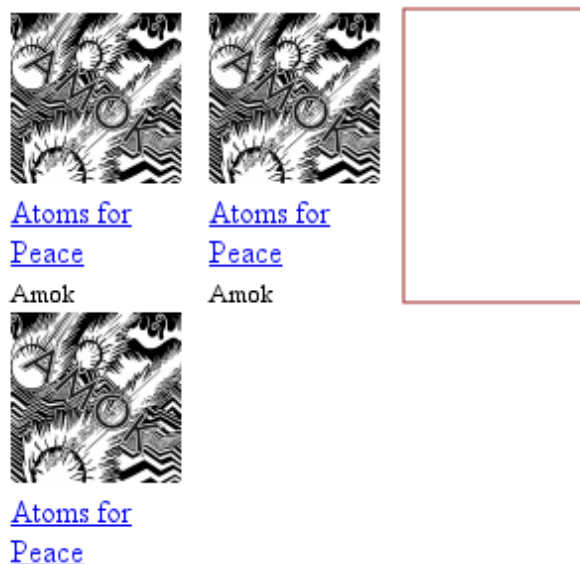
	
This is the conclusion of the Rolling Stone Interview with Pete Townshend, guitarist, composer and leader of the Who. In the first part of the interview he spoke of his guitar smashing techniques, why he does it, the mod revolution in England and narrated the story of his next record, an opera titled Deaf, Dumb and Blind Boy.	

Obrázek A.9: Baseline: Rozhozená koncepce účaří

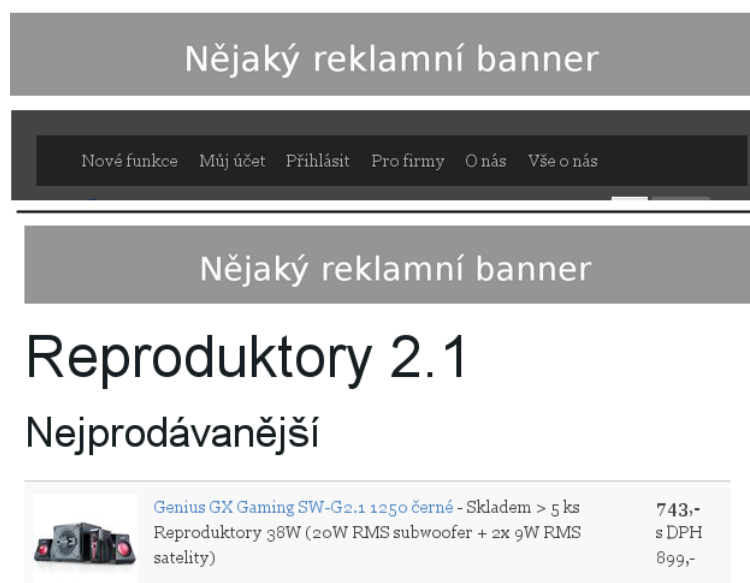


Obrázek A.10: Předloha z RollingStone.com pro vytvoření 3 sloupců v jednom

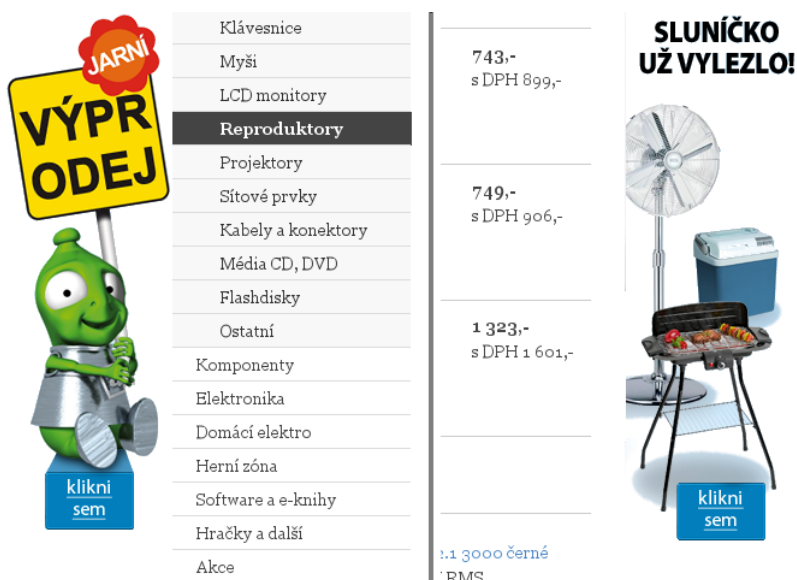
Music Reviews



Obrázek A.11: 960 Grid System: Ve sloupci už nezbylo místo na poslední položku podle předlohy



Obrázek A.12: Komponenta: Libovolný horizontální banner kdekoli na stránce s libovolnou šířkou okna.



Obrázek A.13: Komponenta: Reklamní bannery fixně podél layoutu vlevo a vpravo.



Obrázek A.14: Komponenta: Horizontální obdoba bočních bannerů využívaná při responzivním designu.

Seznam použitých zkratk

Seznam použitých zkratk

CMS (Content Management System) Systém, který slouží pro správu webového obsahu.

CSS (Cascading Style Sheets). Jazyk navržený W3C organizací představuje jednoduchý mechanismus přidávání stylů (např. fontů, barev, mezer) do webových dokumentů.

DSL (Dynamic Stylesheet language) Dynamický jazyk.

HTML (HyperText Markup Language) Značkový jazyk používaný pro tvorbu webových stránek.

IE (Internet Explorer) Internetový prohlížeč.

PHP (PHP: Hypertext Preprocessor) Hypertextový preprocesor. Jedná se o skriptovací jazyk, který je používán k vývoji webových stránek. PHP se provádí na straně serveru.

Sass (Syntactically Awesome Stylesheets) CSS preprocesor.

W3C (World Wide Web Consortium) Organizace vyvíjející webové standardy.

YAML (Yet Another Multicolumn Layout) CSS framework.

Obsah přiloženého CD

readme.txt	stručný popis obsahu CD
src	
├─ css-frameworky	zdrojové kódy testovaných CSS frameworků včetně komponenty
├─ moduly-do-cms	zdrojové kódy modulů CSS preprocesorů do Venne:CMS
├─ thesis ...	zdrojová forma práce ve formátu \LaTeX včetně použitých obrázků
text	text práce
├─ thesis.pdf	text práce ve formátu PDF
├─ thesis.ps	text práce ve formátu PS