ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

ZADÁNÍ BAKALÁŘSKÉ PRÁCE

Název:	Responzivní webové stránky služby Webgarden
Student:	Filip Trefil
Vedoucí:	Mgr. Monika Součková
Studijní program:	Informatika
Studijní obor:	Web a multimédia
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

Cílem práce je úprava a rozšíření stávající prezentační vrstvy služby Webgarden s ohledem na moderní přístupy responzivního rozhraní.

- Navrhněte HTML a CSS strukturu pro použití na webových stránkách klientů vytvořených službou Webgarden.
- Layout navrhněte tak, aby byl automaticky responzivní pro jakýkoliv obsah na webu.
- Při návrhu dbejte na zásady použitelnosti webového rozhraní.
- Navrženou strukturu naimplementujte do XSLT šablon, které Webgarden používá. Zachovejte zpětnou kompatibilitu se stávajícím řešením.
- Upravte také uživatelské rozhraní administrace (pro desktop i mobilní) s cílem zjednodušit a zpřehlednit vytváření struktury webu.
- Vše důkladně otestujte.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 2. února 2016

České vysoké učení technické v Praze

Fakulta informačních technologií

Katedra softwarového inženýrství

Bakalářská práce

Responzivní webové stránky služby Webgarden

Filip Trefil

Vedoucí práce: Mgr. Monika Součková

17. května 2016

Poděkování

Děkuji své vedoucí Mgr. Monice Součkové za vedení mé práce. Děkuji kolegům ve Webgarden za jejich pomoc a konzultace. Také děkuji své rodině za podporu během celého studia.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) 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. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 17. května 2016 .....

České vysoké učení technické v Praze
Fakulta informačních technologií
© 2016 Filip Trefil. 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

Trefil, Filip. *Responzivní webové stránky služby Webgarden.* Bakalářská práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato bakalářská práce se zabývá úpravou populárního systému pro tvorbu webů Webgarden. Cílem práce bylo navrhnout a implementovat responzivní chování na webech vytvořených pomocí Webgarden. Dále je součástí práce návrh nové podoby administrace. Práce je rozdělena na analýzu současného systému, dobré praktiky responzivního designu, návrh a implementaci. Při návrhu byla snaha vymyslet co nejuniverzálnější a zároveň dobře fungující layout. Implementace využívá moderní techniky jako například CSS flexbox.

Klíčová slova Responzivní webdesign, HTML, CSS, preprocesory, flexbox, uživatelské rozhraní, typografie, použitelnost, Webgarden

Abstract

This thesis focuses on improving the web service Webgarden. The aim was to create a responsive layout for websites created with the Webgarden and also to design a new user interface for the administration. The thesis is divided into four parts – analysis, responsive design best practices, design and implementation. The purpose of the design was to create an universal and well-functioning layout. The implementation takes advantage of modern technology like CSS Flexbox.

Keywords Responsive webdesign, HTML, CSS, preprocessors, flexbox, user interface, typography, usability, Webgarden

Obsah

Úvod **1**

1 Cíl práce **3**

2 Současný stav systému **5**
2.1 HTML struktura 5
2.2 CSS 8
2.3 Uživatelské prostředí administrace 9
2.4 Konkurenční služby 13

3 Responzivní webdesign **17**
3.1 Základní principy responzivního webdesignu 18
3.2 Responzivní layout 19
3.3 Komponenty stránky 22
3.4 Typografie 25
3.5 Obrázky 27

4 Návrh **31**
4.1 Responzivita 31
4.2 Prostředí administrace 37

5 Implementace **43**
5.1 HTML 43
5.2 CSS 46
5.3 XSLT 50

Závěr **51**

Literatura **53**

A Testování uživatelského rozhraní **55**

A.1 Testování s uživatelem 1 55
A.2 Testování s uživatelem 2 56

B Návrhy uživatelského rozhraní administrace 60
B.1 Administrace – záložka obsah 60
B.2 Vyskakovací okna 61

C Seznam použitých zkratek 71

D Obsah přiložené SD karty 73

Seznam obrázků

2.1 Struktura skupiny .group 6
2.2 Základní layout stránky 6
2.3 Obsahová sada 7
2.4 Řádek obsahu 8
2.5 Současná administrace Webgarden 10
2.6 Ukázka Webnode 14
2.7 Ukázka Webmium 15
2.8 Ukázka eStránky 15

3.1 Layout: Mostly fluid 20
3.2 Layout: Column drops 20
3.3 Layout: Layout shifter 21
3.4 Layout: Tiny tweaks 21
3.5 Layout: Off-canvas 22
3.6 Navigace: Responzivní lišta 23
3.7 Navigace: Off-canvas 24
3.8 Navigace: Priority+ 25
3.9 Písmová osnova 26
3.10 Základní parametry typografie 26
3.11 Mezery mezi texty 27
3.12 Porovnání velikostí obrázků 29

4.1 Třísloupcový layout 33
4.2 Dvousloupcový layout 34
4.3 „Zpětné zalamování“ bloků 35
4.4 Návrh horizontálního menu 35
4.5 Návrh vertikálního menu 36
4.6 Fixní lišta 36
4.7 Znázornění struktury stránky 38
4.8 Přidat obsah v sekci 38

4.9 Obsahový prvek po najetí myši 39
4.10 Přidat obsah mezi prvky 39
4.11 Umístění prvků pod sebe 40
4.12 Přidání nového obsahu 41
4.13 Grafické zpracování nastavení 41

5.1 Základní layout - nový 44
5.2 Obsahová sada 45
5.3 Porovnání uživatelského obsahu 45
5.4 Třídy podle šířky sloupce 49

B.1 Administrace: Obsah 60
B.2 Popup – Výběr prvků 61
B.3 Sloupec 62
B.4 Titulek 62
B.5 Odstavec textu 63
B.6 Odkaz 63
B.7 Obrázek 64
B.8 YouTube video 64
B.9 Video z PC 65
B.10 QR 65
B.11 Soubor 66
B.12 Formulář 66
B.13 Přihlášení k newsletteru 67
B.14 Přihlašovací okénko 67
B.15 Vyhledávání 68
B.16 Košík 68
B.17 Nejnovější zboží 69
B.18 Facebook 69

Seznam tabulek

2.1 Zjištění expertní analýzy 11

3.1 Přístupy na internet podle druhů zařízení 17

A.1 Uživatelské testy: Uživatel 1 55
A.2 Uživatelské testy: Uživatel 2 56

Úvod

Prohlížení webu již dávno není pouze doménou počítačů. Čím dál čatěji si webové stránky prohlížíme na mobilním telefonu, tabletu či čtečce elektronických knih. Podle většiny předpovědí budou tato zařízení brzy tvořit většinu přístupů.

Tomu se samozřejmě musí přizpůsobit i samotné weby. Správné zobrazení na mobilních zařízeních je dnes naprosto základním požadavkem při tvorbě webových stránek. Nejvhodnějším řešením tohoto problému se ukazuje být responzivní design. Responzivní web se v jediné své variantě zobrazí správně na jakémkoliv zařízení včetně počítače, tabletu, telefonu a téměř čehokoliv jiného.

Webgarden je populární česká webová služba, která umožňuje komukoliv vytvořit si vlastní webové stránky bez znalosti kódování. Vzniklé weby bohužel nejsou nijak přizpůsobené pro mobilní zařízení, což je pro systém poměrně značná konkurenční nevýhoda. Proto jsem se rozhodl upravit jej tak, aby jeho klienti mohli své nově vytvořené weby korektně zobrazit i na displeji mobilního telefonu.

Kapitola 1

Cíl práce

Cílem této práce je upravit systém Webgarden tak, aby weby vytvořené pomocí tohoto nástroje byly responzivní. Záměrem je vytvořit novou verzi šablon, které si uživatel v případě zájmu bude moci zvolit. Navržený systém musí splňovat tyto požadavky:

- Layout musí fungovat automaticky (bez zásahu uživatele) a to pro jakýkoliv obsah na webu
- Zpětná kompatibilita, musí být možné přecházet mezi staršími a novými šablonami
- Podpora ve všech majoritních prohlížečích
- Dodržení standardů responzivního webu a typografie

Navržené řešení naimplementuji pomocí HTML, CSS a XSLT šablon do systému Webgarden.

Dále je potřeba upravit uživatelské prostředí administrace. To je ve své současné podobě pro mnoho uživatelů příliš složité a neintuitivní. Cílem je provést analýzu současného řešení a na základě jejích výsledků provést návrh nové uživatelského rozhraní administrace, konkrétně části týkající se správy a vytváření obsahu.

Kapitola 2

Současný stav systému

Webgarden prošel od svého vzniku v roce 2006 dvěma většími proměnami. Ve své práci jsem vycházel ze současné verze, která vznikla v roce 2010.

Logická vrstva systému je napsaná v PHP, data jsou uložena v MySQL databázi. Vzhledem k tomu, že jsem v rámci práce nepřidával žádné nové funkce, do těchto vsrtev jsem zasahoval minimálně, či vůbec. V analýze jsem se proto soustředil na vrstvu prezentační. Ta využívá XSLT šablon, které převádí data ve formátu XML na výsledný HTML kód stránek. Ten je nastylován standardně pomocí CSS.

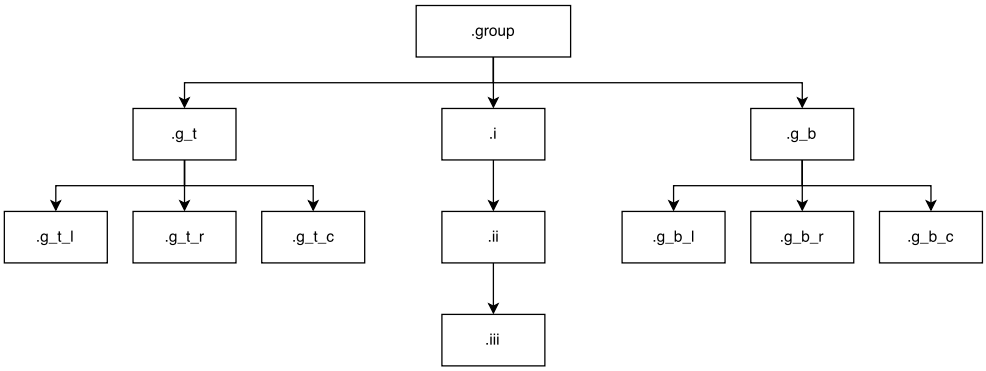
Další částí je analýza uživatelského rozhraní administrace, především záložky „Obsah“ zaměřené na vytváření a správu obsahu a struktury webu. V této části jsem pomocí expertní analýzy a uživatelských testů otestoval použitelnost UI.

Na závěr jsem provedl průzkum nejvýznamnějších konkurenčních služeb. Zkoumal jsem především to, zda a jakým způsobem řeší zobrazování na mobilních zařízeních.

2.1 HTML struktura

Současná HTML struktura systému Webgarden obsahuje velké množství vnořených bloků (prvků s tagem). Hlavní měrou se na tom podílí skupina *.group*. Ta obaluje většinu samostatných obsahových bloků v kódu. Její struktura je znázorněna na obrázku 2.1.

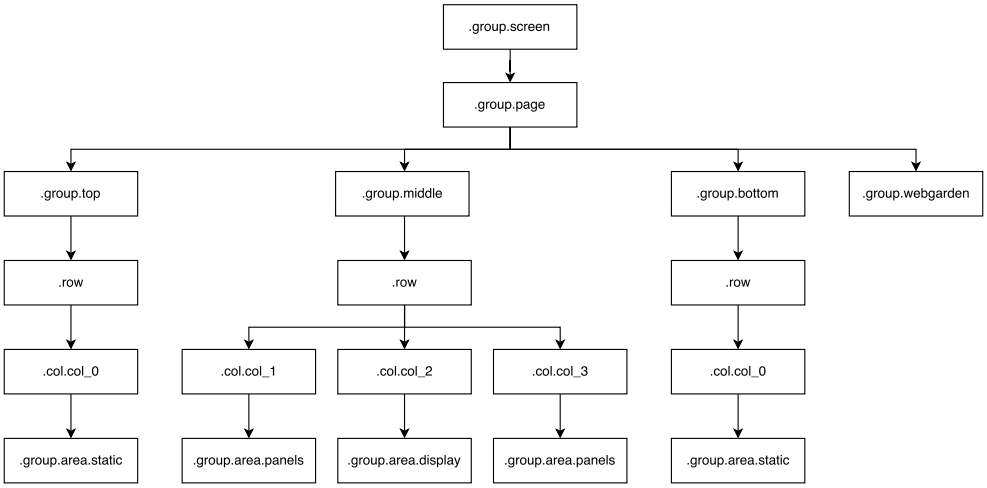
Budeme-li chápat HTML dokument jako strom modelu DOM (Document Object Model), pak každé použití struktury *.group* zvýší hloubku stromu o 4 a navíc přidá dvě slepé větve, které nemají žádný obsah. Toto řešení je dáno především historicky, kdy v minulosti z důvodu kompatibility se všemi prohlížeči bylo pro správné fungování vnějších okrajů nutné použít několik do sebe vnořených bloků. Větve *.g_t* a *.g_b* potom sloužily pro stylování zaoblených rohů.



Obrázek 2.1: Struktura skupiny .group

2.1.1 Základní layout stránky

Základní layout stránky se skládá z několika obsahových částí (sloupců) – hlavičky, prostřední části s levým, prostředním a pravým sloupcem, patičky a také lišty s dodatečnými informacemi (souhlas s cookies, poskytovatel služby). Toto uspořádání je znázorněno na obrázku 2.2. Pokud je některý z bloků *.top* a *.bottom* prázdný, systém jej vynechá. Podobně sloupce *.col_1* a *.col_3* se v kódu nemusí vyskytovat, pokud do nich uživatel nenahrál žádný obsah. V takovém případě má prostřední sloupec třídu *.col_12* (chybí levý sloupec), *.col_23* (chybí pravý sloupec), případně *.col_123* (chybí oba sloupce).

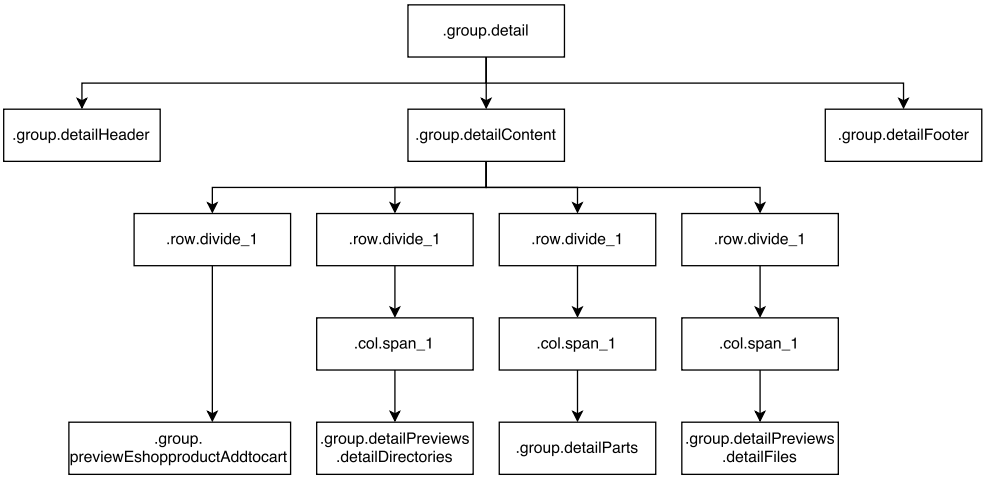


Obrázek 2.2: Struktura základního layoutu stránky (Skupina *.group* je pro přehlednost znázorněna jen jedním uzlem.)

Každý z těchto bloků má v sobě vnořený blok třídy *.area*, který má ještě další třídu označující typ obsahu. Třídy *.static* (hlavička a patička) a *.panels* (postranní sloupce) značí statický obsah, který je na všech stránkách stejný. Třída *.display* je potom určena pro prostřední obsahový sloupec, ve kterém je obsah pro každou stránku jiný. [3][1]

2.1.2 Obsahové sloupce

Obsah každého ze sloupců je ve Webgarden vnímán jako samostatný hierarchický prvek (obsahová sada). Obsah prostředního sloupce a obsah ostatních sloupců jsou tedy z pohledu systému naprosto rovnocenné. V důsledku tak např. pravý sloupec může mít vlastní podkategorie, zboží, atd. Obsahová sada je uspořádána podle schématu na obrázku 2.3. (Ve statických blocích se z praktických důvodů nikdy nezobrazuje *.detailHeader* ani *.detailFooter* a v kódu je proto vložena jen část *.detailContent*).



Obrázek 2.3: Struktura obsahové sady

Část *.detailHeader* obsahuje titulek dané stránky a drobečkovou navigaci pro jednodušší orientaci na webu. Jednotlivé části (případně celá hlavička) mohou být v administraci nastaveny jako skryté.

Část *.detailContent* je rozdělena na čtyři podčásti a je v ní umístěn samotný obsah sady. V *.detailDirectories* jsou zobrazeny náhledy podrubrik. Podčást *.previewEshopproductAddtocart* je přítomna pouze u položek e-shopu a obsahuje základní informace o zboží, jako je název zboží, cena, dostupnost, atd. Jde o obsah, který je sice vložen uživatelem, ale má pevně danou strukturu a je tak u všech položek uspořádán stejně. V *.detailParts* najdeme obsah vložený a uspořádaný uživatelem. Jedná se typicky o text, obrázky nebo for-

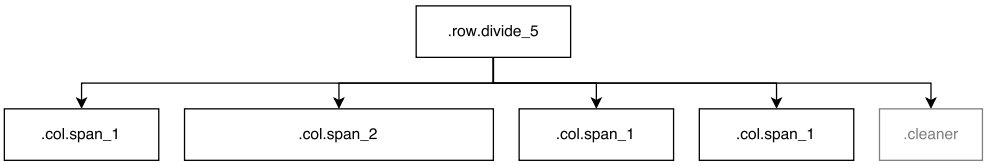
muláře. Nakonec *.detailFiles* slouží pro zobrazení náhledů dokumentů, tzn. miniatur zboží, článků, podkategorií, atd.

Uvnitř *.detailFooter* jsou vloženy informace o autorovi, datu poslední změny a možnosti sdílení.[1]

Náhled obsahové sady, který má třídu *.preview*, má totožnou strukturu jako *.detail*. Liší se jen názvy podtříd, kdy slovo „detail“ je vždy nahrazeno slovem „preview“.

2.1.3 Uživatelský obsah

Obsah uvnitř obsahové sady je uspořádán v matici $m \times n$ o m řádcích a n sloupcích, kde m je libovolné a n je z intervalu $\langle 1, 8 \rangle$. Každý řádek matice je reprezentován blokem *.row*, který obsahuje bloky *.col* o šířce 1 až n sloupců a maximální celkové šířce n. Na konci řádku je prvek *.cleaner* (nutný kvůli použitému stylování). Struktura řádku je znázorněna na obrázku 2.4.



Obrázek 2.4: Ukázka struktury řádku obsahové matice

Uvnitř bloku *.col* může být další matice, nebo některý z obsahových prvků – text, obrázek, formulář, aj.

Tato struktura je pro implementaci responzivního designu nevhodná. Každý řádek je obalen ve svém vlastním bloku a tak je zalamování obsahu možné jen v rámci jednoho řádku.

Samotné obsahové bloky se pak neřídí podle žádných pevných pravidel.

2.2 CSS

Uspořádání prvků na stránce je řešeno pomocí bloků *.row* a *.col* a to technikou plovoucích prvků (floating). Každý řádek má kromě třídy *.row* i třídu *.divide_m*, kde m je počet sloupců matice. Sloupce (*.col*) jsou pomocí deklarace *float: left* ukotveny vlevo a tím pádem se za nimi nezalamuje obsah, i přesto, že se jedná o blokové elementy. Každý sloupec má ještě třídu *.span_n*, kde n je šířka bloku. Na konci každého řádku je vložen blok *.cleaner*, který zalomí obsah za posledním sloupcem. Obecný zápis CSS stylů tedy vypadá nějak takto:

```
.row {
    width: 100%;
```

```
}
.col {
    float: left;
}
.divide_m > .span_n {
    width: (n / m) * 100%;
}
.cleaner {
    clear: both;
}
```

Z ostatních prvků jsou v základních stylech naimplementovány pouze základní funkční prvky jako například vysouvací menu, atd. Většina ostatních prvků je nastylována v jednotlivých šablonách. Šablony jsou vzhledové varianty, mezi kterými si uživatel může volit a přepínat. Prvky tak mohou být v každé šabloně nastylovány a řešeny odlišně.

2.3 Uživatelské prostředí administrace

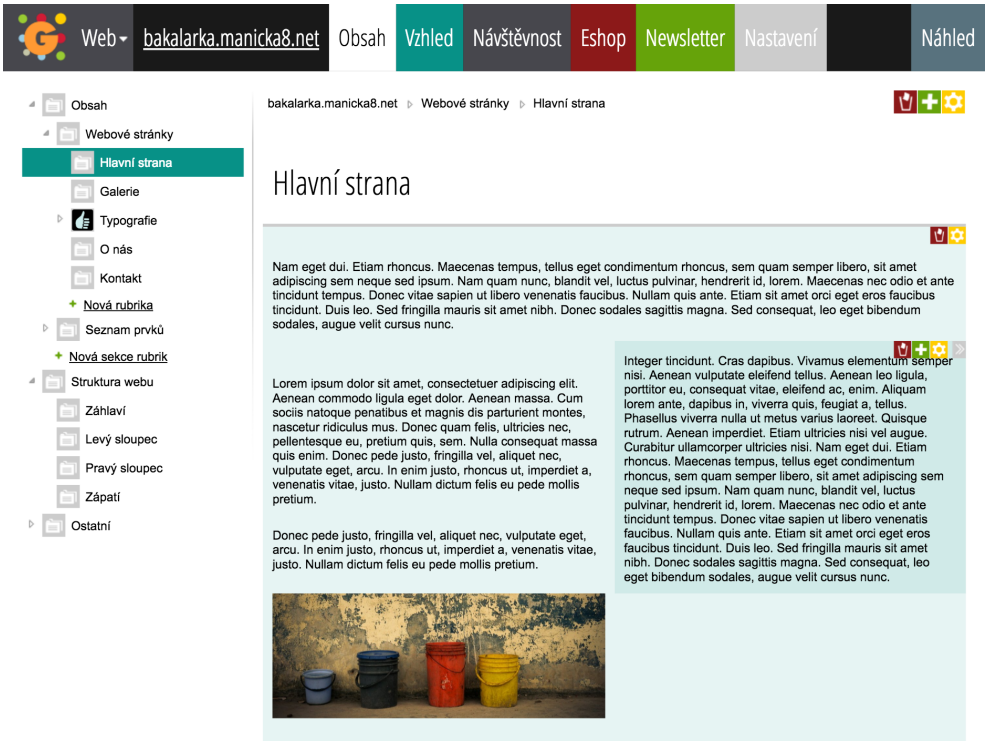
Uživatelské prostředí se skládá ze tří hlavních částí:

Horní lišta V horní liště se nachází menu s odkazy na jednotlivé oddíly administrace (Obsah, Vzhled, Eshop, Nastavení...). Vlevo potom najdeme odkaz na adresu spravovaného webu a také rozcestník, z kterého se lze dostat na všechny weby daného uživatele, založit nový web, koupit prémiové služby, atd.

Levý sloupec Slouží pro výběr stránky webu, kterou chceme upravovat, případně pro výběr některé ze statických částí webu (hlavička, levý sloupec, atd.)

Hlavní část Zobrazuje obsah konkrétní stránky, dokumentu, nebo statické sekce. Tento obsah je možno upravovat, mazat, případně přidávat nový.

Při analýze jsem vycházel z bakalářské práce Františka Baumy „Testování online nástrojů pro tvorbu internetových stránek“.[6] Testuje v ní různé online služby pro tvorbu webů, mimo jiné i Webgarden. Z důvodu porovnatelnosti výsledků jsem zvolil stejné nástroje a postupy testování. U expertní analýzy se jedná o heuristickou evaluaci. Pro testování s uživateli jsem oproti původní metodologii[6, str. 23] lehce pozměnil průběh i hodnocení testů. Provedená analýza se nezabývá založením webu a některými dalšími věcmi a naopak více do hloubky testuje práci s obsahem.



Obrázek 2.5: Současná podoba administrace služby Webgarden.

2.3.1 Expertní alnalýza

Jako forma expertní analýzy byla zvolena heuristická evaluace (heuristická analýza). Při ní se hodnotí všechny části artefaktu a ověřuje se, zda platí heuristická pravidla.[13] Heuristickou analýzu je vhodnější zvolit v případě, kdy není jasný postup a úkoly, které vedou ke splnění cíle. To je přesně náš případ, a proto jsem zvolil tuto metodu. Nejčastěji používaná pravidla pro kontrolu jsou podle tzv. Nielsenovy heuristiky[11], kterými jsem se řídil i při tomto testu.

Při provádění analýzy jsem vytvářel web podle předlohy, která byla zkonstruována tak, aby pokryla většinu nejdůležitějších a nejpoužívanějších nástrojů. Otestovány nicméně byly i ty nástroje, které tato šablona nepokryla.

Nalezené problémy jsem roztřídil do tří skupin podle závažnosti:

Kritické Chyby, které znemožňují, nebo mohou znemožnit dokončení úkolu.

Závažné Jako závažné jsou označeny chyby, které způsobí, že úkol zabere delší čas a znamenají jisté problémy při plnění úkolu.

Mírné Mírné chyby jsou takové, které jsou spíše kosmetické a způsobí pouze menší potíže.

Po dvojím otestování artefaktu jsem nalezl celkem 8 porušení Nielsenových pravidel, z nichž 1 bylo označeno jako kritické, 3 jako závažné a 4 jako mírné. Všechny uvádím v následující tabulce:

Tabulka 2.1: Seznam porušení Nielsenových pravidel zjištěných při expertní analýze

Kritické	Použití prvku „Sloupec“ je velmi složité a pro běžného uživatele nepochopitelné. Tento prvek je přitom jediným způsobem, jak vkládat užší prvky pod sebe.
Závažné	V administraci chybí tlačítko zpět.
Závažné	Při editaci prvku jsou některé možnosti ukryté pod záložkou „Rozšířené“. Tu lze velmi snadno přehlédnout.
Závažné	Mnoho možností má poměrně nejasnou terminologii a není zřejmé, co dělají. U některých vizuálních změn uživatel neví, jak budou vypadat – typicky „Styl zobrazení menu“ nebo „Zvýraznění“.
Mírné	Nastavení šířky a počtu buněk je neintuitivní a složité.
Mírné	Při přidávání obsahu jsou jednotlivé prvky nepřehledně uspořádány, velká část je jich navíc skryta pod záložkou „Pokročilé“.
Mírné	Vyskakovací okno se v některých případech objeví mimo obrazovku.
Mírné	Tlačítko „+“ na konci řádku v některých případech (podle typu obsahu) přidá prvek na nový řádek. Naopak tlačítko „+“ na začátku řádku může v některých případech přidat prvek na konec předchozího...

2.3.2 Testování s uživatelem

Další fází analýzy použitelnosti bylo testování s uživateli. Záměrem bylo otestovat rozhraní adnimistrace s lidmi, kteří znalostmi odpovídají typickým uži-

vatelům služby Webgarden. Tedy lidé, jež mají relativně dobré znalosti práce s počítačem, ale v oblasti tvorby internetových stránek jsou naprostými laiky.

Administraci jsem otestoval s těmito účastníky:

Uživatel 1 17 let, žena, lehce pokročilé znalosti práce s PC, internet používá denně, téměř žádné znalosti tvorby webu

Uživatel 2 43 let, žena, základní znalosti práce s PC, internet používá několikrát do týdne, naprosto žádné znalosti tvorby webu

Účastníci neměli pevně dané úkoly ani postup, pouze stejný výchozí a cílový stav. Zadání bylo stejné jako v případě expertní analýzy. Zachoval jsem i klasifikaci problémů na tři úrovně, jen byl lehce pozměnen jejich popis.

Kritické Účastník potřeboval větší pomoc moderátora, případně vůbec nebyl schopen úkol dokončit.

Závažné Úkol zabral neúměrně mnoho času, účastník si nebyl jistý, co dělá, zkoušel. Případně byla potřebná malá rada od moderátora.

Mírné Uživatel úkol zvládnul s lehkými obtížemi, ale bez pomoci moderátora.

Při tomto testování se potvrdila většina nedostatků zjištěných při expertní analýze. Nicméně účastníci testu měli velké množství problémů, které předchozí analýza neodhalila. Nejčastější problémy byly následující (jejich podrobný výpis je v příloze A).

- Nepochopení základních obsahových částí – záhlaví, zápatí, postranní sloupce a obsahová část.
- Nepochopení prvku „Sloupec“, který je příliš nejasný a komplikovaný.
- Uživatelé nezjistí, že mohou prvky posouvat a měnit jejich velikost.
- Název rubriky vnímají jako součást obsahu webu (a snaží se jej mazat, atd.).
- Problémy při přidávání obsahu.
- Nejasná terminologie – kořen menu, zvýraznění, zarovnání, atd.
- Některé možnosti jsou schované v záložkách. Uživatelé je neobjevili.

Možnostem řešení těchto problémů a návrhu upraveného uživatelského rozhraní administrace se věnuji v kapitole Návrh v sekci 4.2.

2.4 Konkurenční služby

Na domácím trhu existuje poměrně velké množství společností, které nabízí podobné služby, jako nabízí Webgarden. Z těch nejvýznamnějších se jedná například o tyto služby:

- Webnode.cz
- Webmium.cz
- Estranky.cz

Všechny tyto nástroje již responzivní stránky nabízejí, předmětem analýzy proto byl hlavně jejich přístup k responzivnímu designu. Nutno říci, že všechny zkoumané služby řeší responzivitu poměrně povrchně. Žádnou významnější inspiraci pro vlastní návrh jsem proto nenačerpal.

2.4.1 Webnode

Webnode má oproti Webgarden poměrně omezené možnosti uspořádání. Obsah se skládá pod sebe do takzvaných sekcí, což jsou vodorovné bloky obsahu s vlastním pozadím. V rámci těchto sekcí je možné vkládat obsah (text, obrázek, formulář...) do sloupců. Na počátku je sloupec jeden se stoprocentní šířkou, přemisťováním prvků lze sloupce přidávat a libovolně měnit jejich šířku.

Vytváření komplikovanějších struktur je ale (úmyslně [7]) velmi komplikované a nepohodlné, mnohdy nemožné. To je částečně kompenzováno předdefinovanými sekcemi jako „ceník“, „reference“, „produkty“, atd.

Responzivita je u Webnode řešena poměrně jednoduše:

- Všechny sloupce v sekci se při šířce stránky menší než 600 px roztáhnou na 100 % šířky.
- Předdefinované sekce se při 768 px přeuspořádají
- Písmo se se zvětšující se šířkou webu zvětšuje – breakpointy jsou 600 px, 768 px a 900 px

2.4.2 Webmium

Co se týká možností uspořádání i prostředí administrace, je Webmium svým pojetím velmi podobné Webnodu. Prvky se řadí pod sebe a mají 100% šířku. Je nicméně možné použít prvek „Sloupce“, který vytvoří 2 – 4 sloupce, jejichž šířku lze libovolně měnit. Do těchto sloupců lze přesouvat jiné obsahové prvky.

Administrace nabízí stejně jako u Webnodu živý náhled webu, ve kterém si uživatel může upravovat, vytvářet a mazat obsah.

Pravidla pro responzivní design jsou velmi základní:

Obrázek 2.6: Ukázka webu vytvořeného služnou Webnode

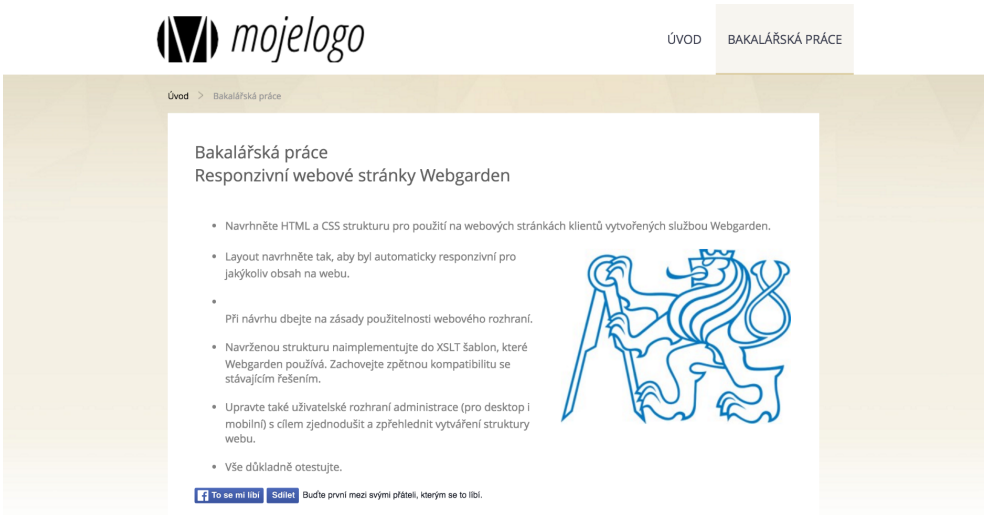
- Všechny sloupce se při šířce stránky menší, než 600 px zalomí pod sebe a roztáhnou na 100 % šířky.
- Rozložení předdefinovaných prvků jako např. galerie se nijak nemění
- Velikost písma u nadpisů se zmenší při šířce 480 px

2.4.3 eStránky.cz

Přístup, který volí eStránky.cz, je od předchozích dvou služeb značně odlišný. Na každé stránce se vyskytuje logo a horní menu. Dále lze pro jednotlivé stránky nahrát úvodní obrázek, který se zobrazí pod menu a nad ostatním obsahem na stránce. Samotný obsah stránky uživatle přidává v editoru, jenž je podobný například programu Microsoft Word. Možnosti vytváření složitějšího rozložení jsou tedy velmi omezené.

Stránky mají možnost zapnutí mobilní verze, ve které se horní menu přesune dospod stránky a jednotlivé položky menu jsou řazeny pod sebou. Jiná pravidla pro responzivní chování web nemá, nicméně vzhledem k téměř nutně jednosloupcovému layoutu stránky ani nejsou potřeba.

Obrázek 2.7: Ukázka webu vytvořeného služnou Webmium



Obrázek 2.8: Ukázka webu vytvořeného služnou eStránky.cz

Kapitola 3

Responzivní webdesign

Zatímco ještě v roce 2013 tvořily přístupy z mobilních zařízení v České republice přibližně 5,5 % všech zhlédnutých stránek, v březnu 2016 to byl již více než čtyřnásobek. Konkrétně přišlo na stránky z mobilních zařízení 40,2 % návštěvníků a prohlédli si 22,9 % z celkového počtu stránek.[5]

Tabulka 3.1: Přístupy na internet podle jednotlivých druhů zařízení (Zdroj: Netmonitor.cz) [5]

	Reální uživatelé	**Zhlédnuté stránky**
Mobilní telefony	3 149 408	1 331 616 309
Tablety	1 358 879	386 451 907
Počítače	6 707 714	5 774 435 823

Je tedy logické, že existuje silná (a neustále se zvětšující) poptávka po tom, aby se webové stránky zobrazily korektně i na těchto typech přístrojů. Pro dosažení tohoto cíle existuje několik přístupů: [2]

Mobilní verze stránek Samostatný web určený pro mobilní zařízení, většinou na doméně *m.example.com*. Nevýhodou je nutnost udržovat vedle sebe dvě verze webu a také např. časová prodleva při přesměrování z klasické verze stránek.

Mobilní aplikace Přístup na web skrz mobilní aplikaci. Tato varianta se však ukazuje jako nákladná a nepříliš efektivní.

Responzivní webdesign Další možností je mít jen jednu verzi webu, jejíž obsah se přizpůsobí právě používanému zařízení. Výhodou tohoto přístupu je snadná udržovatelnost a flexibilita – obsah se i v budoucnu zobrazí správně na jakémkoliv zařízení.

Adaptivní design Podobný přístup, jako v případě responzivního webdesignu. Místo plynulých změn má adaptivní web několik „breakpointů“, tedy hraničních šířek, při kterých se celý layout zásadně promění, nicméně zachová si fixní šířku. Breakpointy jsou zpravidla tři – desktop, tablet a mobil. Vzhledem k tomu, že mobilní zařízení mají obrovské množství form faktorů, je adaptivní přístup velmi omezený a ve většině případů nedostačující.

Právě responzivní design se i podle průzkumů ukazuje být nejvhodnějším a nejeefektivnějším řešením problému.

3.1 Základní principy responzivního webdesignu

Navrhování responzivního webu s sebou nenese jen nutnost „přeskládat“ obsah. Je třeba všechny aspekty navrhnout tak, aby web pro jednotlivé případy užití poskytoval vždy co nejlepší uživatelský zážitek.

Plovoucí struktura Jednotlivé větší obsahové celky webu jsou uspořádány v mřižce. U responzivního webu se tato mřížka plynule mění podle toho, kolik prostoru na stránce je k dispozici. Toto chování zaručuje, že na každé obrazovce se obsah zobrazí co nejideálněji.

Přzpůsobený obsah Na malou zobrazovací plochu se samozřejmě vejde podstatně méně obsahu, než například na 15” obrazovku notebooku. Je proto potřeba pečlivě zvážit, jaký obsah uživateli zobrazit a také jej zobrazit pokud možno ve stručné formě. Takto prezentovaný obsah ve většině případů vede k jasnějšímu a silnějšímu sdělení. Responzivní přístup tak pomáhá vytvářet přehlednější stránky.

Ovládací prvky Způsob ovládání počítače a mobilního telefonu se velmi liší a proto tomu by se měly přizpůsobit i ovládací prvky na webu. Například tlačítka by měla být dostatečně velká, aby je bylo možno bezpečně zaměřit prstem. Stejně tak prvky, které se zobrazí po najetí myši (typicky navigace), by měly reagovat například na klepnutí prstem, či se jinak přizpůsobit.

Typografie Přehledný a čitelný text velmi zásadně přispívá k dobrému zážitku z webu. Je proto i velmi důležitou součástí responzivního designu.

Datová velikost Zatímco rychlost připojení u klasických počítačů bývá zpravidla dobrá, připojení skrze mobilní data může být i velmi pomalé.

> Podle průzkumů 40 % návštěvníků opustí stránku, pokud se nenačte do 3 sekund.[15] Je tedy žádoucí optimalizovat web na co nejmenší datovou velikost.

Ve službě Webgarden obsah webu určují sami uživatelé, proto se jím v následujících kapitolách nebudu více zabývat. Zaměřím se ale na ostatní oblasti jako jsou layout stránky, typografie, či ovládací prvky.

3.2 Responzivní layout

Účelem responzivního layoutu je umožnit, aby se každý obsah na webu vykreslil správně a přehledně. Proto je jeho navrhování vždy specifické s ohledem na to, co se na webu nachází. Rozsáhlou databázi stránek i s jejich řešením responzivity lze najít např. na `mediaqueri.es`.

Můžeme ovšem stanovit několik základních typů layoutů, jenž dobře porkývají většinu řešení a ze kterých lze při návrhu vlastního layoutu vycházet. Tyto typy shrnul Pete LePage na webu Google Developers. [9]

Mostly fluid

U layoutu Mostly fluid se obsahové bloky (kontejnery) postupně zalamují pod sebe a rozvržení stránky se tak mění velmi pozvolna.

- Obsah má většinou stanovenu maximální šířku, nad kterou se již neroztahuje.
- Na větších obrazovkách je hlavní kontejner společně s jedním vedlejším na prvním řádku. Další sloupce jsou rovnoměrně poskládány za sebou a postupně se zalamují.
- Při menší šířce se hlavní kontejner roztáhne na 100 % šířky, vedlejší se zalomí na nový řádek.
- Na nejmenších viewportech jsou obsahové bloky seřazeny pod sebou v jednosloupcovém layoutu.

Podstatnou výhodou tohoto řešení je jednoduchá implementace, často pomocí jednoho breakpointu v CSS.

Column drops

V případě webu s vícesloupcovým layoutem může být vhodné použít Column drops.

- Obsah zpravidla nemá omezenou šířku a roztahuje se až do krajů.
- Na velkých obrazovkách jsou sloupce zobrazeny vedle sebe.

Obrázek 3.1: Znázornění layoutu Mostly Fluid

- Postupně se zmenšující se šířkou se vedlejší sloupce zalamují.
- Po dosažení minimálního breakpointu se všechny kontejnery zalomí pod sebe se 100% šířkou. Hlavní kontejner je zobrazen jako první.

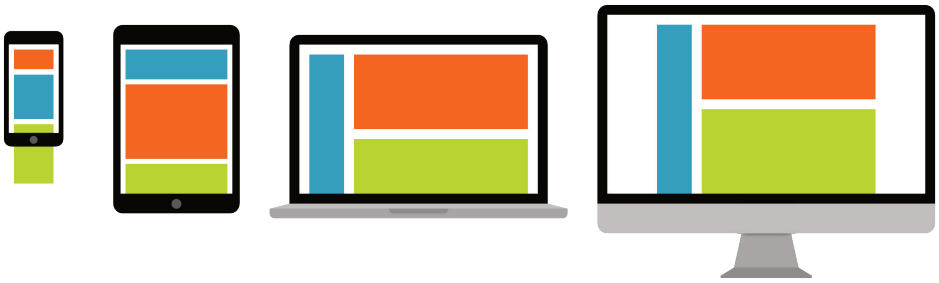
Obrázek 3.2: Znázornění layoutu Column drops

Layout shifter

Označení pro komplexní layout, který se při breakpointech může velmi zásadně proměnit.

Na schématu níže je znázorněn jednoduchý příklad s levým sloupcem. Tento sloupec se při malé šířce přesune do horní části viewportu a současně změní orientaci na horizontální.

Změny mohou být nicméně libovolné. Společným znakem je potřeba měnit nejen kontejnery, ale i obsah uvnitř nich. To zvyšuje složitost a vede k horší udržovatelnosti. Na druhou stranu takto navržený layout může být velmi efektivní.

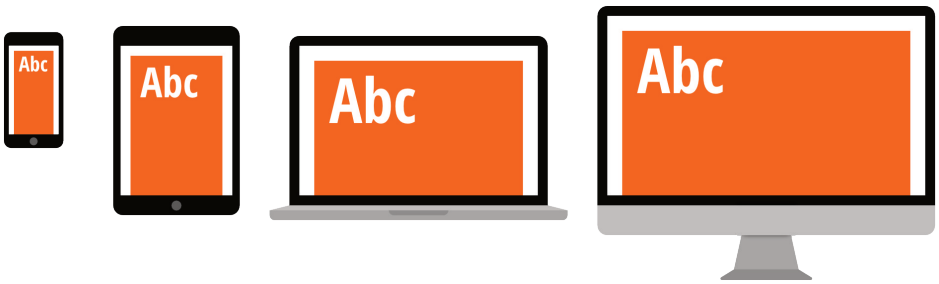


Obrázek 3.3: Znázornění layoutu Layout shifter

Tiny tweaks

Tento layout je oproti předchozímu naopak velmi jednoduchý. Při všech šířkách zachovává jednosloupcový layout. Jen s rostoucí šířkou se zvětšují okraje kontejneru a velikost písma.

Tento layout se velmi často používá na (v dnešní době velmi populární) jednostránkové weby, rozsáhlé články, atd.



Obrázek 3.4: Znázornění layoutu Tiny tweaks

Off-canvas

Layout Off-canvas volí poněkud odlišný přístup. Místo zalamování obsahu se na menších obrazovkách sekundární obsah ukryje po stranách, odkud jej lze vyvolat kliknutím nebo potažením prstem. Tato možnost je vhodná pro navigaci či rozsáhlejší menu v aplikaci, atd.

Obrázek 3.5: Znázornění layoutu Off-canvas

3.3 Komponenty stránky

Kromě responzivního layoutu je třeba upravit i některé specifické komponenty webu. Typicky jsou to interaktivní prvky jako tlačítka, navigace nebo formuláře. Kvůli menší přesnosti při ovládání prstem je potřeba dodržet tyto podmínky:

Umisťování do rohů Důležitá tlačítka je dobré umisťovat do rohů, kde je nejjednodušší dosáhnout prstem (levý dolní roh u telefonů, horní rohy u tabletů).

Dostatečná velikost Interaktivní prvky (CTA tlačítka, odkazy menu, formulářová pole) by měly být dostatečně velké, aby bylo možné jednoduše na ně zamířit prstem. Obecně se jako minimum považuje 44 bodů.

Dostatečné mezery Stejně tak by mezi tlačítky měl být dostatek místa, aby necházelo k „ukliknutím“.

Nepoužívat hover stavy Zobrazení obsahu při „hover“ stavu (při najetí myši nad prvek) může být v některých případech efektivní. Na dotykových zařízeních nicméně takový stav neexistuje. Proto je vhodné se na malých obrazovakách této techniky vyvarovat.

Využití gest Oproti klasickým počítačům naopak nabízí dotyková zařízení velké množství gest (swipe, pinch-to-zoom, atd.) Tato gesta jsou běžně používaná v operačním systému i napříč aplikacemi, proto jejich implementace do webu zlepší uživatelský zážitek.

Dále je potřeba pro zobrazení na menších viewportech přizpůsobit složitěji strukturovaný obsah jako jsou například tabulky.

3.3.1 Navigace

Navigace webu bývá běžně umístěna v hlavičce. Odkazy v menu jsou uspořádány vodorovně (vedle sebe) a i na širokých obrazovkách zabírá často celou šířku stránky. Takto uspořádaná navigace není vhodná pro zobrazení na zařízeních s úzkým viewportem. Položky se nevejdou vedle sebe a začnou „přetékat“ ze stránky ven.

Proto je navigace pro malé šířky přizpůsobena tak, aby nenarušovala strukturu a zároveň byla zachována její funkčnost. Stejně jako v případě responzivního layoutu existuje několik přístupů, které mohou posloužit jako základ pro vytvoření vlastního řešení.

Responzivní lišta

Jednou z možností je položky menu zalomit pod sebe, podobně jako u kontejnerů. Toto řešení je velmi jednoduché a jeho výhodou je, že zobrazuje navigaci přímo, bez nutnosti klikání.

Bohužel při větším počtu odkazů je takto řešená navigace velmi rozměrná a může odsunout důležitý obsah mimo zobrazenou plochu.

Obrázek 3.6: Znázornění zalamování položek menu pod sebe

Off-canvas

Zřejmě nejčatějším řešením je skrytí / schování navigace mimo stránku. Na malých obrazovkách je místo menu zobrazeno pouze tlačítko, které slouží k jeho zobrazení. Často najdeme kombinaci s fixní lištou, která při scrollování zůstává v horní části obrazovky, aby bylo menu přístupné z kterékoli části stránky.

Výhodou tohoto řešení je jednoduchost a také to, že umožnuje zobrazit i velké množství odkazů pod sebou. Také lze snadno vytvářet elegantní layouty – drobnou ikonku je totiž daleko snažší umístit, než rozsáhlou navigaci.

Nevýhodou je nenápadnost. Měření ukazují, že použití off-canvas menu ve většině případů má za následek méně prokliků na další stránky.

Obrázek 3.7: Znázornění řešení navigace Off-canvas

Priority+

Řešení nazývané „Priority+“ může být dobrým kompromisem, který eliminuje nevýhody obou předchozích řešení.

Ve výchozím stavu, pokud je na stránce dostatek horizontálního místa, jsou zobrazeny všechny položky menu, seřazené zleva doprava podle důležitosti. Při menších šířkách se zobrazí jen položky, které se vejdou na řádek. Na konci řádku je potom tlačítko „vše“. To zobrazí všechny položky, ať už odkrytím ostatních odkazů, nebo např. zobrazením překryvného okna.

Takto koncipované menu tedy zobrazuje nejdůležitější odkazy (na které uživatel s největší pravděpodobností bude chtít kliknout) přímo, ale zároveň nezabírá příliš mnoho prostoru na stránce.

Navigace jako obsah

Některé weby, jako například informační portály, mají velmi komplikovanou, složitě strukturovanou navigaci s několika úrovněmi. V těchto případech je zpravidla samotná navigace hlavním obsahovým prvkem a proto se více hodí zobrazit ji klasicky jako obsah. Uživatel se musí úrovněmi proklikat, nicméně stále má dobrý přehled o tom, kde se právě nachází.

Obrázek 3.8: Znázornění řešení navigace Priority+

3.3.2 Tabulky

Tabulky jsou velmi složitě strukturovaný obsah a proto není jednoduché stanovit obecná pravidla pro jejich responzivní chování. Velmi záleží na konkrétním obsahu. V některých případech je možné určitou část obsahu skrýt a nechat tak více prostoru pro důležitější část dat. Pokud toto nelze udělat, existuje možnost porušit strukturu tabulky a zobrazit každý její řádek samostatně s přizpůsobeným uspořádáním.

3.4 Typografie

Účelem webu je zobrazit návštěvníkovi obsah. Nejčastěji jde o obsah ve formě textu, proto je kvalitní typografie velice důležitá. Pojem typografie v tomto kontextu zahrnuje strukturování textu a volbu parametrů písma jako font, stupeň, řez, výška řádku, atd. Dále do typografie patří některé vlastnosti obsahových elementů jako vnější a vnitřní okraje nebo rámečky. [12]

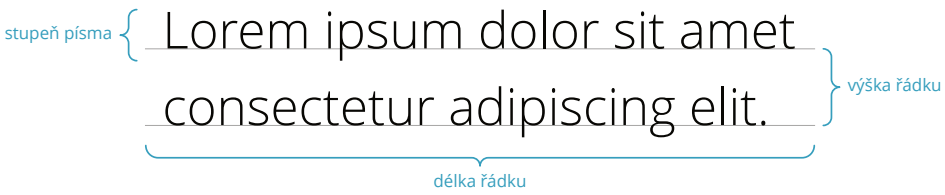
3.4.1 Základní pojmy

Stupeň písma Označuje velikost písma. Odpovídá výšce od horní po dolní dotažnici.

Výška řádku Vzdálenost od účaří dvou následujících řádků textu.

Střední výška písma Určuje výšku minusek („malých písmen“).

Obrázek 3.9: Písmová osnova



Obrázek 3.10: Základní parametry pro typografii na webu

3.4.2 Velikost písma

Optimální velikost písma úzce souvisí s pozorovací vzdáleností. Orientačně můžeme prohlásit, že zatímco pozorovací vzdálenost u telefonu je přibližně 25 cm, u notebooku jde přibližně o dvojnásobek.[14] Proto je dobré na menších obrazovkách používat menší písmo.

Výchozí stupeň písma pro běžný text na desktopových prohlížečích je zpravidla 12 pt. Tato velikost je ve většině případů přiměřená, nicméně je možné ji v intervalu několika málo stupňů měnit. Podle použitého fontu se totiž může písmo se stejným stupněm jevit jako různě veliké. Obecně platí, že čím větší střední výška písma, tím je písmo opticky větší.

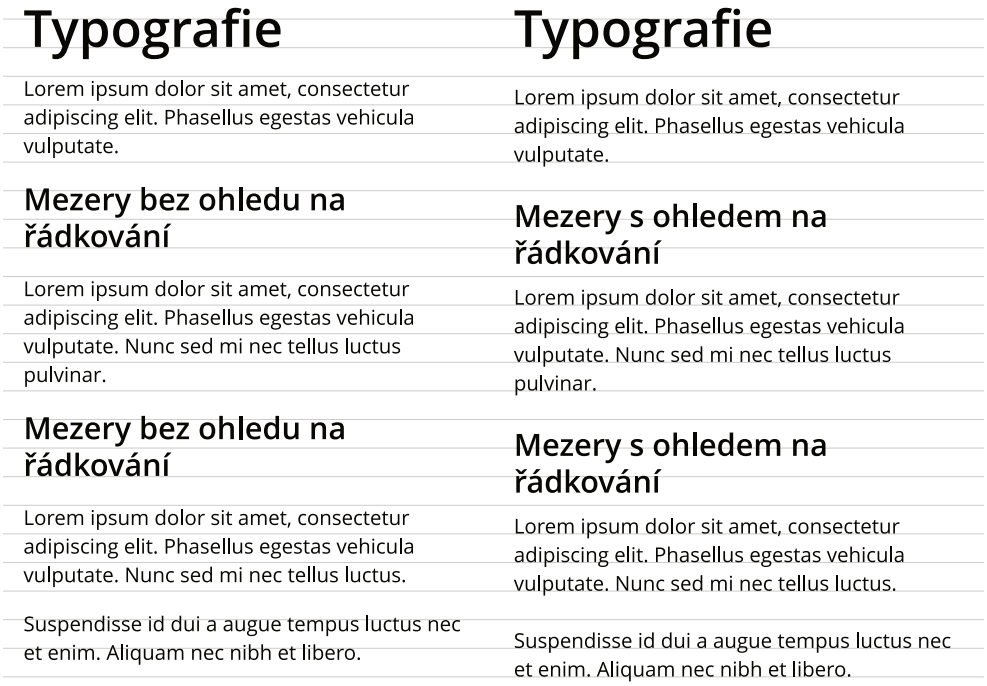
Co se týká nadpisů, jejich stupeň by měl být minimálně $1,6\times$ větší, než stupeň písma odstavce. Často je používán poměr zlatého řezu, tedy $1 : 1,618$. Jedná se ale spíše o doporučení, hlavním cílem je především to, aby byly nadpisy dostatečně vizuálně oddělené od odstavcového textu.

3.4.3 Výška řádku

Řádkování má značný vliv na celkovou přehlednost a čitelnost textu. Příliš malá výška řádku může způsobit špatnou čitelnost a překrývání znaků, příliš velká zase narušuje celistvost textu.

Pro výpočet výšky řádku neexistují přesná pravidla. Záleží na použitém fontu a stupni písma. Pro stupeň 12 pt jsou vhodné hodnoty okolo 140 % (1.4násobek stupně písma). Se zvětšující se velikostí písma tato hodnota klesá.

S výškou řádku souvisí i mezery mezi jednotlivými odstavci či mezi odstavcem a nadpisem. Mezi odstavcem a následujícím nadpisem musí být větší odstup, než mezi nadpisem a souvisejícím odstavcem.



Obrázek 3.11: Porovnání mezer mezi texty

3.4.4 Délka řádku

Dalším z faktorů, které mají vliv na čitelnost textu, je délka řádku. U delších řádků je těžší udržet oči na řádku, užší texty zase nutí čtenáře příliš často přecházet na nový řádek.

Obecně doporučovaná hodnota je mezi 45 a 75 znaky na řádek, nicméně na webu, který se může zobrazit na obrovském počtu různých obrazovek, je těžší tuto hodnotu dodržet. Ani na webu by ale za žádných okolností délka řádku neměla přesáhnout 110 znaků.[8] Na menších obrazovkách mobilních telefonů jsou řádky kratší, maximálně okolo 60 znaků.

3.5 Obrázky

Podle průzkumů 40 % návštěvníků opustí stránky, pokud se nenačtou do 3 sekund.[15] Při pomalém mobilním připojení tak každý ušetřený kilobajt

může znamenat rozdíl mezi tím, zda si návštěvník stránku prohlédne, či nikoliv.

Průměrná datová velikost webové stránky v březnu 2016 byla 2296 kB. Z tohoto čísla 1457 kB, tedy celých 63,5 %, tvořily obrázky.[4]

3.5.1 Optimalizace

Prvním krokem je optimalizace samotných obrázkových souborů. Výběrem správného formátu a optimalizací je možné výrazně snížit datovou velikost obrázku.

Fotografie či realistické ilustrace je vhodné ukládat ve formátu JPG. Tento formát používá ztrátovou kompresi a není vhodný pro obrázky obsahující větší plochy stejné barvy, neboť na nich zanechává viditelné artefakty. Pro jednodušší grafiku je daleko vhodnější bezzstrátový formát PNG. Pro jednoduché tvary potom může být vhodné použít vektorový formát SVG. Jeho výhodou je, že jej lze neomezeně škálovat bez ztráty kvality a ideálně se proto zobrazí na každém zařízení.

Pro další snížení datové náročnosti existují služby, které obrázek ještě více zoptimalizují. Pro formát JPG jde například o `compressjpeg.com`, pro PNG pak `tinypng.com`. Použitím těchto služeb lze ušetřit až desítky procent dat.

3.5.2 Různé varianty obrázků

Velikost, ve které se obrázky zobrazí na webu, se podle použitého zařízení může lišit až několikanásobně. Představme si situaci na obrázku 3.12, kde je na dvou různých zařízeních (moderní laptop s vysokým rozlišením a obyčejný smartphone) zobrazen obrázek o šířce 100 % obrazovky. Pokud porovnáme rozlišení, v jakém se obrázek vykreslí na telefonu (320 px) a na laptopu (2560 px), vidíme, že rozdíl v šířce je osminásobný a rozlišení je tedy 64× větší. V případě, že bychom chtěli na obou zařízeních použít stejný zdrojový soubor a zároveň zachovat kvalitu, museli bychom na mobilním telefonu zbytečně stahovat mnohonásobně větší obrázek, než je reálně potřeba.

Daleko vhodnější je ovšem každému z těchto zařízení poskytnout obrázek jen tak velký, jak je potřeba. K tomu slouží v HTML atribut *srcset*. Ten se přiřazuje k obrázku a obsahuje informace o různých variantách obrázku. Na základě těchto informací potom prohlížeč rozhodne, který z nabízených souborů zobrazí. Syntaxe vypadá následovně:

```
<img src="obrazek" srcset="alternativni_velikost rozmer">
```

Fixní velikost obrázku

Pokud máme na webu obrázek, jenž je za všech okolností zobrazen ve stejné velikosti, stačí vytvořit varianty obrázku pro zařízení s různou hodnotou *device-*

Obrázek 3.12: Porovnání velikostí obrázku zobrazeného na telefonu a na desktopu

pixel-ratio, neboli s různou hustotou pixelů. Místo rozměrů obrázku se v tomto případě zadává hodnota device-pixel-ratio, při které se má obrázek zobrazit.

```
<img src="obrazek.png" srcset="obrazek_2.png 2x, obrazek_3.png
↪  3x">
```

Proměnná velikost obrázku

Pokud se velikost obrázku na různých zařízeních liší, je třeba k atributu *srcset* přidat ještě atribut *sizes*. V něm jsou uloženy informace, jak široký se obrázek na webu zobrazí. Ve výchozím stavu prohlížeče počítají s šířkou 100 vw (100 % šířky viewportu), čili v takovém případě není potřeba atribut *sizes* vyplňovat.

```
<img src="obrazek.png" sizes="33vw" srcset="obrazek_2.png 500w,
↪  obrazek_3.png 1000w">
```

Velmi často je obrázek na různých šířkách zobrazen v jiné velikosti vůči šířce stránky. V takových případě lze v *sizes* nadefinovat pravidla podobná media-queries z CSS. Platná je první vyhovující varianta.

```
<img src="obrazek.png" sizes="(min-width: 1024px) 500px,
↪  (min-width: 768px) 50vw, 100vw" srcset="obrazek_2.png 500w,
↪  obrazek_3.png 1000w">
```

Kapitola 4

Návrh

4.1 Responzivita

Při návrhu responzivního layoutu jsem potřeboval navrhnout řešení použitelné na jakémkoliv webu. Vzhledem k poměrně širokým možnostem systému Webgarden to znamená, že toto řešení musí fungovat na jednoduchém jednosloupcovém layoutu (např. `speechmate.cz`) i na daleko komplikovanějších stránkách s postranními sloupci a složitou obsahovou strukturou (`vykuppc.cz` či `pedag-poradna.pel.cz`).

Tato variabilita také znamená, že jsem pro návrh měl velice málo informací o struktuře obsahu. V následujícím seznamu shrnuji informace, ze kterých jsem při návrhu vycházel.

- Web je rozdělen na 5 základních obsahových částí. Jsem schopen zjistit, které z těchto části jsou na stránce přítomny a které ne.
- Uživatelský obsah je rozdělen do matic. U každého bloku vím, jakou má šířku vzhledem ke svému rodiči. Jsem tím pádem schopen vypočítat jeho reálnou velikost vzhledem ke stránce.
- Neznám celkovou strukturu webu. Vždy mám informace pouze o konkrétním prvku, nevím, s jakými jinými prvky sousedí.
- Neznám sémantiku obsahu. Například pokud uživatel vloží menu, nejsem schopen poznat, zda se jedná o hlavní navigaci stránky, či o nepodstatný seznam odkazů.
- Uživatelé navrhují strukturu obsahu pro desktop. Není proto možné použít přístup mobile-first.

Protože na webu znám strukturu základních obsahových částí, mohu pro jednotlivé varianty uspořádání navrhnout specifické responzivní chování. Naopak jediná informace, kterou mám o obsahu vloženém uživatelem, je jeho

velikost. Mohu tedy zajistit, aby se žádný z těchto prvků nezobrazil menší, než je je stanovená hranice. Pokud by k tomu mělo dojít, zalomí se blok na nový řádek, kde bude mít dostatek prostoru.

U konkrétních obsahových prvků neznám jejich význam, proto mám velmi omezené možnosti manipulace s nimi. Nemohu například na mobilním telefonu přesunout hlavní navigaci do horní lišty, neboť nevím, které z vložených menu k tomuto účelu slouží.

Pro velké obrazovky si strukturu webu tvoří sám uživatel. Od určitého bodu (breakpointu) je potřeba, aby fungovalo responzivní řešení.

4.1.1 Hlavní obsahové sloupce

Pro základní layout stránky jsem vytvořil tři varianty podle (ne)přítomnosti postranních sloupců. Prostřední sloupec je přítomen vždy a hlavička s patičkou mají 100% šířku a nehrají v tomto smyslu roli.

Při vytváření layoutu jsem vycházel z návrhového vzoru Column drops (viz obrázek 3.2). Následující varianty fungují pro hodnoty breakpointu okolo 1000 – 1200 px a pro poměr mezi sloupci přibližně $1 : 2 : 1$ v případě třísloupcového layoutu a přibližně $1 : 3$ u dvousloupcového. Konkrétní hodnoty se nicméně mohou lišit.

Pro každou z variant platí následující pravidla a proměnné:

- *breakpoint* = šířka, při které se začne zalamovat obsah
- *obsah* = procentuální šířka prostředního sloupce
- *sloupec* = procentuální šířka postranního sloupce
- Limitní šířka prostředního sloupce je $obsah \times breakpoint$
- Limitní šířka postranních sloupců je $sloupec \times breakpoint$
- Pokud je šířka sloupce menší než limitní, začne se uvnitř něj zalamovat obsah
- Maximální šířka sloupce je $1,5 \times$ *limitní šířka*

Třísloupcový layout

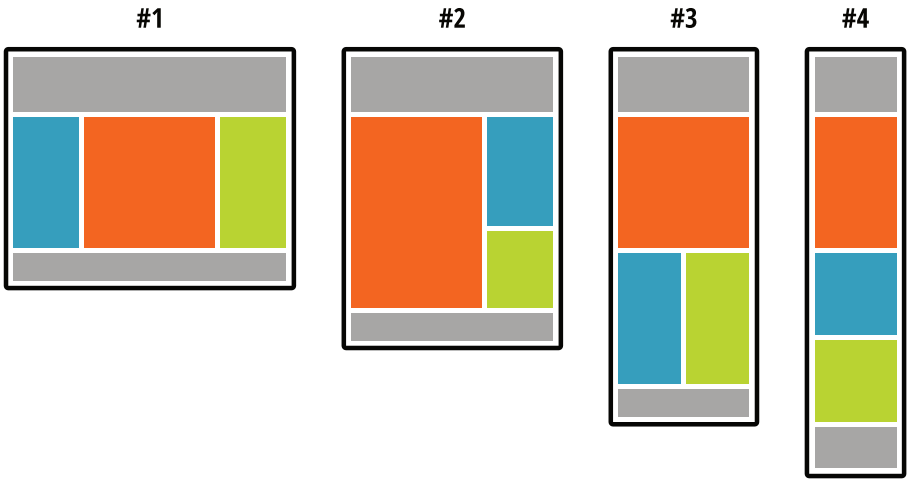
U třísloupcového rozložení se nejprve postranní sloupce umístí pod sebe a přesunou doprava. Hlavní část tak má dostatek prostoru pro vykreslení obsahu a zároveň není potřeba sloupce odsouvat do spodní části webu. Sloupce a obsahová část jsou zobrazeny v poměru $obsah : sloupec$

Při šířce $breakpoint \times (obsah + sloupec)$ se zalomí oba postranní sloupce a zobrazí se vedle sebe pod obsahovou částí.

V momentě, kdy už je stránka příliš úzká na to, aby se některé dva sloupce vešly vedle sebe, dojde k přechodu na jednosloupcový layout. Vzhledem k tomu, že součet limitních šířek postranních sloupců může a nemusí být větší, než limitní šířka obsahového sloupce, vypočítá se hraniční šířka jako

$$breakpoint \times \max(obsah,\ 2 \times sloupec)$$

To zaručí, že každý ze sloupců bude vždy dostatečně široký.



Obrázek 4.1: Ukázka zalamování třísloupcového layoutu

Dvousloupcový layout

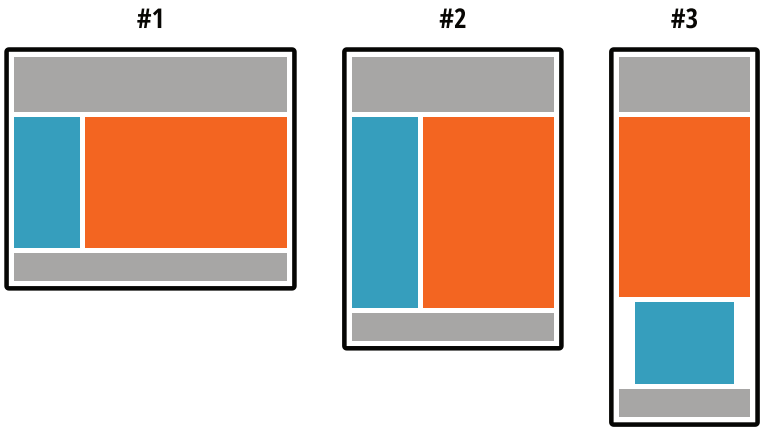
Trochu jiný přístup jsem zvolil v případě dvousloupcového layoutu. Abych co nejdéle zachoval rozložení webu se dvěma sloupci, dojde k zalomení postranního sloupce až při šířce *breakpoint* × *obsah*. To ovšem znamená, že obsahová část je užší než je limitní šířka. Dojde proto k zalamování vnitřního obsahu už od breakpointu.

Jednosloupcový layout

U této varianty je zobrazen pouze prostřední blok, který má šířku 100 %. Už od šířky *breakpoint* zde proto dochází ke změně uspořádání vnitřního obsahu podle pravidel v kapitole 4.1.2.

4.1.2 Uživatelský obsah

Pro vnitřní obsah, tedy obsah vytvořený uživatelem, jsem zvolil řešení podobné layoutu Mostly fluid (obrázek 3.1). Každý blok má stanovenou výchozí



Obrázek 4.2: Ukázka zalamování dvousloupcového layoutu

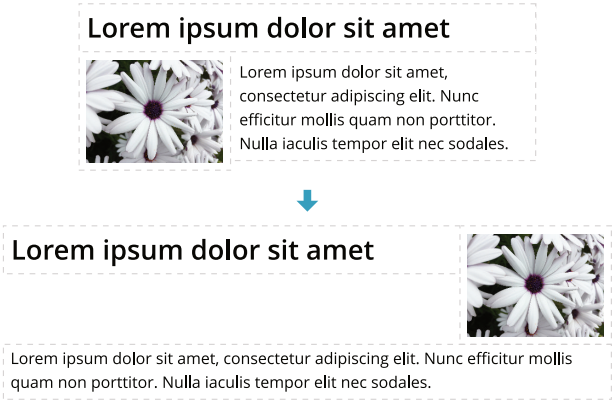
šířku, která se rovná šířce, kterou má při breakpointu. Záměrem je pokud možno nechat prvky uspořádány tak, jak je uživatel „poskládal“ v administraci. V případě potřeby se přeuspořádají podle následujících pravidel:

- Blok nesmí být užší, než je jeho výchozí šířka.
- Zalamování probíhá od hierarchicky nejvyšších bloků (prvků s nejmenším stupněm v DOM stromu).
- Pokud se prvky na řádek nevejdou (součet výchozích šířek je větší, bež šířka řádku), poslední blok na tomto řádku se zalomí na další řádek. Zde se zobrazí jako první v pořadí.
- Volné místo na řádku se rozdělí mezi prvky v poměru výchozíh šířek. Dohromady prvky na řádku zabírají 100 % šířky.
- V případě, že je na řádku již jen jeden blok, začíná se zalamovat obsah uvnitř tohoto bloku.

4.1.2.1 „Zpětné zalamování“ obsahu

V některých situacích mohou hlavní obsahové bloky mít až 1,5násobek své limitní šířky. To může v některých případech znamenat, že se na řádek vejde více prvků, než ve výchozím uspořádání.

To samo o sobě není špatně. Při nevhodně strukturovaném obsahu ale mohou vznikat podobné situace, jako na obrázku 4.3. Z tohoto důvodu není takové chování v návrhu povoleno.



Obrázek 4.3: Ukázka „zpětného zalamování“ bloků

4.1.3 Navigace

Pro horizontální menu jsem zvolil provedení Priority+ (viz 3.3.1), neboť dobře funguje a zároveň umožňuje menu ponechat na původním místě.

Obrázek 4.4: Návrh horizontálního menu pro malé šířky

V případě vertikální navigace je její chování závislé na počtu položek:

- Pokud je počet položek menší, nebo roven 5, vzhled menu se nemění
- Při více než 5 položkách se menu zmenší do rozklikávacího proužku podle obrázku 4.5

Toto dvojí chování při rozumném počtu odkazů ponechá položky přímo viditelné a tedy přehlednější. Pro dlouhá menu už je potřeba zredukovat výšku, proto se položky schovají.

Všechny druhy zobrazení podmenu (přímo, po kliku, po najetí) jsou na malých obrazovkách sjednoceny – menu se zobrazí vždy po kliknutí. Tento způsob je pro dotyková zařízení nejvhodnější, neboť na dotykových zařízeních neexistuje stav „najetí myší“.

4.1.4 Fixní lišta

Fixní lišta v horní části stránky je velmi používaným prvkem na mnoha stránkách. Tato lišta je stále zobrazena i při posouvání stránky a obsahuje nejdůleži-

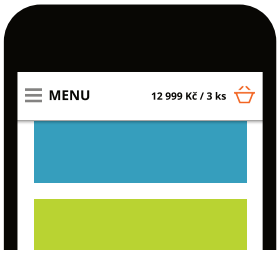
Obrázek 4.5: Návrh vertikálního menu pro malé obrazovky

tější prvky jako logo (s odkazem na homepage), menu či košík. Na Webgarden se tato lišta zobrazí, pokud je splněna alespoň jedna z následujících podmínek:

Na stránce je košík Pokud je v některém ze statických bloků vložen košík.

Navigace V rámci hlavičky a postranních sloupců je vloženo pouze jedno menu, případně je jich více, ale pouze jedno v hlavičce.

Abych předešel situaci, že lišta překryje obsah stránky, či naruší design webu, zobrazí se až při odscrollování stránky pod hlavičku.



Obrázek 4.6: Fixní lišta s menu a košíkem

4.1.5 Typografie

Na Webgarden si uživatel pro svůj web vybírá z velkého množství grafických šablon. Veškerá typografie (i vzhled obecně) je tedy řešena v těchto šablonách a její pojetí se může pro každou šablonu velmi lišit. Proto jsem sepsal několik základních pravidel, kterými by se šablony vytvořené pro responzivní Webgarden měly řídit.

- Základní velikost textu je dobře čitelná, v rozmezí 14 až 18 px (lze přizpůsobit podle použitého fontu).
- Nadpisy jsou dostatečně odlišené od běžného textu. Stejně tak jsou jednotlivé velikosti nadpisů dobře odlišeny od sebe navzájem.
- Na menších obrazovkách se stupeň písma nadpisů zmenšuje, u běžného textu se nemění.
- Výška řádku je zvolena pro co nejlepší čitelnost textu (typicky okolo 140 %). U nadpisů je výška řádku nižší.
- Rozestupy mezi prvky mají velikost výšky řádku (případně jejích násobků).

4.2 Prostředí administrace

Na základě výsledků expertní analýzy a testování s uživatelem jsem rozdělil zjištěné problémy na několik základních skupin:

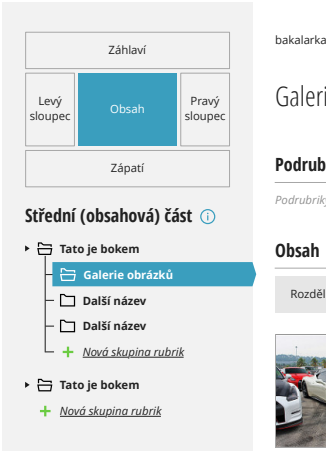
- Nepochopení rozdělení stránky na hlavičku, patičku, postranní sloupce, atd.
- Není jasné, co je součást obsahu stránky a co jen obsah administrace (např. název rubriky).
- Neintuitivní manipulace s obsahem (přesouvání, změna velikosti, uspořádání)
- Špatná přehlednost nastavení. Možnosti schované v záložkách uživatelé často neobjeví.
- Nejasná terminologie, špatná vizualizace – není jasné, co která akce provede či jak bude vypadat výsledek.

Návrhy na řešení těchto problémů jsem shrnul v následujících kapitolách. Navržené rozhraní jsem poté otestoval expertní analýzou.

4.2.1 Levý sloupec

Při testování uživatelé často vůbec nepochopili rozdělení stránky na jednotlivé základní části. Tím pádem například přidávali obsah vícekrát na různá místa.

Proto jsem místo slovního výčtu zvolil grafické zpracování, které lépe ilustruje, jak jsou bloky na stránce ve skutečnosti uspořádány. K dispozici je také ikona nápovědy, která uživatelům v případě potřeby poskytne podrobnější informace.

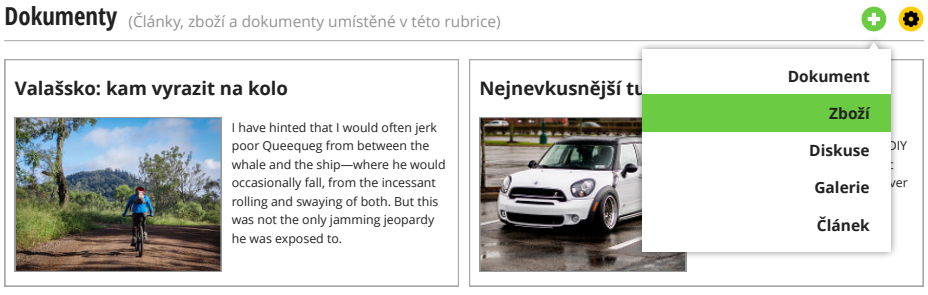


Obrázek 4.7: Znázornění struktury stránky v levém sloupci administrace

4.2.2 Detail rubriky

Detail rubriky jsem se pokusil lépe strukturovat, aby bylo na první pohled zřejmé oddělení jednotlivých typů obsahu (podrubriky, uživatelský obsah, dokumenty). Stránka je proto rozdělena na tři vizuálně oddělené sekce, z nichž každá má svůj nadpis. Tyto sekce jsou zobrazeny vždy, i pokud jsou prázdné.

Tlačítko „+“ (přidat) v horní části stránky jsem zrušil a nahradil samostatným tlačítkem pro každou z částí. Toto řešení dává větší logiku, neboť se obsah přidává právě tam, kde se bude i nacházet.



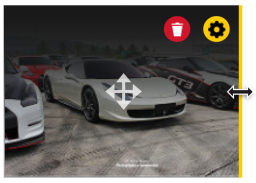
Obrázek 4.8: Tlačítko pro přidání obsahu do sekce

Název rubriky je výrazně oddělen od ostatního obsahu. Také u něj při přejmenování přibyla možnost zobrazení/skrytí na stránce. Doposud bylo toto nastavení umístěno pouze odděleně na kartě „Nastavení“.

Celkový pohled na detail rubriky je zobrazen na obrázku B.1.

4.2.3 Manipulace s obsahem

Při najetí myší nad prvek je tento prvek překryt poloprůhledným přechodem, aby bylo uživateli zřejmé, jaký obsah upravuje. Uprostřed je zobrazen čtyřsměrný kurzor naznačující možnost přesouvání. Podobně jako v operačním systému, při najetí myší poblíž hrany prvku se kurzor změní na šipku a šumožní změnu velikosti. Tlačítka pro smazání a editaci zůstala na svém místě vpravo nahoře.



Obrázek 4.9: Obsahový prvek po najetí myši

Přesunulo se tlačítko pro přidání nového prvku, neboť nedávalo příliš velký smysl (uvnitř jednoho prvku se přidával jiný). Nově se zobrazí mezi dvěma prvky po najetí myši.

Obrázek 4.10: Tlačítko pro přidání obsahu mezi dva prvky

Přesouvání obsahu pod sebe

V systému Webgarden je obsah chápán po řádcích a jednotlivé obsahové bloky se tedy skládají vedle sebe. Pro umístění pod sebe je sice možné využít prvek „Sloupec“, ale ten je komplikovaný a uživatelé jej nechápou. Proto jsem se snažil přidat jednodušší způsob, jak umístění prvků pod sebe umožnit. Při přesouvání prvků pod sebe se tedy administrace chová takto:

- Přesouváme blok B pod blok A
- Pokud má A šířku 100 %, B se pouze přesune pod něj, tak jako doposud
- V opačném případě se na místě A vytvoří nový prvek „Sloupec“ o šířce A a počtu buňek 1
- Do tohoto sloupce se pod sebe umístí A a B, oba se šířkou 100 %

Obrázek 4.11: Ukázka chování při umisťování prvků pod sebe

- Pokud někdy ve sloupci zbude jen jeden prvek (např. když ostatní odstraníme), sloupec se zruší a na jeho místě zůstane jen samotný zbylý blok.

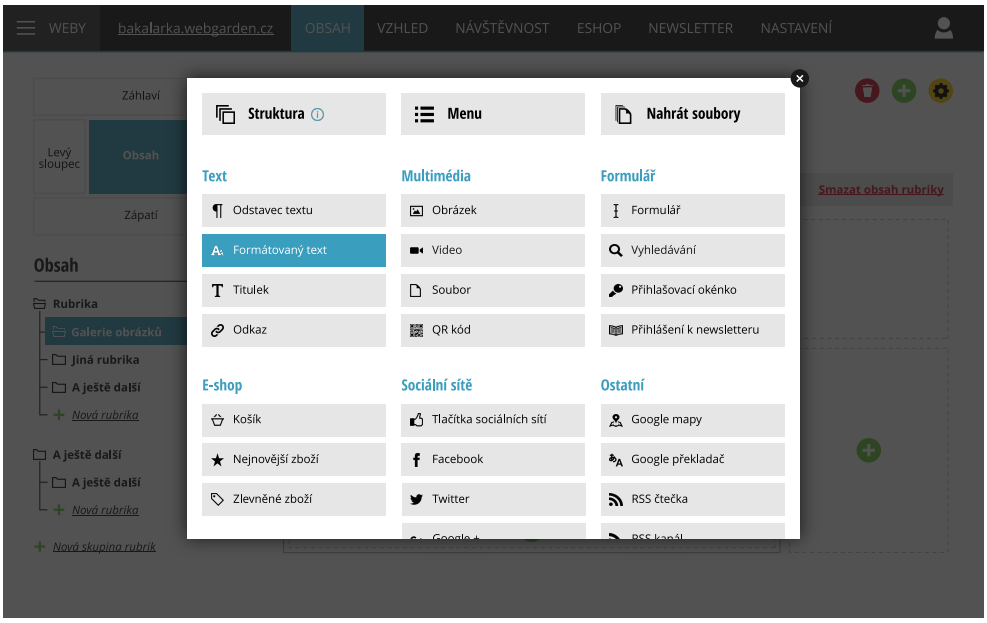
4.2.4 Přidávání a úprava obsahu

Seznam prvků ve vyskakovacím okně při přidávání obsahu jsem přeuspořádal a logicky rozčlenil do skupin. Také jsem odstranil záložky, vše je nyní na jediné „straně“. Samotné vyskakovací okno je pozicováno vždy na střed stránky, aby nedocházelo k jeho zobrazení mimo obrazovku.

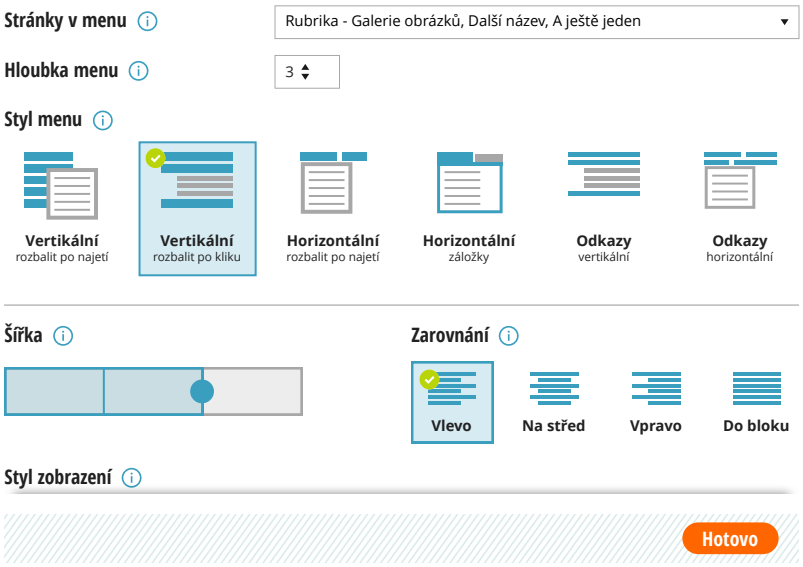
4.2.5 Vysvětlivky a vizualizace

U editace jednotlivých obsahových prvků jsem kladl důraz především na lepší vizuální podání. Volby jako zarovnání, šířka, počet buněk nebo zvýraznění jsou graficky zpracovány, aby uživatel věděl, co dělá. Stejně tak jednotlivé typy menu či velikosti titulků mají svou názornou ikonu.

V případě, že uživatel neporozumí názvu nastavení, je u většiny voleb umístěna ikona, která po kliku zobrazí jednoduchý popis nastavení a vysvětlí uživateli, k čemu se volba používá.



Obrázek 4.12: Vyskakovací okno pro přidání nového obsahu



Obrázek 4.13: Ukázka grafického zpracování jednotlivých typů menu, šířky či zarovnání

4.2.6 Testování navrženého rozhraní

Navržené rozhraní jsem následně otestoval. V následujícím seznamu je seznam problémů nalezených při expertní analýze a uživatelském testování. Ke každému problému je doplněno navržené řešení.

Sloupec - nepochopení principu Nová možnost při spořádání obsahu, viz obrázek 4.11

Chybí zpět + snaha ovládání klávesnicí Namapování některých akcí na tlačítka klávesnice

Nepřehlednost obsahu, záložky Záložky byly zrušeny. Obsah je lépe roztříděn a strukturován.

Nejasná terminologie, špatná vizualizace Problematické volby doplněny ikonami. U většiny popisků přibyly vysvětlivky.

Neintuitivní manipulace s obsahem Přepracovaná práce s obsahem – jasnější ikony, změna kurzoru.

Vyskakovací okno mimo obrazovku Okno má fixní pozici ve středu stránky. Na prvky mimo okno nelze klikat.

Tlačítko „+“ má nelogické chování Tlačítka jsou nyní umístěna tam, kde se obsah skutečně přidává.

Rozdělení stránky – nepochopeno Vizuálně znázorněné obsahové bloky. Rubriky mají nyní podobu stromu.

Neintuitivní vytváření a přesouvání rubrik V detailu rubriky je nyní možnost přidat podrubriku přímo tlačítkem „+“. Tlačítka v levém sloupci jsou lépe viditelná.

FB LikeBox – nastavení stránky Nastavení Facebookové stránky je nyní možné přímo ve vyskakovacím okně.

Vnímání nadpisu jako části webu Byla zpřehledněna podoba detailu rubriky.

Kapitola 5

Implementace

5.1 HTML

Základní logiku uspořádání stránky jsem musel kvůli zachování zpětné kompatibility nechat stejnou. Změny v HTML struktuře proto spočívají především v masivním zjednodušení kódu.

Kladl jsem důraz také na sémantiku – např. každé menu je nyní řešeno jako číslovaný seznam, texty jsou v odstavcích, atd. V neposlední řadě jsem změnil logiku tříd a vytvořil pravidla pro jejich názvosloví. Třídy mají jasně daný význam a hierarchii, což by mělo umožnit jednodušší stylování bez několikanásobných deklarací a dlouhých selektorů.

5.1.1 Názvosloví tříd

Třídy jsem rozdělil do tří skupin. Každá skupina má svá pravidla pro vytváření názvů tříd, takže je na první pohled jasné, o jaký typ třídy se jedná (případně se kterou jinou třídou souvisí).

Obecná Třída, která nemá žádnou nadřazenou třídu, ani žádnou jinou třídu nerozšiřuje – např. *.area*, *.group*, *.block*, *.content*, *.menu*, atd. Označuje prvek nebo skupinu prvků s určitým významem, kdy jednotlivé prvky mohou být upřesněny pomocí jiných tříd. Název třídy sestává vždy z jednoho slova a začíná malým písmenem.

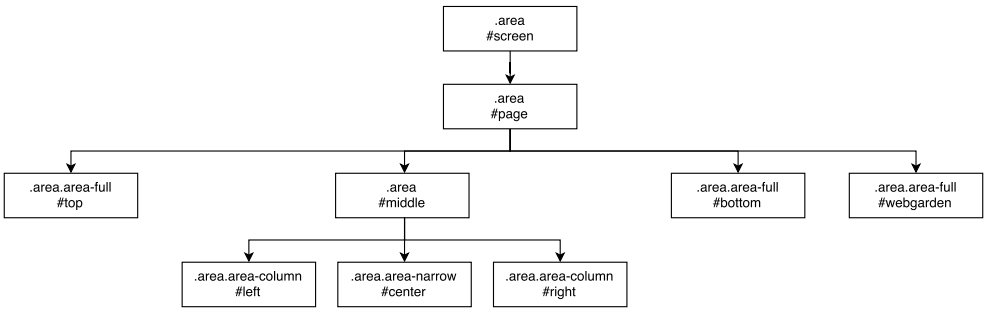
Podtřída Třída, která se v kódu nevyskytuje samostatně, ale vždy pouze jako potomek jiné třídy – např. *.detailHeader*, *.productImage*, atd. Označuje prvky, které jsou součástí nějakého většího obsahového bloku. K názvu rodičovské třídy vždy přidá jedno slovo začínající velkým písmenem.

Rozšiřující Rozšiřující třídy upřesňují význam jiných tříd, jedná se například o třídy *.menu-vertical*, *.block-extra*, atd. Prvku je společně s touto třídou přiřazena i třída základní, dostaneme potom například *.menu.menu-*

vertical. Rozšiřující třída přidává k názvu třídy, z které vychází, jedno slovo začínající malým písmenem a oddělené spojovníkem.

5.1.2 Struktura

Odstranil jsem třídu *.group* (resp. její podtřídy *.i*, *.ii*, *.g_t*, atd.). Tato struktura již není potřeba, neboť problémy, kvůli kterým byla do HTML kódu implementována (zaoblené rohy, správné odsazování), se již dnes dají řešit jednoduše pomocí CSS. Základní struktura stránky zůstala stejná, jen jsem odstranil některé zbytečné prvky (*.row*, *.col*) a změnil názvy tříd a identifikátorů. Layout je znázorněn na obrázku 5.1.



Obrázek 5.1: Nová struktura základního layoutu stránky

Jelikož každý ze základních bloků se na stránce může vyskytovat maximálně jednou, má svůj identifikátor. K tomu mají bloky přiřazenou třídu *.area* a vnitřní bloky (listy stromu) ještě rozšiřující třídu podle jejich šířky. U bloku *#center* se šířka mění podle toho, zda jsou přítomny boční sloupce, u ostatních bloků je pevně daná.

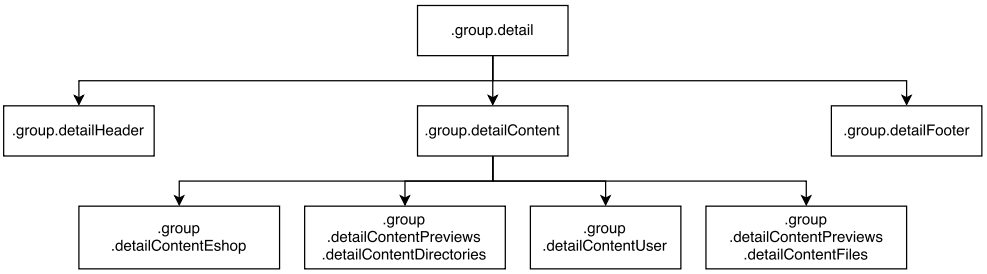
5.1.3 Obsahová sada

U obsahové sady jsem pouze odstranil bloky *.row* a *.col* a přejmenoval třídy podle vytvořeného názvosloví. Struktura zůstala totožná, viz obrázek 5.2.

Stejně je strukturovaný i náhled obsahové sady, jen s rozdílem v pojmenování tříd, kdy *detail* je nahrazeno za *preview*. Každý prvek *.preview* má potom nově specifikovaný typ zobrazení (*.preview-tiled* pro dlaždice, *.preview-list* pro seznam, atd.)

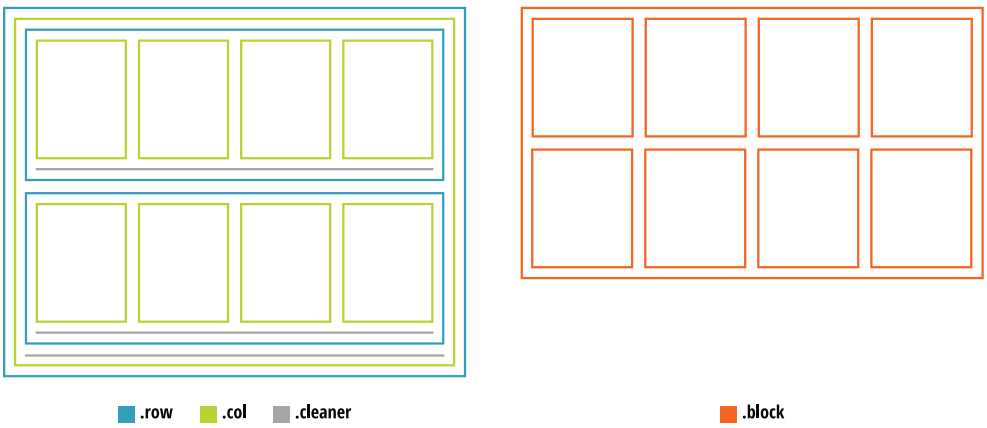
5.1.4 Uživatelský obsah

Uživatelský obsah prošel zásadní změnou. Logiku matic $m \times n$ jsem z důvodu zpětné kompatibility zachoval, nicméně došlo k úplnému odstranění řádků s bloky *.row* a *.col*. Ty jsem nahradil prvkem *.block*.



Obrázek 5.2: Nová struktura obsahové sady

Element *.block* zastává zároveň úlohu řádku i sloupce. Jednotlivé bloky jsou seřazeny v kódu za sebou, řádky jsou vytvořeny nastavením patřičné šířky v CSS. Každý z bloků má třídu *.divide-m*, která určuje počet sloupců uvnitř něj. Třída *.width-n* pak slouží ke stanovení šířky v rámci rodičovského bloku. Toto řešení umožňuje daleko volnější manipulaci s obsahem, což je pro responzivní layout nezbytné. Každý z bloků má nakonec ještě třídu *.width-total-x*, kde x odpovídá procentuální šířce bloku vůči rodičovskému obsahovému sloupci. Rozdíl mezi současnou a novou implementací je znázorněn na obrázku 5.3.



Obrázek 5.3: Porovnání původní a nové struktury matice uživatelského obsahu

Struktura se řídí několika jednoduchými pravidly:

- Každý blok může obsahovat buď další bloky (*.block*), nebo obsah (*.content*)

- Blok může mít neomezené množství potomků třídy *.block*, ale pouze jednoho třídy *.content*

- Nevyšší blok má třídu *.width-0* a může obsahovat pouze bloky (nikoli obsah)

Třída *.content* obsahuje jednotlivé obsahové prvky. Obsahovým prvkem se rozumí každý ze seznamu prvků, které lze přidat z administrace. Kromě třídy *.content* má tento prvek ještě další třídy, které specifikují typ obsahu (například pro odstavec textu je to *.text.text-paragraph*). Díky těmto třídám je podstatně jednodušší v CSS zacílit konkrétní obsahový prvek. To umožní jednodušší tvorbu šablon bez zbytečně dlouhých selektorů. Uvnitř prvku *.content* může být obsah libovolný.

5.2 CSS

5.2.1 Výběr preprocesoru

Pro stylovací část projektu jsem se rozhodl zvolit některý z CSS preprocesorů. Preprocesory jsou jazyky postavené nad CSS, ke kterému přidávají nové možnosti. Často se snaží poskytnout některé vlastnosti známé z jiných programovacích jazyků jako proměnné, funkce, objekty, atd. [10]

Nejznámějšími a nejrozšířenějšími preprocesory jsou Less, SASS a Stylus. Všechny nabízí již výše zmíněné vlastnosti, dále vnořování selektorů, vnořené media queries, mixiny, více či méně pokročilé matematické operace nebo třeba práci s barvami.

Jsou mezi nimi ovšem i celkem podstatné rozdíly. Ty nejdůležitější jsem shrnul v následujícím přehledu:

Less Zachovává deklarativní povahu CSS. Obsahuje velmi dobrou práci s vnořováním a dynamickými proměnnými. Naopak nabízí velmi omezené podmínky a cykly.

SASS Více se přibližuje imperativním jazykům. Díky tomu umožňuje vytvářet pokročilé cykly či podmínky.

Stylus Svým pojetím je velmi podobný SASS. Oproti němu poskytuje o trochu více možností. Na druhou stranu, kvůli své malé rozšířenosti pro Stylus existuje podstatně méně knihoven.

Nakonec jsem se rozhdol použít SASS. Proti Less hovoří absence cyklů, které se pro realizaci navrženého layoutu velmi dobře hodí. Pro Stylus zase neexistují některé knihovny, jako například Modular scale, kterou jsem použil pro responzivní typografii.

5.2.2 Flexbox

Pro řešení responzivního layoutu jsem použil modul CSS Flexible Box Layout Module, zkráceně flexbox. Jedná se o relativně nový modul, který umožňuje jednoduše uspořádat prvky.

Flexbox umožňuje nastavit prvkům na stránce takové vlastnosti, že se obsah sám přizpůsobí, jak je to v danou situaci nejvýhodnější. Při správném použití flexboxu je možné vytvářet responzivní struktury, aniž by bylo třeba přesně specifikovat breakpointy.

Toho jsem využil například u náhledu zboží. Protože nevím, v jak širokém bloku se náhled zobrazí a kolik náhledů bude vedle sebe, bylo by velmi obtížné (a datově náročné) konkrétně specifikovat chování pro každý případ. S použitím flexboxu stačilo několik deklarací pro každou část náhledu:

```
.product
  display: flex
  flex-grow: 1
  flex-flow: row wrap
  justify-content: center

  .productImage
    flex-basis: $width-productContent
    flex-grow: 0

  .productData
    display: flex
    flex-basis: $width-productContent
    flex-grow: 1
    flex-flow: column nowrap
    justify-content: space-between

  .productDataInfo
    display: flex
    flex-flow: row wrap

    > div
      flex-basis: $width-productContent
      flex-grow: 1

  .productDataSummary
    display: flex
    flex-flow: row wrap
    justify-content: space-between
    align-items: flex-end
```

```
> div
  flex-basis: $width-productContent * (2/3)
  flex-grow: 1
```

Navzdory tomu, že podle mnohých názorů flexbox není vhodný ke stylování celkových layoutů a hodí se spíše pro menší moduly (jako v předchozím případě), jsem se rozhodl pro tento účel flexbox použít.

Hlavním důvodem je možnost roztáhnout prvek, pokud je kolem něj volné místo. Tato vlastnost je velmi užitečná při implementaci responzivního uživatelského obsahu podle návrhu v kapitole 4.1.2.

Dále je také možné zobrazit prvky v jiném pořadí, než v jakém jsou seřazeny v HTML kódu. To jsem využil při přemístění levého sloupce pod obsahovou část na úzkých obrazovkách.

5.2.3 Responzivní bloky

Základním parametrem navrženého responzivního layoutu je šířka bloku při breakpointu. K určení této šířky potřebuji znát hodnotu breakpointu, šířku obsahového sloupce a část, kterou zabírá blok v tomto sloupci. Breakpoint je stanovený, obsahový sloupec může mít 4 šířky (rozlišeny třídami *.area-full*, *.area-wide*, *.area-narrow* a *.area-column*). Zbývá tedy zjistit (relativní) šířku bloku.

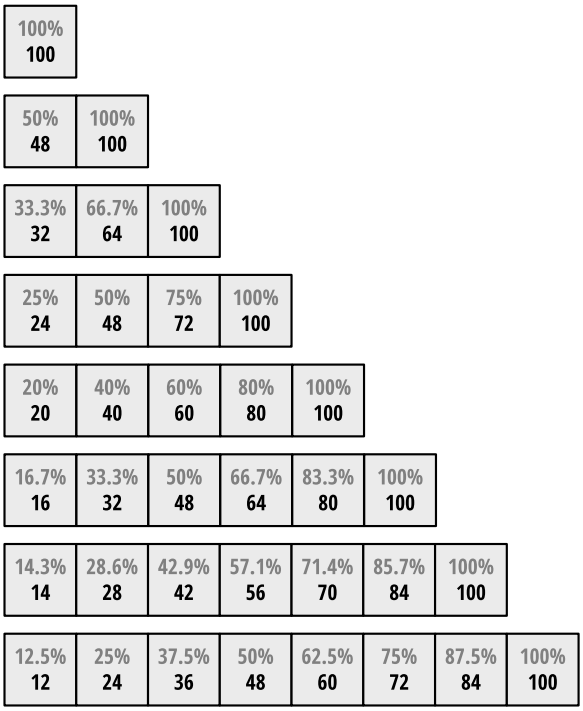
To lze udělat poměrně jednoduše. U každého bloku známe jeho šířku vzhledem k rodiči, takže stačí rekurzivně spočítat tuto šířku a postup opakovat pro každý nadřazený blok. V pseudojazyce vypadá zápis takto:

```
function getWidth(block) {
    if (parent)
        return getWidth(parent) * (block.width / parent.divide);
    else
        return 1;
}
```

Možných výsledků je ale příliš mnoho na to, aby se pro každý z nich dalo nadefinovat chování v CSS. Proto jsem vytvořil 19 tříd, do kterých blok může spadat. Tyto třídy pokrývají všechny možné šířky bloků v situaci, kdy má rodičovský blok šířku 100 %. Čísla tříd jsem stanovil podle následujících pravidel:

- Musí to být celé číslo menší, než je skutečná šířka prvku
- Stejně široké bloky mají stejnou třídu
- Pokud má blok šířku, která je násobkem šířky jiného bloku, musí být i hodnoty jejich tříd ve stejném poměru

Po aplikaci těchto pravidel vzniklo zmíněných 19 tříd. Jejich grafické znázornění je vidět na obrázku 5.4. Každému bloku s procentuální šířkou w se přiřadí třída *.width-total-x*, kde x je nejbližší možné číslo, které je menší než w.



Obrázek 5.4: Rozdělení tříd uživatských bloků podle procentuální šířky vůči obsahovému sloupci

Samotný SASS kód vypadá takto. Mixin *responsive-blocks* se volá pětkrát – pro každou šířku sloupce jednou při šířce, ve které se začne zalamovat obsah. Pro postranní sloupce ve volá dvakrát, neboť jejich chování se liši v závislosti na tom, zda se jedná o dvoj- či třísloupcový layout.

```
=responsive-blocks($width)

    @for $i from 1 through 8
        .width-#{$i}
            flex-grow: $i

        @for $j from 1 to $i
            .divide-#{$i} > .width-#{$j}
                min-width: 100% * ( $j / $i )
```

```
    @each $i in $responsive-widths
        .width-total-#{$i}
            flex-basis: $width * ($i / 100)


// full-width areas
@media screen and (max-width: $breakpoint-full)
    .area-full
        +responsive-blocks($breakpoint-full)

// center wide (with one column)
@media screen and (max-width: $breakpoint-full)
    .area-wide
        +responsive-blocks($breakpoint-wide)

// center narrow (with two columns)
@media screen and (max-width: $breakpoint-narrow)
    .area-narrow
        +responsive-blocks($breakpoint-narrow)

// columns
@media screen and (max-width: $breakpoint-full)
    .area-wide + .area-column,
    .area-narrow ~ .area-column
        +responsive-blocks($breakpoint-column)

@media screen and (max-width: $breakpoint-wide)
    .area-narrow + .area-column
        +responsive-blocks($breakpoint-column)
```

5.3 XSLT

V XSLT šablonách jsem v drtivé většině případů hlavně změnil HTML kód uvnitř šablony, aby odpovídal návrhu. U šablon vykreslujících matice uživatelského obsahu (*.row* a *.col*) jsem musel udělat změny o něco větší, neboť došlo k výraznější změně logiky uspořádání.

Závěr

Provedl jsem analýzu současného stavu systému Webgarden. Z té vyplynulo, že aktuální řešení není pro implementaci responzivního layoutu vhodné a bude potřeba významnější zásah do struktury stránek. Součástí bylo i stručné porovnání s konkurenčními službami. Druhou částí analýzy bylo otestování uživatelského rozhraní administrace. Provedl jsem expertní analýzu pomocí heuristické evaluace, kterou jsem doplnil testováním s uživateli.

V následující kapitole jsem se věnoval principům responzivního webdesignu. V jejím úvodu jsem se zabýval výhodami responzivního webdesignu oproti jiným metodám. Součástí kapitoly je také několik základních kategorií responzivních layoutů a doporučené praktiky pro interaktivní prvky na stránce. Na konec jsem připojil i část o typografii, která je velmi důležitou součástí webdesignu.

Kapitola Návrh je rozdělena na dvě části – Responzivita a Prostředí administrace. V první části jsem navrhnul univerzální responzivní layout pro klientské weby Webgarden. Tento layout splňuje zadané požadavky a funguje automaticky pro jakýkoliv obsah webu. Prostředí administrace potom obsahuje nové UI administrace navržené podle výsledků testování.

Při implementaci jsem se snažil o maximální zjednodušení HTML struktury. Vytvořil jsem názvosloví tříd pro lepší přehlednost kódu. Ke stylování jsem se rozhodl použít preprocesor LESS. Layout jsem vytvořil pomocí nového CSS modulu flexbox (CSS Flexible Box Layout Module).

Výsledkem práce je nová verze šablon pro systém Webgarden. Na SD kartě přiložené k práci je možné spustit ukázkovou verzi stránek, která demonstruje všechny základní typy layoutů. Dále je součástí práce 17 obrazovek navrženého uživatelského prostředí administrace.

Literatura

[1] CSS koutek 3 - Hierarchie html elementů. *Webgarden* [online]. 2014, 2014-03-27 [cit. 2016-02-14]. Dostupné z: http://www.webgarden.cz/help/pokrocile/css-koutek-3-hierarchie-html

[2] *Responsive Web Design Best Practices*. UXPin Inc., 2015.

[3] Struktura. *Webgarden* [online]. 2015, 2015-09-18 [cit. 2016-02-14]. Dostupné z: http://www.webgarden.cz/help/tvorime-web/struktura-2

[4] Interesting stats. *HTTP Archive* [online]. 2016 [cit. 2016-05-04]. Dostupné z: http://httparchive.org/interesting.php?a=All&l=Apr%2015%202016&s=All

[5] V březnu rostl počet uživatelů streamovaného obsahu z mobilních zařízení. *SPIR* [online]. 2016, 2016-04-19 [cit. 2016-04-28]. Dostupné z: http://www.netmonitor.cz/v-breznu-rostl-pocet-uzivatelu-streamovaneho-obsahu-z-mobilnich-zarizeni

[6] Bauma, F.: *Testování online nástrojů pro tvorbu internetových stránek*. Bakalářská práce, České vysoké učení technické v Praze, Fakulta elektrotechnická, Katedra počítačové techniky a interakce, Praha, 2013.

[7] Kasík, P.: Česká firma provozuje 22 milionů webů. Nový systém uznal, že lidé nečtou. *iDNES.cz* [online]. 2016, 2016-03-08 [cit. 2016-03-14]. Dostupné z: http://technet.idnes.cz/novy-editor-webnode-rozhovor-cesky-projekt-tvorba-webu-p2f-/sw_internet.aspx?c=A160205_153005_sw_internet_pka

[8] Křovák, J.: Webová typografie rozhoduje o tom, zda si lidé vůbec přečtou vaši stránku. *eBRÁNA* [online]. 2013 [cit. 2016-05-04]. Dostupné z: http://ebrana.cz/magazin/webova-typografie-rozhoduje-o-tom-zda-si-lide-vubec-prectou-vasi-stranku

[9] LePage, P.: Responsive web design patterns. *Google Developers* [online] [cit. 2016-04-29]. Dostupné z: https://developers.google.com/web/fundamentals/design-and-ui/responsive/patterns

[10] Michálek, M.: Průvodce CSS preprocesory: co a jak? *Vzhůru dolů* [online]. 2014 [cit. 2016-05-09]. Dostupné z: http://www.vzhurudolu.cz/blog/12-css-preprocesory-1

[11] Nielsen, J.: 10 Usability Heuristics for User Interface Design. *Nielsen Norman Group* [online]. 2005, 1995-01-01 [cit. 2016-04-24], ISSN 1548-5552. Dostupné z: https://www.nngroup.com/articles/ten-usability-heuristics/

[12] Pecina, M.: Typografie na webu. *Typomil* [online] [cit. 2016-05-02]. Dostupné z: http://typomil.com/typografie-na-webu.htm

[13] Snozová, M.: Heuristická analýza. *Inflow* [online]. 2013, 2013-01-09 [cit. 2016-04-24]. Dostupné z: http://www.inflow.cz/heuristicka-analyza

[14] Urbánek, P.: Monitory s vysokým rozlišením: Proč už je u PC dávno nemáme? *PCTuning* [online]. 2014 [cit. 2016-05-03]. Dostupné z: http://pctuning.tyden.cz/hardware/monitory-lcd-panely/30549-monitory-s-vysokym-rozlisenim-proc-uz-je-u-pc-davno-nemame?start=3

[15] Work, S.: How Loading Time Affects Your Bottom Line. *Kissmetrics* [online]. 2011 [cit. 2016-05-04]. Dostupné z: https://blog.kissmetrics.com/loading-time/

Příloha A

Testování uživatelského rozhraní

A.1 Testování s uživatelem 1

Tabulka A.1: Seznam problémů zaznamenaných při testování s uživatelem 1

Kritické	Rozdělení stránky na záhlaví, obsah, atd. – absolutně nepochopeno.
Kritické	Neschopnost uspořádat obsah podle představ. Nepochopení sloupce, snaha simulovat jej přesouváním prvků pod sebe.
Závažné	Snažila se vrátit provedené změny tlačítkem „Zpět“ v prohlížeči.
Závažné	Kořen menu – nevěděla, co to znamená. Nejasná souvislost s rubrikami, snažila se je přejmenovat přímo v menu.
Závažné	Název rubriky – nadpis vnímala jako součást obsahu webu (může, ale nemusí být pravda).
Závažné	Tlačítko „+“ se nechová předvídatelně. Horní tlačítko při kliku místo rozbalení možností přidá nový dokument. Tlačítko u konkrétního prvku přidá nový obsah před prvek, atd.
Závažné	Uspořádání a hranice prvků nejsou jasně naznačené, uživatelka se v administraci nevyznala a byla zmatená.
Závažné	Přesouvání a zvětšování/zmenšování prvků je nejasně naznačeno. Byla nutná rada moderátora.

Mírné	Šířka – není jasné, vůči čemu se počítá. Nastavovala náhodné hodnoty.
Mírné	Vytváření a přesouvání rubrik – neintuitivní.
Mírné	Progress bar je špatně viditelný. Při nahrávání většího souboru (trvá dlouho) si myslela, že se soubor nepodařilo nahrát.
Mírné	Náhodně zvolila možnost „Zobrazit v ukázce“. Obsah se potom v rubrice vůbec nezobrazil a nevěděla proč.
Mírné	Při editaci prvku si nejdříve vůbec nevšimnula záložky „Rozšířené“.
Mírné	Přidávání prvků – prvky jsou nepřehledně seřazené, Facebook LikeBox (na druhé záložce) vůbec nemohla najít.
Mírné	Facebook LikeBox nefungoval bez zadání FB stránky. Toto nastavení uživatelka nemohla najít.

A.2 Testování s uživatelem 2

Tabulka A.2: Seznam problémů zaznamenaných při testování s uživatelem 2

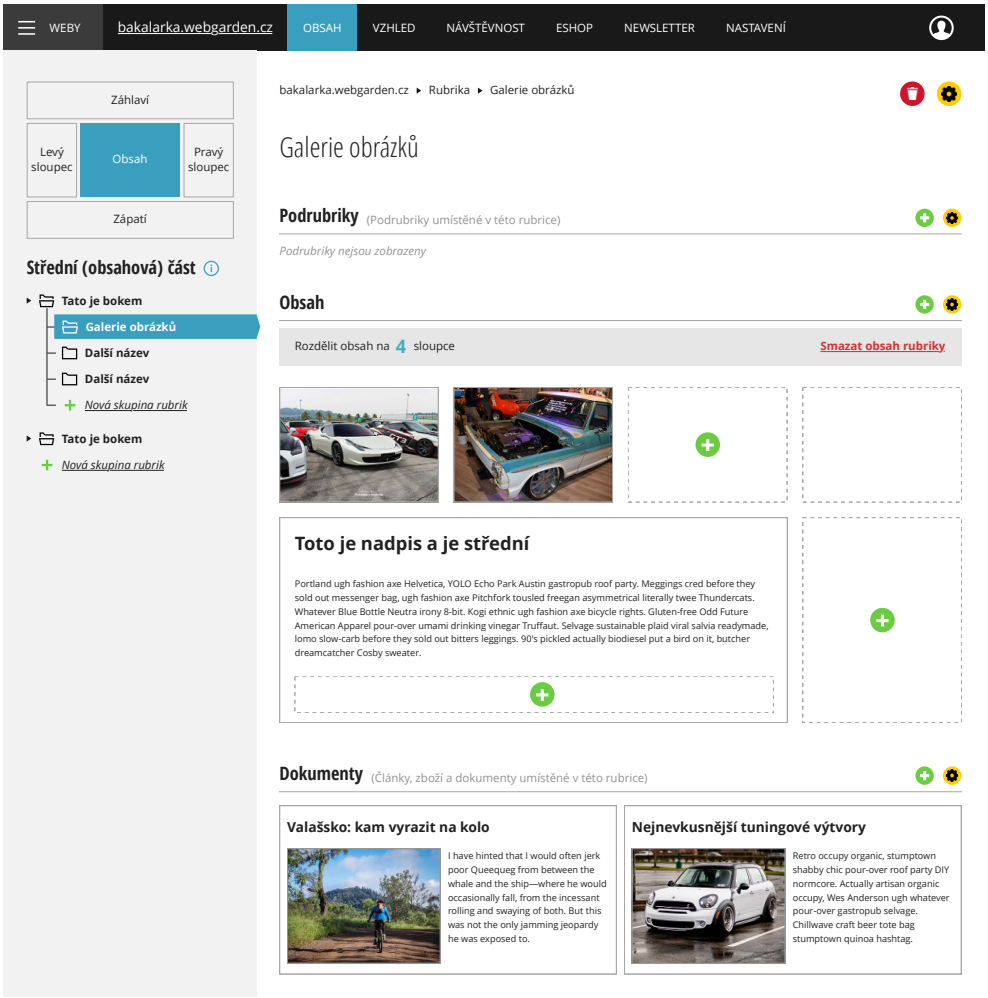
Kritické	Rozdělení stránky na záhlaví, obsah, atd. – absolutně nepochopeno.
Kritické	Neschopnost uspořádat obsah podle představ. Nepochopení sloupce, snaha simulovat jej přesouváním prvků pod sebe.
Kritické	Problém s pochopením rubrik a struktury webu.
Závažné	Snažila se vrátit provedené změny tlačítkem „Zpět“ v prohlížeči.
Závažné	Kořen menu – neví, co to znamená. Nejasná souvislost s rubrikami, snaží se je přejmenovat přímo v menu.
Závažné	Název rubriky – nadpis vnímá jako součást obsahu webu (může, ale nemusí být pravda).

Závažné	Nejasné názvosloví – „zarovnání“ vnímala jako zarovnání prvku, nikoliv textu. Nepochopila pojmy jako „kořen menu“, „zvýraznění“, atd.
Závažné	Tlačítko „+“ u jednotlivých prvků přidává obsah před tento prvek. Nový obsah se většinou roztáhne na 100 % šířky a rozbije strukturu stránky.
Závažné	Vytváření a přesouvání rubrik – neintuitivní.
Závažné	Facebook LikeBox nefungoval bez zadání FB stránky. Toto nastavení uživatelka nemohla najít.
Mírné	Snažila se používat klávesy Esc a Backspace pro ovládání administrace.
Mírné	Při editaci prvku si nejdříve vůbec nevšimnula záložky „Rozšířené“.
Mírné	Přidávání prvků – prvky jsou nepřehledně seřazené, Facebook LikeBox (na druhé záložce) vůbec nemohla najít.
Mírné	Vyskakovací okno se objevilo mimo obrazovku.

Příloha B

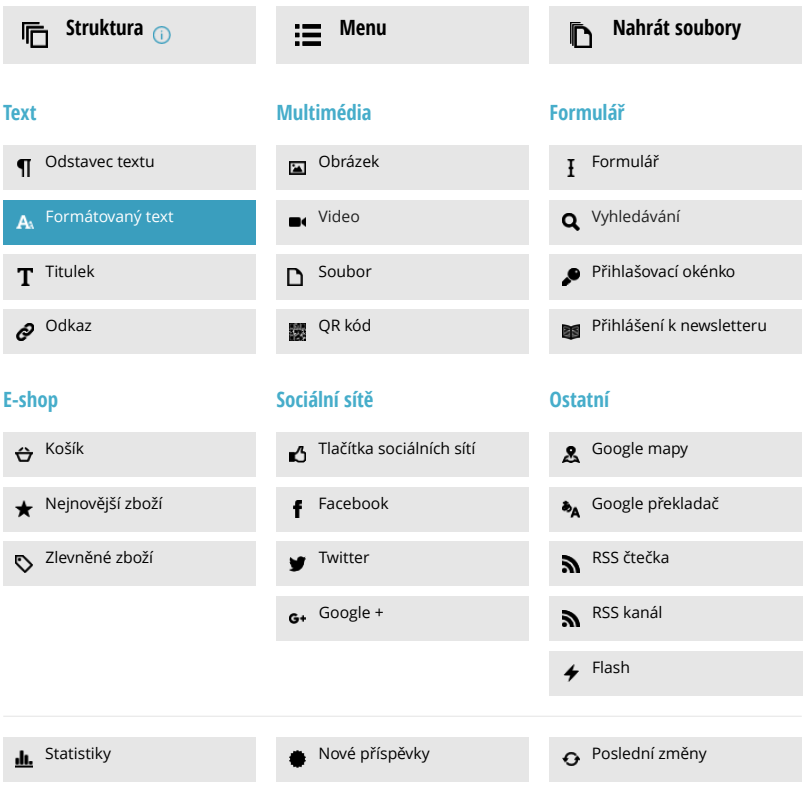
Návrhy uživatelského rozhraní administrace

B.1 Administrace – záložka obsah



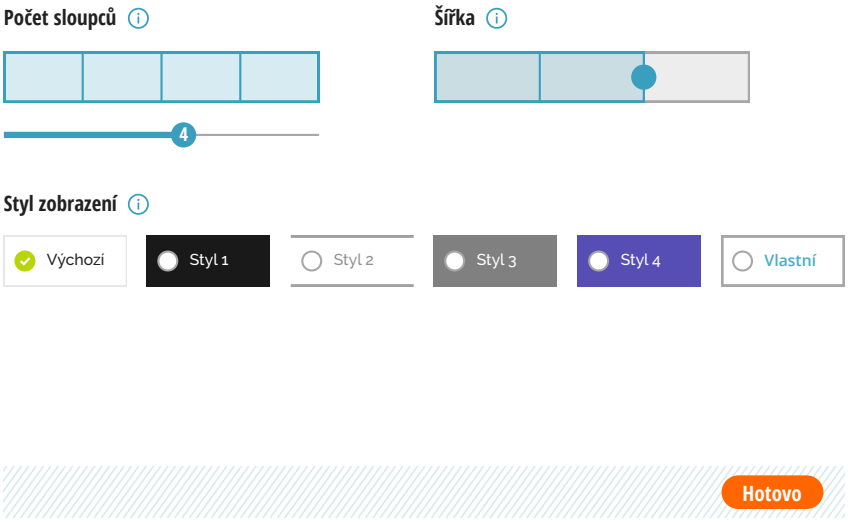
Obrázek B.1: Uživatelské rozhraní adminsitrace pro správu obsahu

B.2 Vyskakovací okna



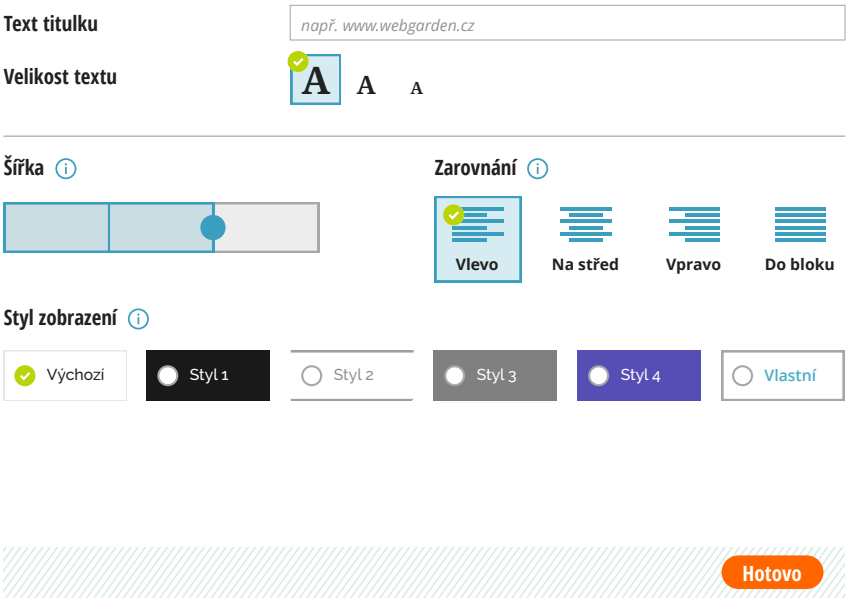
Obrázek B.2: Vyskakovací okno s přehledem obsahových prvků

B.2.1 Sloupec

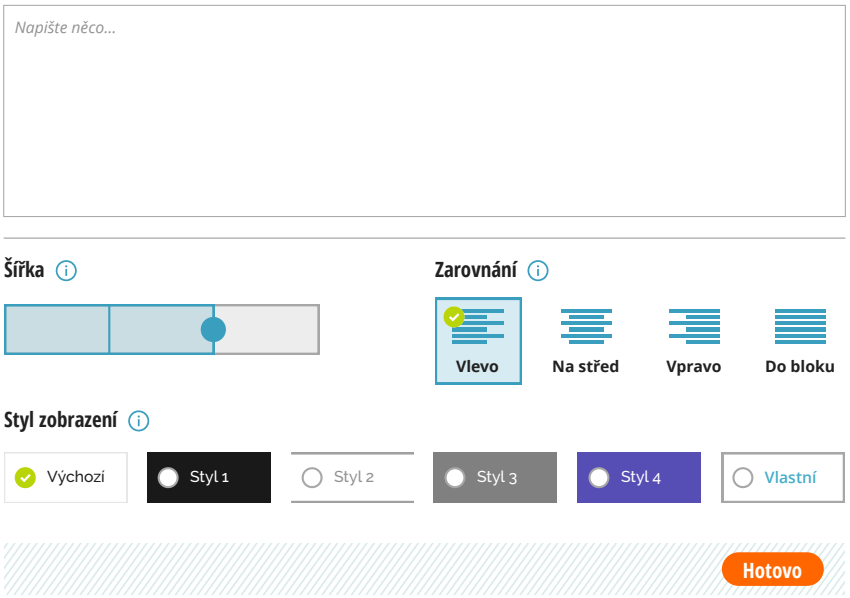


Obrázek B.3: Vyskakovací okno k vytvoření sloupce

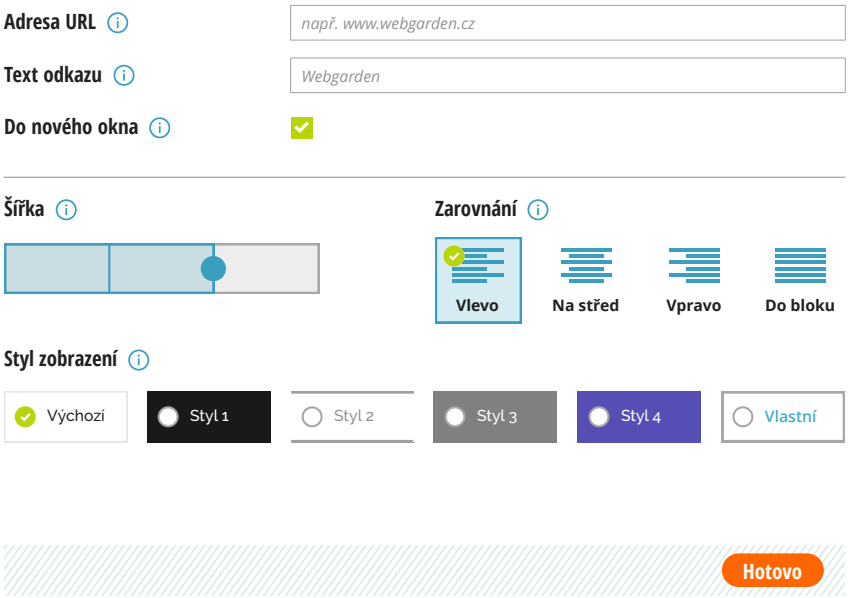
B.2.2 Text



Obrázek B.4: Vyskakovací okno k vložení titulku

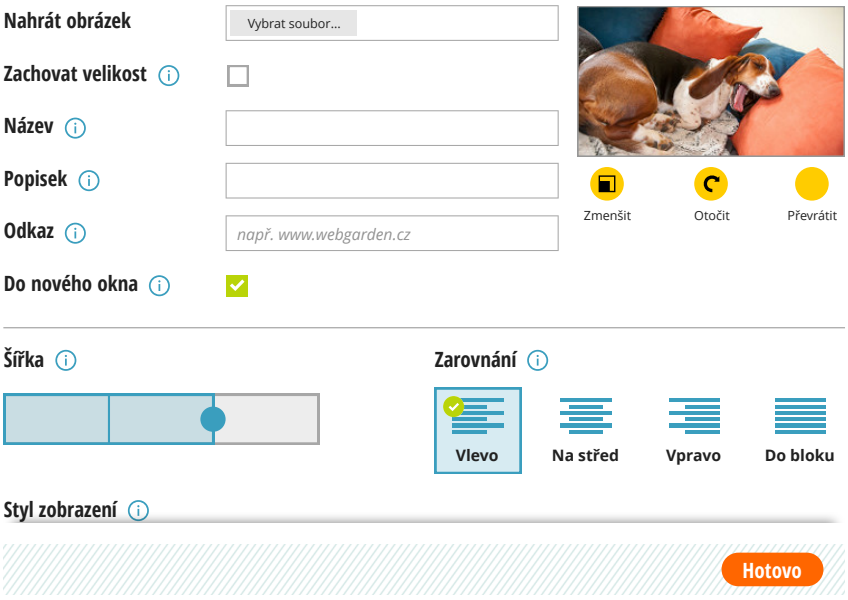


Obrázek B.5: Vyskakovací okno k vložení odstavce textu

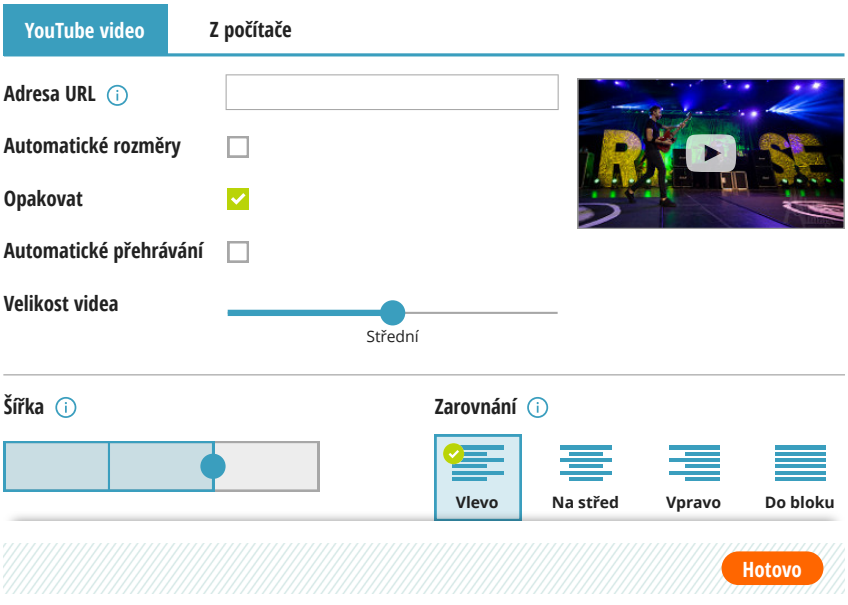


Obrázek B.6: Vyskakovací okno k vložení odkazu

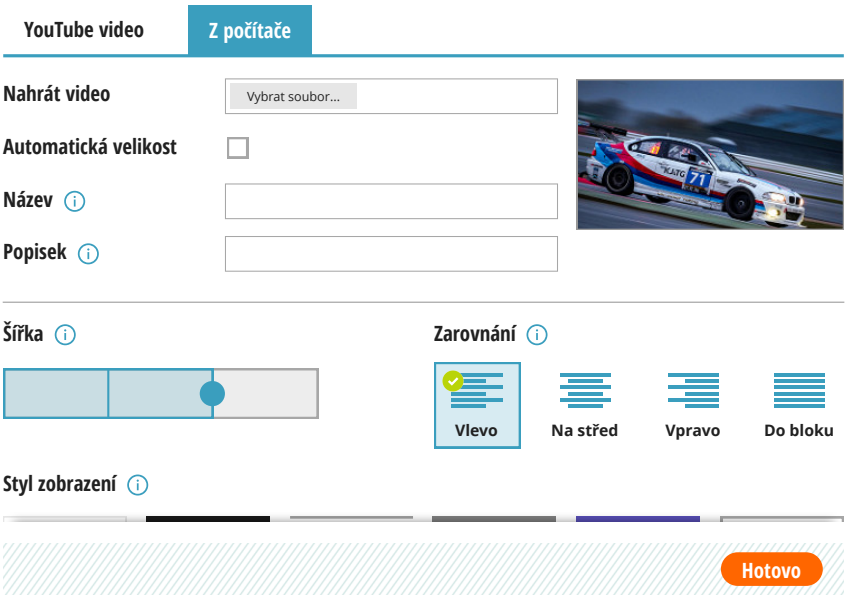
B.2.3 Multimédia



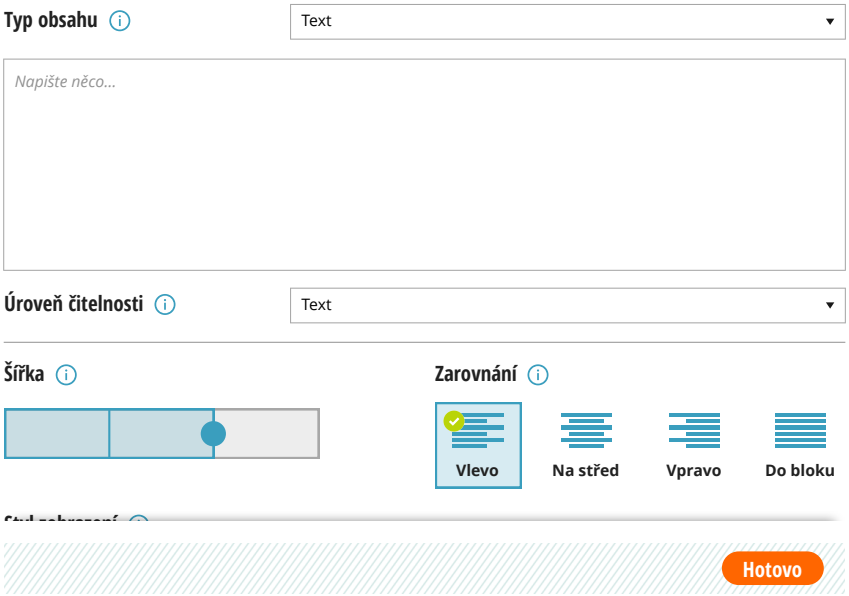
Obrázek B.7: Vyskakovací okno k vložení obrázku



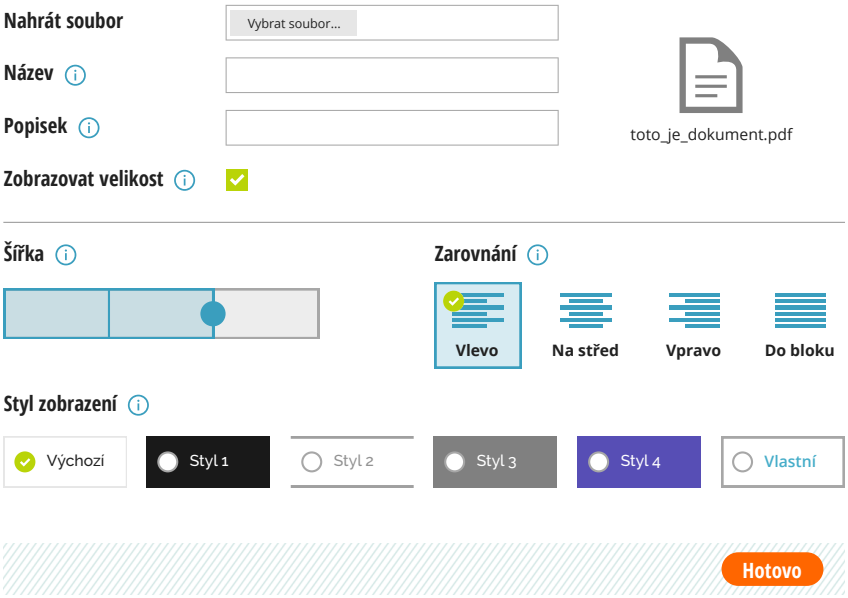
Obrázek B.8: Vyskakovací okno k vložení videa z YouTube



Obrázek B.9: Vyskakovací okno k nahrání videa z počítače

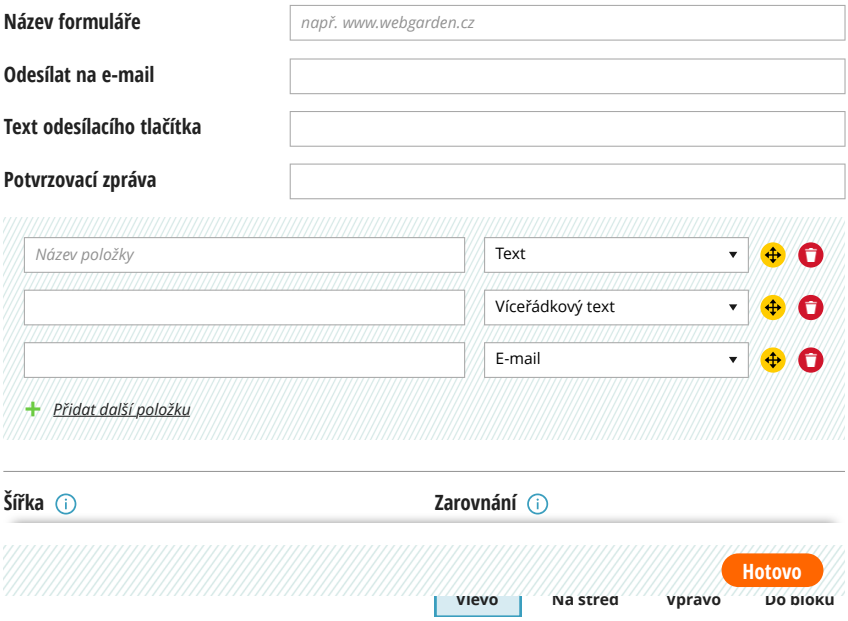


Obrázek B.10: Vyskakovací okno pro vygemerování QR kódu

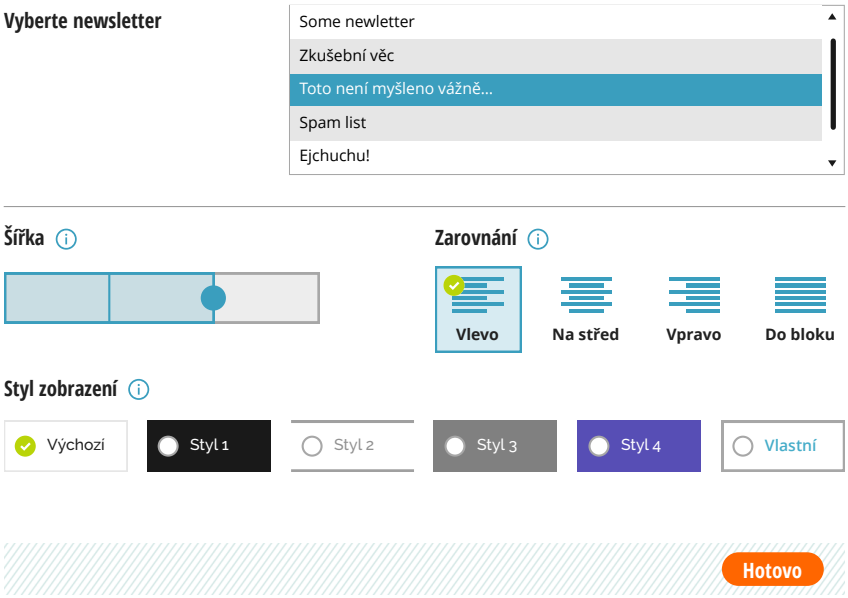


Obrázek B.11: Vyskakovací okno k nahrání souboru

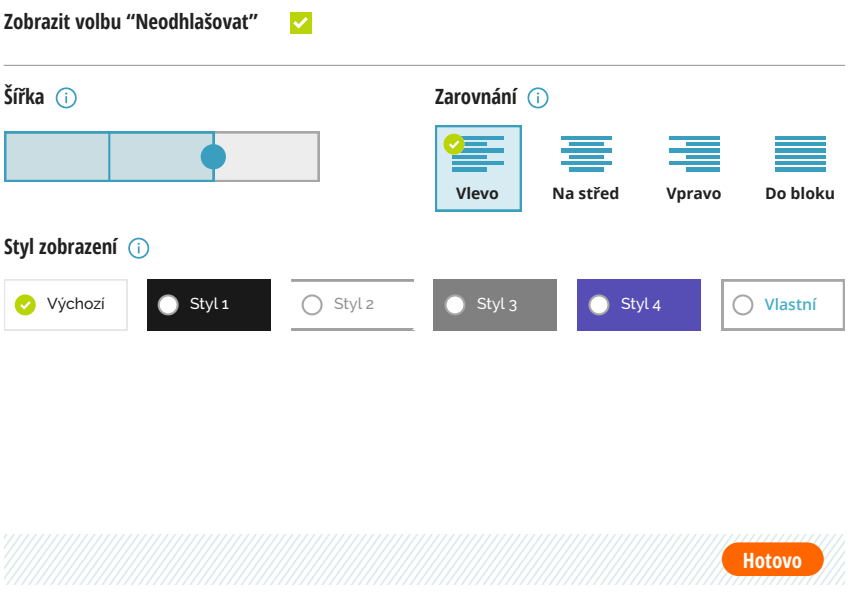
B.2.4 Formuláře



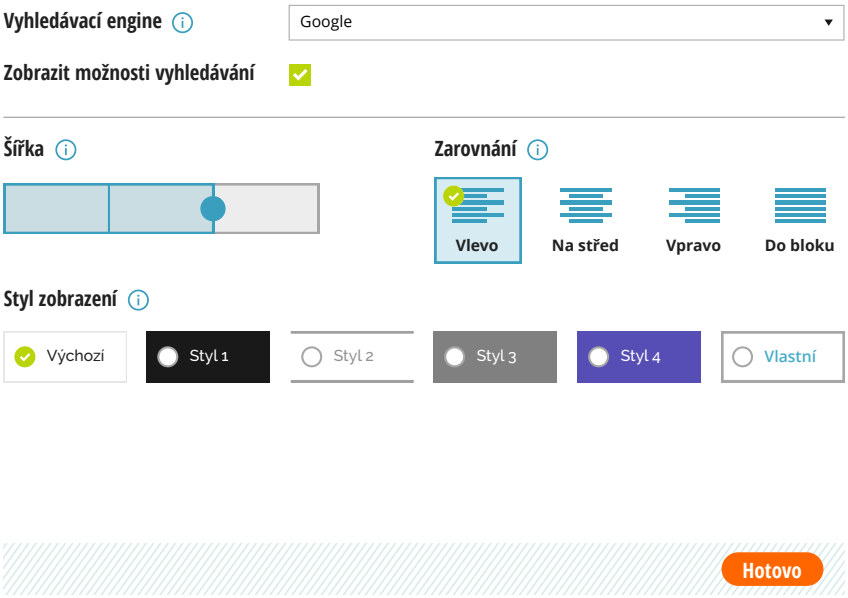
Obrázek B.12: Vyskakovací okno k vytvoření formuláře



Obrázek B.13: Vyskakovací okno k vložení formuláře pro newsletter

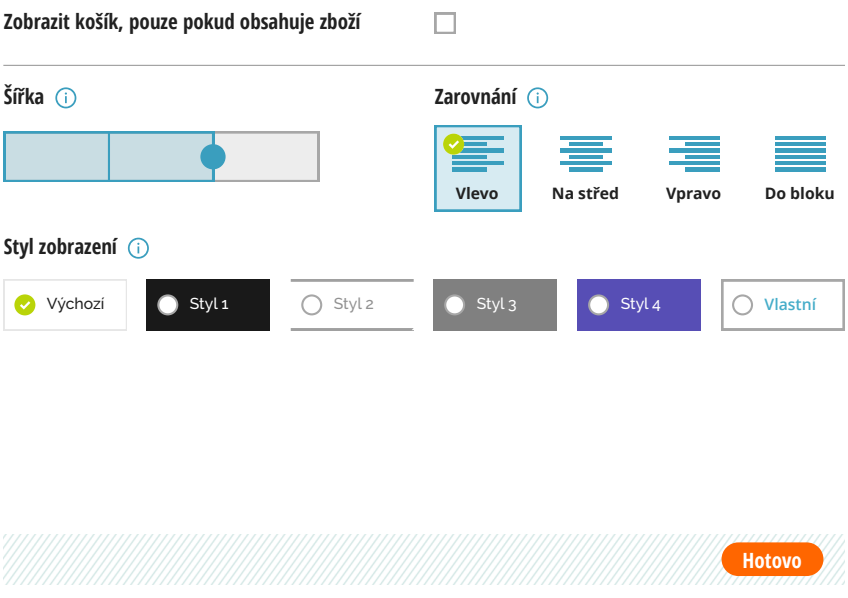


Obrázek B.14: Vyskakovací okno k vložení přihlašovacího formuláře

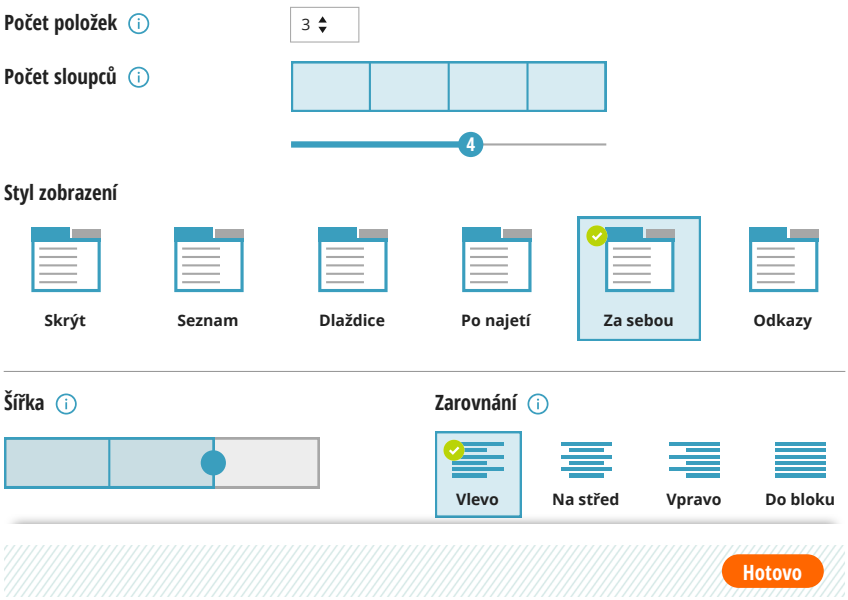


Obrázek B.15: Vyskakovací okno k vložení vyhledávacího formuláře

B.2.5 E-shop



Obrázek B.16: Vyskakovací okno k vložení košíku



Obrázek B.17: Vyskakovací okno k vložení seznamu nejnovějšícho zboží

B.2.6 Sociální sítě

Obrázek B.18: Vyskakovací okno k vložení Facebook komentářů

Příloha C

Seznam použitých zkratek

PHP PHP: Hypertext Preprocesor

XSLT eXtensible Stylesheet Language Transformations

XML Extensible markup language

HTML Hypertext Markup Language

CSS Cascading Style Sheets

DOM Document Object Model

UI User interface

WYSIWYG What You See Is What You Get

FUP Fair Use Policy

CTA Call to action

SASS Stylistically Awesome Style Sheets

Příloha **D**

Obsah přiložené SD karty

- `src` ..... soubory návrhu a implementace
  - `demo` ..... Ukázkový web s HTML, CSS a SASS soubory
  - `xslt` ..... změněné XSLT šablony
  - `administration` ..... Návrhy uživatelského rozhraní administrace
- `thesis` ..... text práce
  - `latex` ..... zdrojová forma práce ve formátu LaTeX
  - `BP_Filip_Trefil_2016.pdf` ..... text práce ve formátu PDF