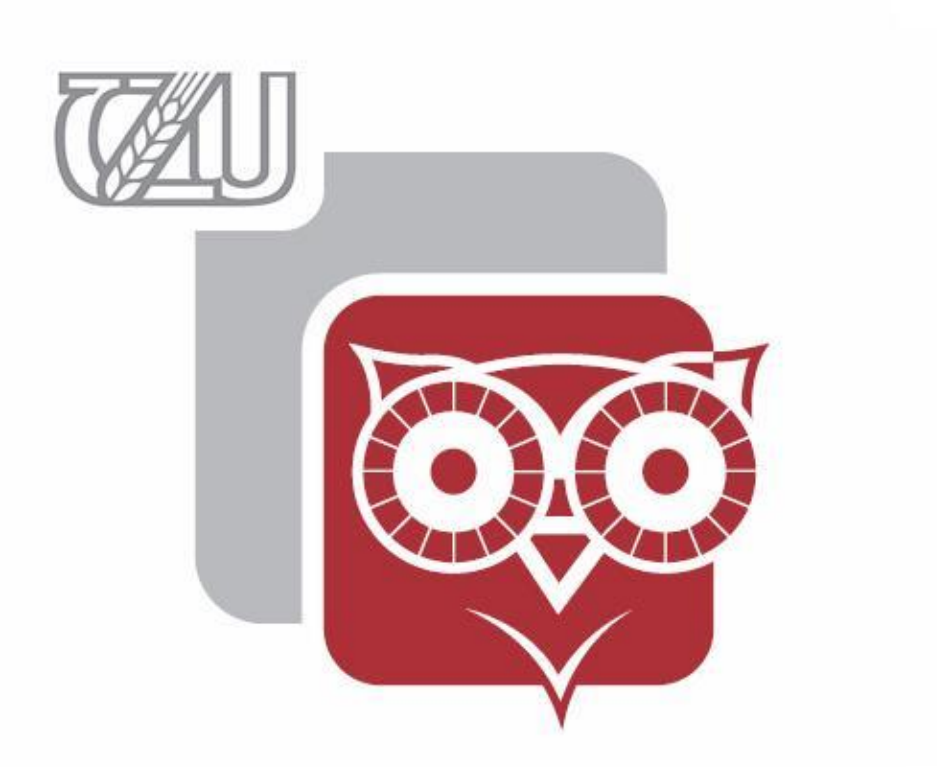
ČESKÁ ZEMĚDĚLSKÁ UNIVERZITA V PRAZE
FAKULTA PROVOZNĚ EKONOMICKÁ

KATEDRA INFORMAČNÍCH TECHNOLOGIÍ



CSS preprocesory

BAKALÁŘSKÁ PRÁCE

Ondřej Havazík

Vedoucí práce Ing. Pavel Šimek, Ph.D.

2016

Čestné prohlášení

Prohlašuji, že svou bakalářskou práci „CSS preprocesory“ jsem vypracoval samostatně pod vedením vedoucího bakalářské práce a s použitím odborné literatury a dalších informačních zdrojů, které jsou citovány v práci a uvedeny v seznamu literatury na konci práce. Jako autor uvedené bakalářské práce dále prohlašuji, že jsem v souvislosti s jejím vytvořením neporušil autorská práva třetích osob.

V Praze dne.

Poděkování

Rád bych tímto poděkoval vedoucímu bakalářské práce panu Ing. Pavlu Šimkovi Ph.D. za vedení práce, velmi užitečné rady, konzultace a hlavně velkou trpělivost po dobu psaní této práce.

CSS preprocesory

CSS preprocessors

Souhrn

Bakalářská práce je tematicky zaměřena na problematiku CSS preprocesorů. Vlastní práce se skládá z vytvoření webové prezentace v preprocesoru SASS. Poukázaní na rozdíly mezi jednotlivými vybranými preprocesory SASS a LESS na reálných vytvořených příkladek dostupných na webové prezentaci. Porovnání jednotlivých preprocesorů SASS a LESS pomocí metod vícekriteriální analýzy variant.

Summary

Bachelor work is thematically focused to problematic of CSS preprocessors. Own work is consist from creating web presentation in SASS preprocesor. Differences between each preprocessors SASS and LESS on real examples which you can find on web presentation. Comparing each preprocessors SASS and LESS with colections metods of multicriterial analyzes of variants.

Klíčová slova

CSS, preprocesor, SASS, LESS, kód, nástroj, HTML, Stylus

Keywords

CSS, preprocessors, SASS, LESS, code, tool, HTML, Stylus

Obsah

Seznam obrázků

Seznam tabulek

1 Úvod:

CSS preprocesor je Framework, který běží nad klasickými kaskádovými styly. V dnešní době mezi neznámější patří preprocesory SASS (Syntactically Awesome Stylesheets), LESS a nejnovější Stylus. Preprocesory využívají spoustu praktických nástrojů, které uživateli velmi usnadňují práci a umožňují velice jednoduše provádět změny v kódu a to například změnou pouze jednoho řádku. Bohužel tento užitečný Framework nevyužívají lidé ještě tolik a to většinou z důvodu, že nemají ponětí o jeho existenci.

Bývá využíván především na velkých projektech a to z důvodu již zmíněné velmi snadné editace a dále také přehlednosti celého zdrojového kódu. Každý preprocesor disponuje pár rozdílnými nástroji (cca 20%), kterými se liší, ale většinu nástrojů (cca 80%) jako například proměnné, mixiny, matematické funkce, importování, vnořování apod. mají stejné. K aktivnímu využívání jednoho z preprocesorů je bezpodmínečně nutná znalost značkovacího jazyka HTML (HyperText Markup Language) a kaskádových stylů. Výhodou je mít i základní povědomí o programovacím jazyce JavaScript.

Téma bakalářské práce bylo zpracováváno z důvodu dlouhodobého zájmu autora o problematiku webových prezentací. K této problematice je minimum informačních zdrojů v českém jazyce, a tudíž v současné době nemá široká odborná veřejnost dostatečné znalosti v dané problematice.

2 Cíle a metodika:

2.1 Cíle práce:

Bakalářská práce je tematicky zaměřena na problematiku CSS preprocesorů. Hlavním cílem práce je porovnání vybraných CSS preprocesorů a ověření jejich použitelnosti na vybraných reálných případech. Dílčí cíle práce jsou:

- Charakteristika principu CSS preprocesorů
- Sumarizace a vývoj CSS

2.2 Metodika práce:

Metodika řešené problematiky bakalářské práce je založena na studiu a analýze odborných informačních zdrojů. Nezbytné je také poukázání na základy problematiky HTML a kaskádových stylů a jejich vývoje, protože bez znalosti těchto dvou jazyků není možné pochopit funkci preprocesorů. Dále bude následovat charakteristika nejvyužívanějších jednotlivých preprocesorů, jako jsou SASS, LESS a Stylus a jejich následná instalace. Instalace bude vysvětlena podle toho, jestli se instaluje na straně klienta či serveru, na jaký operační systém apod. Vlastní práce bude spočívat v komparaci vybraných CSS preprocesorů na vybraných praktických příkladech, které jsou běžně využívány při vytváření webových stránek a aplikací. Pro názornou ukázku bude vytvořena vzorová webová prezentace ve zvoleném frameworku SASS, aby byla prokázána praktická znalost preprocesorů. Webová prezentace bude kompatibilní pro všechny hardwarová i softwarová zařízení a je validován podle standardů koncorsia W3C. Na webové prezentaci se budou nacházet informace o bakalářské práci, zdroje k jednotlivým teoretickým částem na dané stránce a dále praktické příklady nástrojů preprocesorů SASS a LESS s poukázáním na podobnosti a také rozdíly mezi nimi. Také ukázky zdrojových kódů i jejich praktické výstupy. Komparace Frameworků SASS a LESS bude provedena pomocí vícekriteriální analýzy variant. Pomocí Saatyho metody budou určeny váhy jednotlivých kritérií pro dané dvě varianty a posléze vybrána vhodnější varianta pomocí bodovací metody. Bodovací metoda byla vybrána na základě toho, že je přesnější než například metoda pořadí. Na základě syntézy teoretických poznatků a výsledků praktické části budou formulovány závěry bakalářské práce.

3 Teoretická východiska:

3.1 HTML:

Neboli HyperText Markup Language je značkovací jazyk, který se v dnešní době využívá k tvorbě webových stránek, které jsou propojeny hypertextovými odkazy. HTML je hlavní z jazyků, pro vytváření stránek v systému World Wide Web, díky němu je možné publikovat dokumenty na internetu. V důsledku vývoje webových prohlížečů byla nutnost vyvíjet i tento značkovací jazyk. Bez znalosti HTML nemá žádný smysl, se učit jakýkoliv jiný webový jazyk, protože tvoří kostru (strukturu) všech webových stránek. Vzhledem k vysoké provázanosti jazyků HTML a kaskádových stylů je naprostá nutnost ho ovládat. HTML umožňuje autorům například publikovat na internetu dokumenty, co obsahují hlavičku, tabulky, seznamy, fotografie, odkazy, formuláře apod. Dále také získávat informace z internetu pomocí hypertextových odkazů, při kliknutí na ně. [1]

3.1.1 HTML (jedna):

Základy jazyka HTML byly vytvořeny roku 1990 Timem Bernersem Leem s koncepcí World Wide Web. Dříve se nazýval pouze HTML a neměl žádné číselné označení, která jsou v dnešních verzích, i přesto se tomuto standartu dalo říkat HTML 1. Bylo zde pouze pár základních a nejdůležitějších značek, které si uchovali v naprosté většině všech případů svou funkci až do dnešní doby. Dřívější prohlížeč nesl název WorldWideWeb a byl samozřejmě tudíž plně kompatibilní.

Později vznikaly další prohlížeče, které HTML samozřejmě také uměly zobrazovat. [2]

3.1.2 HTML 2.0:

Pro internetovou komunikaci primitivní prohlížeče se základním HTML samozřejmě přestaly logicky stačit, tudíž si autoři nových prohlížečů v následujících letech původní jazyk HTML trošku sami rozšířili. Roku 1994 byl vydán standart, který nesl číselné označení 2.0 a který ve své podstatě shrnoval všechno, co uměly tehdejší prohlížeče. Jednalo se o první verzi, která zcela odpovídá syntaxi SGML. Novinkou této verze je především interaktivní formulář a podpora grafiky. [2]

3.1.3 HTML 3.0:

Velmi nadějná verze na svou dobu, bohužel upadla do zapomenutí, díky své nevalné

podpoře webovými prohlížeči. [2]

3.1.4 HTML 3.2:

Vrchol barev, tabulek a pozadí stránek. Osekaná verze 3 s číselným označením 3.2 se dočkala dvou vynikajících implementací. Zprvu to byl Netscape Navigator 3 a po nějaké době také Microsoft Internet Explorer 3. Netscape sice nepracoval s kaskádovými styly, ale měl už poměrně zajímavou podporu JavaScriptu. HTML 3.2 se začalo využívat již od roku 1995 a v praxi ho bylo možné zaznamenat u nových projektů a prací až někdy do roku 2005. V důsledku toho, že tento standart nepodporoval kaskádové styly, kód obsahoval mnoho formátovacích značek a poměrně pokročilou práci s tabulkami. Autoři se při práci ve verzi 3.2 naučili, že nezáleží tak úplně na tom co definuje norma, ale že jde především o to, co definují prohlížeče. [2]

3.1.5 HTML 4.0:

Tento standart nesoucí číselné označení 4.0 byl vydán 18 prosince 1997 komunitou W3C.Byla to vůbec první verze jazyka, která se snažila výrazně zjednodušovat zápis kódu, avšak byla velmi brzy nahrazena nadcházející verzí 4.01. [2]

3.1.6 HTML 4.01:

Tento standart nesoucí číselné označení 4.01, který nahradil svého předchůdce, 4.0 byl vydán 24. prosince 1999 komunitou W3C. Tato verze sebou přinesla několik drobných vylepšení. Přibylo pár nových tagů jako například div, span a pokročilejší tabulky. Nese sebou také zjednodušení v tom směru, že jasně deklarovala oddělení smyslu a vzhledu. Tato verze počítala již s tím, že existují kaskádové styly, tudíž formátování prvků přímo v HTML kódu vyšlo z módy a bylo označováno za zastaralé. Autoři webů si ovšem velmi brzo zvykli pracovat s takovou kombinací HTML 3.2, 4.01 a CSS 1, kterou jim prohlížeče dovolovali. A je nutné podotknout, že toho umožňovali opravdu hodně. Od roku 1999 může autor použít vesměs cokoliv a dodnes mu to prochází. [2]

3.1.7 XHTML 1.0:

Byla to uplně první verze jazyka XHTML, která byla dokončena a představena roku 2000. V podstatě kopírovala svého předchůdce HTML 4.01, avšak byla vytvořena tak, aby

vyhovovala normě XML.

To znamenalo:

- Uzavírat tagy
- Psát tagy malými písmeny
- Povinné hodnoty atributů
- ….apod

Postoje prohlížečů k XHTML byly různé. Některé na něj používaly (a stále používají) parser na HTML. Jiné využívají XML parser nebo se chovají odlišně v důsledku toho, s jakým mime typem soubor na klienta dorazí. [2]

3.1.8 XHTML 1.1:

Tento standart definitivně vycházel z 1.0. V XHTML 1.1 bylo zakázané co bylo v předešlé verzi označeno za nedoporučené. Zavrhnuty byly atributy jako například font, čímž byl vytyčen cíl, aby byl obsah pouze v XHTML souboru a grafická část se měla přesunout do souboru CSS. Bohužel tento standart měl v praxi minimální využití, šlo tam spíše o krásný cíl, než o jeho praktické využití. Autoři, kteří ho používali, to spíše činili z důvodů náboženských než praktických. [2]

3.1.9 XHTML 2.0:

Tento standart nestál ani za řeč. Rušily se zde různé tagy, ale přidávaly se některé nové. Tato technologie nebyla hlavními prohlížeči naštěstí podporována. [2]

3.1.10 Rozdíl mezi XHTML a HTML:

- Tady a atributy jsou malými písmeny
- Nepárové tagy končí lomítkem
- Párové tagy jsou párové povinně
- Všechny atributy musejí mít hodnotu
- Interní JavaScript a styly se zapisují jiným způsobem
- Dokument má mít XML prolog
- Dokument požaduje správný DOCTYPE [2]

3.2 HTML5:

Standart, jehož finální specifikace byla vydána 28. října 2014. Přináší opravdu mnoho novinek oproti svému předchůdci. Verze 5 sebou nese spoustu novinek, přibyly zde nové tagy, definující strukturu stránky, perzistentní úložiště formou asociativního pole, relační databáze s podporou transakcí a podpora offline aplikací, což je v podstatě nejpřevratnější novinka v novém HTML. [2]

3.2.1 Novinky:

3.2.1.1 DOCTYPE:

Velmi zkrácený zápis DOCTYPE, oproti předešlé verzi. Není tedy nutné zadávat dlouhé zápisy o verzi a DTD specifikaci dokumentu, tudíž zápis dokumentu začíná pouhým <DOCTYPE html>.

Největší výhodou tohoto zápisu je, že všechny dnešní prohlížeče tomuto zápisu zcela rozumí a tudíž se stránky zobrazí ve standartním zobrazovacím režimu. [3] [4]

3.2.1.2 Jazyk dokumentu a kódování:

Na rozdíl od svého předchůdce se zápis jazyku, který dokument využívá, také dosti zjednodušil na prostou informaci v kořenovém prvku (<html>). Zapisuje se to v HTML5 asi takto: <meta charset="UTF-8">, oproti zastaralému a složitému zápisu, který vypadá takto: <meta http-equiv="Content-Type" content="text/html; charset=UTF-8">. [3] [4]

3.2.1.3 Struktura dokumentu:

Důraz byl při vývoji velmi kladen na sémantiku webových stránek a přehlednost webového kódu stránek. Webové stránky se klasicky rozdělují do dnes na obvyklé části jako je například hlavička, patička, různé sloupce, sekce apod. Tyto části byly v HTML 4.01 odlišeny pomocí prvku div s přiřazenou vlastností třída nebo id, protože zde neexistovali žádné speciální prvky, které by byly toto schopné rozlišit.

V HTML 5 se proto zavedli nové značky, za účelem strukturování stránek.

- <section>
    - Představuje část stránky, například kapitoly
- <article>
    - Představuje nezávislé části stránky, například články či komentáře

- <main>
    - Představuje hlavní obsah stránky
- <aside>
    - Představuje část stránky, které nepatrně souvisí s obsahem zbytku stránky, například to bývají poznámky po straně webové stránky
- <hgroup>
    - Představuje skupinu nadpisů od h1 až po h6
- <header>
    - Představuje hlavičku, může obsahovat navigační odkazy nebo například nadpis
- <footer>
    - Představuje patičku, většinou obsahuje copyright (autorská práva) a různé informace
- <nav>
    - Představuje část stránky, která je určena k navigaci
- <figure>
    - Představuje samostatný obsah stránky, který doplňuje hlavní stať, ale není její součástí, například graf, obrázek, video nebo ukázka kódu
- <figcaption>
    - Představuje popisek pro figure [5]

3.2.1.4 Multimediální obsah stránky:

V předcházejících verzích HTML se nevyužívají nástroje pro vkládání multimediálního obsahu na stránky, proto dříve byla nutnost využívat alternativní varianty například formou plug-in či dnes hodně využívaný flash. Verze HTML5 už ale oproti svým předchůdcům umožňuje provádět základní operace například s videem vkládaným na stránce nebo různá audia.

Příklady tagů:

- <video>
    - Představuje nám video nebo filmový obsah
- <audio>
    - Představuje nám hudbu nebo písničky

- <track>
    - Definuje nám stopy ve <video> či <audio>
- <source>
    - Definuje nám zdroj <video> či <audio>
- <embed>
    - Definuje nám obal pro externí aplikace (plug-in) [6]

3.2.1.5 Nové vstupní typy:

Nové vstupní typy:	Nové vstupní atributy:
color	autocomplete
date	autofocus
datetime	form
datetime-local	formaction
Email	formenctype
month	formmethod
number	formnovalidate
range	formtarget
search	height and width
tel	list
time	min and max
url	multiple
week	pattern (regexp)
	placeholder
	required
	step

[6]

3.3 CSS:

Neboli Cascading Style Sheets je to webový stylistický jazyk, který slouží na popis způsobu zobrazení elementů na stránkách, které jsou napsány v jazyce HTML, XML či XHTML. CSS popisuje, jak se elementy mají zobrazovat na obrazovce, papíru či přenosném médiu. Jazyk byl navrhnut standardizační organizací W3C, autorem prvotního návrhu byl Hakon Wium Lie. Kaskádové styly vznikly v roce 1997. Hlavním cílem kaskádových stylů

je umožnění uživatelům oddělit strukturu a obsah webové stránky a grafickou část. Tuto funkci mělo zastávat již HTML, ale v důsledku nedostatečných standardů a konkurence výrobců webových prohlížečů se vyvinul jinak. Kaskádové se jím říká proto, že mají možnost se na sebe vrstvit, této vlastnosti se říká dědičnost. Styly jsou provázané s HTML, je nutná plná znalost tohoto jazyka. [7] [8]

3.3.1 Syntaxe:

Kaskádové styly obsahují pravidla, kterými se musí řídit. Každé z vybraných pravidel obsahuje selektor a blok deklarací. Bloky jednotlivých deklarací se oddělují středníky, naproti tomu deklarace se sestaví z tzv. identifikátoru vlastnosti, následné dvojtečky a dále hodnota dané vlastnosti.

```
/* CSS Document */
p
{
  color:blue;
  font-size: large;
  border:  solid black;
}
```

Obrázek 1- Příklad CSS syntaxe

Na tomto příkladu je p selektor, obsahu v závorkách se říká blok deklarací, deklarace samotná je v tomto případě například „color:blue;“. Color je indentifikátor vlastnosti a blue je její hodnota. Tento kód nastavuje v odstavci p barvu na modrou, velikost písma na velkou a černé plné ohraničení. [9] [10]

3.3.2 Zavedení CSS do HTML dokumentů:

Máme tři základní typy, jak můžeme zapisovat do HTML dokumentu kaskádové styly. Některé jsou využívanější a některé méně, z důvodu jejich přehlednosti a možnosti editace. [10]

3.3.3 Přímý zápis do HTML dokumentu (stylopis):

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250">
  <meta name="generator" content="PSPad editor, www.pspad.com">
  <title></title>
<style type="text/css">
h1
{
color: blue; font-style: italic;
}
</style>
  </head>
  <body>
  <h1>Nadpis H1</h1>
  </body>
</html>
```

Obrázek 2 - Přímý zápis do HTML dokumentu

3.3.4 Pomocí externího souboru .css:

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
<html>
  <head>
  <meta http-equiv="content-type" content="text/html; charset=windows-1250">
  <meta name="generator" content="PSPad editor, www.pspad.com">
  <title></title>
  <link rel="stylesheet" href="styly.css" type="text/css">
  </head>
  <body>
  <h1>Nadpis H1</h1>
  </body>
</html>
```

Obrázek 3 - Pomocí externího souboru .css

3.3.5 Přímý zápis inline:

```
0        10        20        30        40        50        60        70
1 <!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN">
2 <html>
3   <head>
4   <meta http-equiv="content-type" content="text/html; charset=windows-1250">
5   <meta name="generator" content="PSPad editor, www.pspad.com">
6   <title></title>
7   </head>
8   <body>
9   <h1 style="color: blue; font-style: italic">Nadpis H1</h1>
10  </body>
11 </html>
```

Obrázek 4 - Přímý zápis inline

3.4 CSS1:

Internet explorer byl v podstatě první velmi rozšířený prohlížeč, který podporoval CSS1, byla to verze 3 z roku přibližně 1996. Podpora CSS1 se omezovala na písma a barvy. Už podstatně lepší podpory se dočkaly kaskádové styly později ve čtvrtých verzích prohlížečů jak už Internetu Exploreru tak Netscape Navigatoru někdy kolem roky 1998. Oproti svým předchůdcům se již tyto prohlížeče snaží podporovat kompletní CSS včetně pozicování. V různých podverzích čtyřkových verzí se postupně podpora CSS1 zdokonalovala. Především u Nescapu nebyla ale moc dobrá podpora. Chování tohoto prohlížeče je velmi zmatené a chaotické a spoustu vlastností vůbec nepodporuje. Toto je jeden z hlavních důvodů proč kaskádové styly byly dlouhou dobu v pozadí a nevyužívalo je mnoho webmasterů, právě díky jejich nespolehlivosti. Ani Internet Explorer 4 nepodporuje CSS1 naprosto bez chyb. Sice má podporovat všechny vlastnosti, ale ani zdaleka není možné je aplikovat na veškeré elementy na webové stránce (nejdou nastavit okraje u odkazů apod.). Naopak Internet Explorer 4 obsahuje vlastnosti, které nejsou v CSS1 definovány (filtry, systémová písma a barvy apod.) a zavádí také velmi užitečnou pseudotřídy a:hover, která umožnuje při přejetí myší změnu textových odkazů.

Ani jeden ze čtyřkových webových prohlížečů nedovoluje speciálně formátovat tzv. pseudoelementy (tzn. první písmeno a první řádek elementu, nejlépe odstavce), i když to CSS1 přikazuje (zvládají to až pozdější verze prohlížečů typu Mozilla a Internet Explorer 5.5). Nová verze 5.5 přišla s lepší podporou CSS. Nové možnosti jako například barvení rolovací lišty, zvětšování objektů a převracení textu, což jsou věci, které do CSS1 nepatří. [11]

3.5 CSS2:

Je další verze kaskádových stylů, která sebou přináší mnoho novinek a vylepšení v podobě nových vlastností formátování písma, ale i mnoho dalších praktických věcí. Ustanovení konečné verze bylo někdy kolem roku 2000. Převratnou novinkou, kterou sebou nese nástup CSS2 je, že se měla stát formátovacím jazykem i pro jiné webové platformy a i pro jiné jazyky než jen HTML, ale například i pro formátování XML dokumentů. Sice novou verzi kaskádových stylů začalo podporovat mnohem více prohlížečů, avšak žádný z prohlížečů nepodporuje z CSS2 všechno, ale jenom něco. Nejlépe na tom v tomto ohledu je Mozilla, která se snaží dodržovat standardy. Prohlížeče Microsoftu s každou novou verzí vylepšují podporu, avšak plné kompatibility se jim nikdy nepodařilo dosáhnout. Právě proto je CSS2 zatím vágní technologií, která se spíše cituje, než používá. Také jí totiž nelze spolehlivě použít.

Hlavní výhodou CSS2 jsou různá média, na které se zaměřuje. Mimo výstup na obrazovce se totiž styly dají také využít na odlišný tisk, na zobrazení na mobilech, projekcích, či dokonce hlasový výstup nebo zařízení pro slepecké písmo. Už v dnešních prohlížecích je podpora odlišného tisku. [11]

3.6 CSS3:

Již třetí a doposud poslední verze kaskádových stylů a to již od roku 2005. Vývoj této technologie byl zahájen společností W3C. Plné dokončení této technologie se očekává roku 2015, avšak již dnes je většina vlastností podporována nejběžnějšími webovými prohlížeči. Tvůrci technologie CCS2 jsou si vědomi důležité lekce, změnili zcela převratně způsob vzniku CSS3. Celá specifikace je rozdělena do modulů, dělí se na individuální vývojové cykly a je tvořen těmito fázemi: [12]

3.6.1 Fáze vývojových cyklů:

- Pracovní návrh
- Poslední výzva
- Kandidát k doporučení
- Navržený k doporučení
- Doporučení

Specifikace CSS3 dovoluje rychlejší, pružnější a efektivnější změny ve specifikaci jako reakce na potřeby vývojářů webových stránek a zpětnou vazbu od výrobců webových

prohlížečů při implementaci nových vlastností.

CSS3 byl rozdělen do tzv. „modulů“. Obsahuje starou specifikaci CSS, které byly rozděleny na menší kousky, a navíc byli přidány nové moduly: [12]

3.6.2 Moduly CSS3:

- Selektory
- Box modely
- Pozadí a ohraničení
- Hodnoty obrázku a nahrazení obsahu
- Textové efekty
- 2D a 3D transformace
- Animace
- Vícenásobné uspořádání layoutu
- Uživatelské rozhraní [12]

3.6.3 Základní vlastnosti CSS3:

3.6.3.1 Zaoblené rohy:

V jazyce CSS3 je možné zaoblit rohy téměř všech elementů od formulářů přes obrázky až po odstavce textů, bez potřeby dodatečného značkovacího kódu. Řeší se pomocí border-radius. Tato vlastnost se dá použít s jednou, dvěma či čtyřmi hodnotami. Je nutné zapsat s vlastnostmi pro jednotlivé prohlížeče z důvodu správné kompatibility. [13]

```
div
{
  background: #999;
  float: left;
  height: 150px;
  margin: 10px;
  width: 150px;
}
```

```
.vsechny-rohy
{
  -moz-border-radius: 20px;
  -webkit-border-radius: 20px;
  border-radius: 20px;
  }
.jediny-roh
{
  -moz-border-radius-topleft: 75px;
  -moz-border-top-left-radius: 75px;
  -webkit-border-top-left-radius: 75px;
  border-top-left-radius: 75px;
}
```

```
.elipticke-rohy
{
-moz-border-radius: 40px / 20px;
-webkit-border-radius: 40px / 20px;
border-radius: 40px / 20px;
}
.kruh
{
-moz-border-radius: 50%;
-webkit-border-radius: 50%;
border-radius: 50%;
}
```

Obrázek 5 - Zaoblené rohy (CSS3)

3.6.3.2 Vržení stínu na text:

Tato vlastnost nastavuje vržený stín u obyčejného textu. Používá se na to vlastnost

text-shadow. Tato vlastnost se skládá ze čtyř hodnot. Z posunu po ose-x, po ose-y, poloměr rozostření a barva vrženého stínu. Je nutné zapsat s vlastnostmi pro jednotlivé prohlížeče z důvodu správné kompatibility. [13]

```
h1
{
  font-family: Helvetica, Arial, sans-serif;
  font-size: 72px;
  line-height: 1em;
  text-shadow: 2px 2px 5px #999;
}

.vice
{
  text-shadow: 2px 2px 0 rgba(255,255,255,1),
  6px 6px 0 rgba(50,50,50,.25);
}
```

Obrázek 6 - Vržení stínu na text

3.6.3.3 Vržení stínu na element:

Tato vlastnost nastavuje vržený stín na obyčejný text, jako to bylo ve výše uvedeném případě, ale na samostatný element. Používá se na to vlastnost box-shadow. Na rozdíl od vrženého stínu na text, kde se používaná vlastnost skládá ze čtyř hodnot, tady je možné navíc nastavit, zda daný element bude vypadat jako vsazený či vystouplý, za pomocí hodnot inset a outset. Tato vlastnost se liší oproti stínu na text také tím, že není tolik rozšířená ve všech prohlížečích. [13]

```
div
{
  background: #fff;
  height: 150px;
  float: left;
  margin: 10px;
  width: 150px;
}

.stin
{
  -moz-box-shadow: 2px 2px 5px #000;
  -webkit-box-shadow: 2px 2px 5px #000;
  box-shadow: 2px 2px 5px #000;
}

.vsazeny-stin
{
  -moz-box-shadow: inset 2px 2px 10px #000;
  -webkit-box-shadow: inset 2px 2px 10px #000;
  box-shadow: inset 2px 2px 10px #000;
}

.vice
{
  -moz-box-shadow:2px 2px 10px rgba(0,255,0,.75),5px 5px 20px rgba(125,0,0,.5);
  -webkit-box-shadow: 2px 2px 10px rgba(0,255,0,.75), 5px 5px 20px rgba(125,0,0,.5);
  box-shadow: 2px 2px 10px rgba(0,255,0,.75), 5px 5px 20px rgba(125,0,0,.5);
}
```

Obrázek 7 - Vržení stínu na element

3.6.3.4 Přechody na pozadí:

K novinkám v CSS3 patří také barevné přechody. Díky nimž je možné vytvářet

přechody z jedné barvy na druhou. [13]

3.7 CSS preprocesory:

Preprocesor je webový jazyk, který stojí nad CSS. Do jazyku jsou pomocí nich přidány nové vlastnosti a vylepšují a doplňují technické nedostatky. Mezi nejznámější běžně využívané preprocesory patří Sass, Less a Stylus. Preprocesory se dají využívat od malých webů až po velmi rozsáhlé projekty. Čím více lidí kód kaskádových stylů upravuje nebo čím komplexnější projekty jsou vytvářeny, tím jsou preprocesory více důležité. Preprocesory jsou kompilovány přímo do kódu CSS obvykle lokálně během práce formou (*.less, *.sass, *styl) souboru. [14] [15]

3.7.1 Výběr preprocesoru:

Rozdíl mezi jednotlivými preprocesory není tak veliký, jako samotný rozdíl mezi klasickým CSS3 a preprocesorem. Preprocesor se vybírá především podle využití a sympatiím uživatele. Záleží zde na tom, zda se uživatel cítí například více jako grafik nebo naopak jako programátor.

Pro koho je jaký jazyk určen?	
Designér	LESS
Programátor	SASS, Stylus
Snadné učení	LESS
Technická pokročilost	Stylus
Velikost komunity	LESS
Bootstrap	LESS, SASS
Foundation	SASS

Tabulka 1 - Výběr preprocesoru

[16]

3.8 SASS:

Obrázek 8 - SASS logo

Sass neboli Syntactically Awesome Stylesheets je CCS preprocesor – to je vrstva, mezi autorem a CSS soubory, které se předávají prohlížeči. Tento preprocesor vyplňuje mezery v klasických kaskádových stylech, umožňuje psát čistý kód, který bude rychlejší, mnohem efektivnější a v neposlední řadě lepší na údržbu a editaci. Sass zprvnu navrhnul Hampton Catlin a vyvinula Natalie Weizenbaum. Poté Weizenbaum a Chris Eppstein pokračovali ve vývoji Sass se SassScriptem, jednoduše skriptovací jazyk použitý v Sass souborech. [17]

3.8.1 Dvojí syntaxe:

Ve skutečnosti existují dva druhy odlišných syntaxí. Nejstarší je SCSS syntaxe. SCSS soubory se značí na rozdíl od klasického .css takto .scss . Výhody jsou například v tom, že nevyžaduje změnu ve formátování kódu nebo je snazší postupně převádět existující kaskádové styly na funkční SASS.

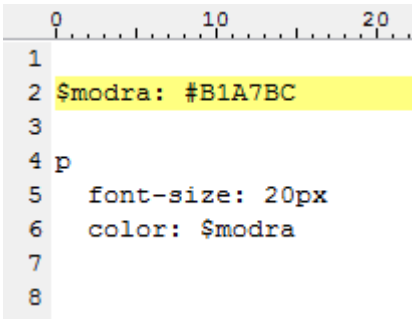
Jednoduchá ukázka jak funguje syntexe SCSS. Definuje se proměnná a poté se použije v deklaraci CSS.

```
0         10        20
1
2 $modra: #B1A7BC;
3
4 p
5 {
6   font-size: 20px;
7   color: $modra;
8 }
9
10
```

Obrázek 9 - Syntaxe .scss

Druhá varianta je tzv. originální členitá SASS syntaxe. Někteří lidé jí preferují více

z důvodů, že se zde nevyužívají složené závorky a je celkově zjednodušenější.
Jednoduchá ukázka jak funguje originální SASS syntaxe. Stejný příklad jako u SCSS. [17]



Obrázek 10 - Syntaxe originálního SASS

3.8.2 Instalace:

3.8.2.1 Vývojová prostředí:

Existuje zde mnoho dobrých aplikací, ve kterých je možné se SASSem pracovat. Je možné aplikace stáhnout bud zadarmo či placené. [17]

Název aplikace:	Licence:	Podpora:
Codekit	Placená	Apple
Compass.app	Placená, Open source	Apple, Windows, Linux
Hammer	Placená	Apple
Koala	Open source	Apple, Windows, Linux
LiveReload	Placená, Open source	Apple, Windows
Prepros	Placená	Apple, Windows, Linux
Scout	Open source	Apple, Windows
GhostLab	Placená	Apple, Windows

Tabulka 2 - Vývojová prostředí SASS

3.8.2.2 MAC OS:

Instalace je opravdu velice jednoduchá. MAC OS X přišel jako jediný operační systém s předinstalovaným Ruby a SASS je v balíčku jako Ruby „gem“, což je chytrý programátorský termín pro Ruby aplikace. Stačí jen otevřít Terminal.app, napsat příkaz $ gem install sass potvrdit pomocí enter a to je vše. Vše se samo nainstaluje samo. [17]

3.8.2.3 Linux:

Instalace v Linuxu není nikterak těžká, ale stejně jako Windows nemá předinstalovaný Ruby, tudíž je nutné ho nejprve nainstalovat přes apt packet managera, rbenv či rvm. Použijeme k tomu příkaz sudo su –c "gem install sass. [17]

3.8.2.4 Windows:

Bohužel oproti MAC OS X, Windows nemá předinstalovaný Ruby. Na oficiálních webových stránkách (http://bkaprt.com/sass/5/) se musí stáhnout RubyInstaller pro Windows, aby se dal SASS spustit na počítači s Windows. Instalátor také nainstaluje příkazovou Ruby řádku powershell, díky které je možné využívat Ruby knihovny. [17]

3.9 LESS:

Obrázek 11 - LESS logo

Less rozšiřuje kaskádové styly o dynamické prvky, jako jsou například proměnné, mixiny, výpočty a funkce. Běží jak na straně klientské (Chrome, Safari, Firefox, IE), tak na straně serveru, s Node.js a Rhino. Byl navrhnut Alexisem Sellierem v roce 2009, Less je ovlivněn Sassem a ovlivnil novější „SCSS" syntaxi Sassu. Less je open-source, což značí počítačový software s otevřeným zdrojovým kódem, s otevřeným znamená, že při dodržování jistých podmínek, je uživateli umožněno zdrojový kód využívat, prohlížet či upravovat. Původně byl napsán v Ruby a později byl předělán do JavaScriptu. Nachází se v něm i mnoho nástrojů třetích stran, umožňujících kompilovat soubory a sledovat změny. Hlavní podstatou Lessu je jednoduchost na naučení, respektuje tedy maximálně deklarativní povahu CSS. Sice se poměrně rychle učí, avšak u pokročilejších postupů (například u cyklů a podmínek) není již zápis tak elegantní. [18] [19] [20]

3.9.1 Instalace:

Pro instalaci LESS je nutné si nainstalovat na platformu speciální Eclipse LESS plug-

in, který potřebujeme pro instalaci. Eclipse Kepler určený pro verze 4.3. a starší se dá stáhnout zde.

Po instalaci tohoto pluginu stačí pouze vytvořit soubor s koncovkou .less, který je možné otevřít v libovolném LESS editoru a je s ním dále možné pracovat. [21]

3.9.1.1 Na straně klienta:

Pro použití na klientské straně se připojí do souboru stylesheet.less s atributem rel nastaveným na "stylesheet/less".

<link rel="stylesheet/less" type="text/css" href="styles.less">

Soubor less.js je možné stáhnout například odsud přímo z vrchu stránky a vložit jej do hlavičky souboru na své stránce. Důležité je připojit stylesheet před skriptem. [18]

3.9.1.2 Na straně serveru:

Nainstalování LESSu na straně serveru je velmi jednoduché. Lze toho docílit pomocí nástroje npm (node package manageru):

$ npm inslall less

Po instalaci je možné vyvolat kompilátor z nodu kupříkladu tímto způsobem: [18]

```
var less = require('less');

less.render('.class { width: 1 + 1 }', function (e, css) {
    console.log(css);
});
```

výsledkem bude:

```
.class {
  width: 2;
}
```

Obrázek 12 - Vývolání Node kompilátoru

3.10 Stylus:

Obrázek 13 - Stylus logo

Nejmladší preprocesor, který vychází ze svých předchůdců SASS a LESS a do jisté míry se poučuje z jejich chyb či nedostatků. Stylus je inovativní stylistický jazyk, který se posléze kompiluje do čistého CSS. Je postaven na node.js a schopný běžet na webovém prohlížeči jako webová prezentace. Stylus sebou nese spoustu nástrojů, které využívali jeho předchůdci jako například vytváření mixinů, importování stylů, definování proměnných. Ale přináší sebou i mnoho pokročilých funkcí. [22]

3.10.1 Instalace:

Instalace preprocesoru stylus je velice snadná. Stačí mít pouze nainstalovaný Node.js. Poté stačí zadat pouze příkaz do terminálu:

$ npm install stylus –g [23]

3.10.1.1 Ruční vygenerování CSS:

Soubor stylusu je možné vytvořit kdekoliv v jakémkoliv projektu s koncovkou .styl. Není třeba žádný konfigurační soubor na kompilaci. Jednoduše se rozjede stylus služba, díky které se vygeneruje CSS výstupní soubor.

$ stylus stylus/soubor.styl -out /css –compress [23]

3.10.1.2 Automatické vygenerování CSS:

Pokud uživatel preferuje automatické vygenerování CSS oproti ručnímu je možné využít možnost –watch.

$ stylus –watch stylus/soubor.styl [23]

4 Vlastní práce:

4.1 Webová prezentace v preprocesoru SASS:

Webová prezentace k bakalářské práci na téma „CSS preprocesory“ je vytvořena kompletně ve značkovacím jazyku HTML5 a graficky upravena pomocí CSS3 a za využití preprocesoru SASS. Jednotlivé stránky jsou ovšem označeny koncovkou .php, nikoli .html. Webová prezentace je plně kompatibilní ve všech internetových prohlížečích i na jiných hardwarových zařízeních jako například na tabletu, chytrém telefonu či pda (web však není plně mobile-friendly). K vytvoření webu bylo nutné si připravit vyhovující prostředí, ve kterém je možné daný preprocesor SASS plně využívat. (jak toto prostředí nainstalovat je popsáno v bodu 7.2.3 instalace či na oficiálních stránkách sass-lang.com). Detaily zdrojového kódu jako celku včetně obrázků je možné nalézt na přiloženém cd.

Celý web se skládá ze 7 odkazů, na kterých se nachází:

- Úvod
- HTML
- CSS
- Preprocesory
- Instalace
- Nástroje
- Zdroje

4.1.1 Zvolená syntaxe:

Webová prezentace je napsána pomocí preprocesoru SASS a je možné využívat dvojí syntaxi. Syntaxi SCSS nebo tzv. originální členitá SASS syntaxe. V tomto případě se jedá o syntaxi SCSS. Tzn. že jednotlivé sobory preprocesoru jsou označeny koncovkou .scss a posléze se kompilují do výsledného souboru kaskádových stylů .css, ve kterém se nikdy nic nesmí měnit! Slouží jako výchozí soubor pro vygenerované css a nesmí být v žádném případě editován, editovat se mohou pouze scss soubory.

4.1.2 Vývojové prostředí (NetBeans):

Netbeans IDE je free Java IDE, podporující několik jazyků jako například PHP, JavaFX, C/C++, JavaScript, Html, Css a především preprocesory SASS a LESS. Podporuje také

nespočet frameworků. Tento software nese označení open-source licence, což znamená, že software může být různě modifikován, upravován a volně šířen.

Tudíž se dá volně stáhnout například z oficiálních stránek https://netbeans.org . Avšak je třeba mít předtím nainstalovanou Javu v počítači. Netbeans umí již zmíněný SASS preprocesor, ve kterém je web napsaný. Je možná kompilace přímo uvnitř programu, tudíž není nutné využívat například program Scout, který se využívá na právě již zmíněnou kompilaci zdrojového kódu, což je skvělé pro uživatelé, kteří při tvorbě rádi využívají co nejméně programů naráz.

Obrázek 14 - Webová stránka pro stažení NetBeans

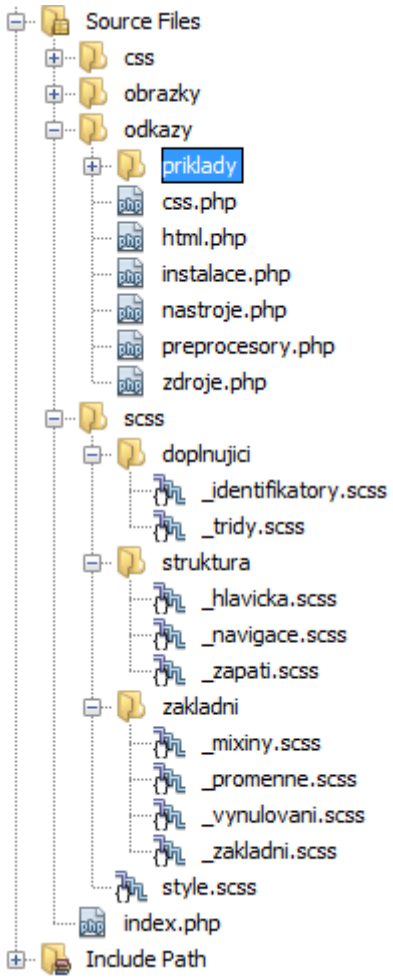
4.1.3 Struktura webu:

Struktura webové prezentace se skládá ze 4 složek a 1 souboru:

- **obrázky**
  - zde se nacházejí obrázky vyskytující na webu
- **odkazy**
  - zde se nacházejí odkazy na jednotlivé stránky
  - uvod, html, css, preprocesory, instalace, nastrojes, zdroje
  - dále se zde nachází složka „priklady“, ve které se nacházejí jednotlivé souboy výstupů příkladů
  - matematicke_funkce, mixiny, promenne, rozsireni, vnorovani
- **scss**

- složka, která se skládá ze 3 podsložek (doplňující, struktura, zakladni)
- souboru style.scss, do kterého se importují ostatní .scss soubory z ostatních podsložek, které byli vytvořeny kvůli přehlednosti a velmi snadné případné změny v nich

- **css**
  - do této složky se generuje výsledné css do souboru style.css a vytvářejí se zde stejnojmenné podsložky jako v složce scss
- **index.php**
  - spouštěcí a zároveň úvodní soubor stránky

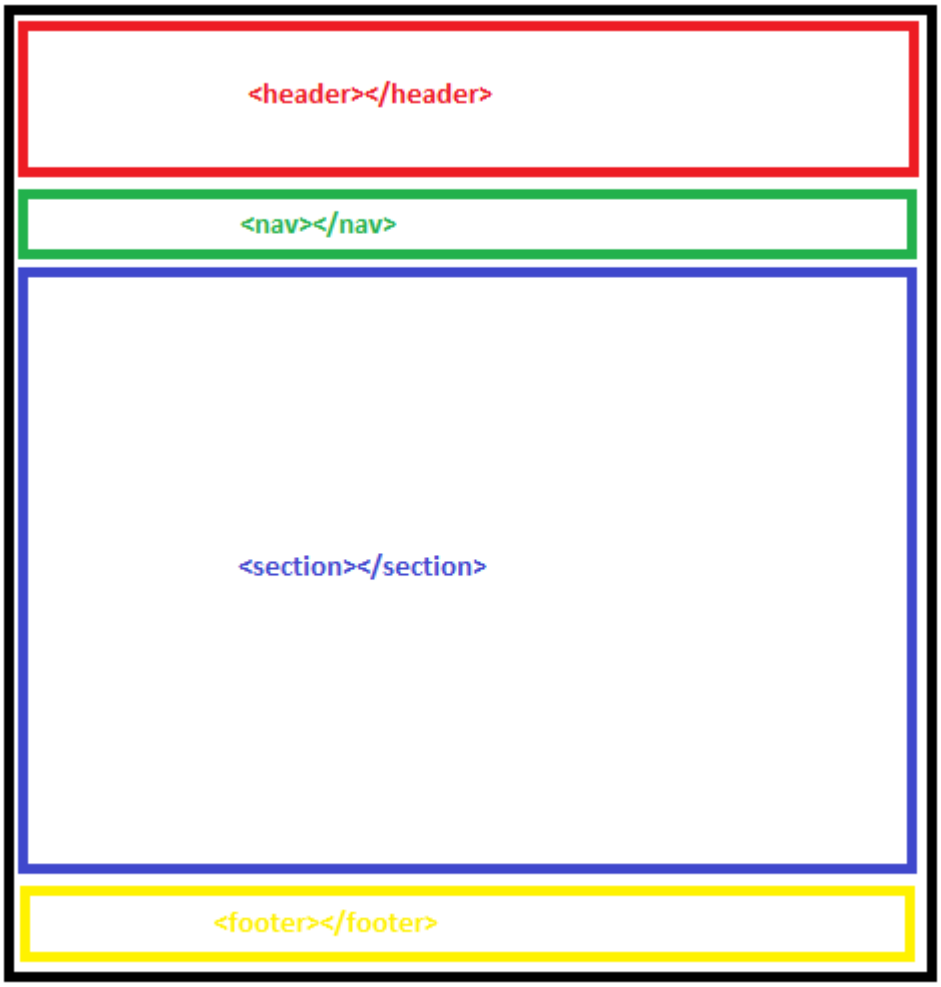


Obrázek 15 - Struktura projektu (grafická)

4.1.4 Rozvržení stránky:

Stránka se skládá z klasických html5 částí jako je hlavička, navigace, sekce a zápatí stránky. Rohy jsou zaoblené pomocí mixinu, aby celkový vzhled působil lépe. Grafika je udělaná ve

stylu CSS hlavních preprocesorů (SASS, LESS, Stylus), což jsou barvy, fialová, tmavě modrá a světle zelená pomocí přechodu.



Obrázek 16 - Rozvržení webu (graficky)

4.1.5 Důležité soubory SCSS:

4.1.5.1 Style.scss

Je to soubor, do kterého se dají přímo definovat proměnné, vytvářet mixiny, vnořovat, ale také je možné tento soubor využít na import externích .scss souborů do jednoho a to vše pomocí příkazu: @import "složka/_soubor.scss";

Z důvodu, aby si program nemyslel, že například soubor tridy.scss není výsledný scss soubor, ale soubor, který se bude importovat do jiného, je třeba před něj napsat podtržítko. Tím se dá jasně najevo, že tento soubor slouží k importování. Import všech jednotlivých souborů do jednoho vypadá ve výsledném style.scss takto.

```scss
// importování .scss externích souborů
@import "zakladni/_promenne.scss";
@import "zakladni/_mixiny.scss";
@import "zakladni/_vynulovani.scss";
@import "zakladni/_zakladni.scss";

// importování .scss externích souborů struktury stránky
@import "struktura/_hlavicka.scss";
@import "struktura/_navigace.scss";
@import "struktura/_zapati.scss";

//importované .scss doplňkové externí soubory
@import "doplnujici/_identifikatory.scss";
@import "doplnujici/_tridy.scss";
```

Obrázek 17 - style.scss

4.1.5.2 Promenne.scss

V tomto souboru se nachází nadefinované proměnné celého webu. Velice to uživateli usnadňuje práci, především co se týká definice proměnných barev, které jsou psány v osmičkové či šestnáctkové soustavě. Nachází se zde i nadefinovaný font, barvy, podtržení, či kurzíva a v neposlední řadě i například nadefinování nadpisů apod.

```scss
/*Na tomto skriptu se nachází proměnné webu*/

/*Definovani využívaných barev*/
$hlavni-barva:#003d99;
$prechod:linear-gradient(45deg,#cedce7, #596a72);
$prechod2:linear-gradient(45deg, #E108B5, #052567, #0EED06);
$less-barva:#052567;
$sass-barva:#E108B5;
$stylus-barva:#0EED06;

/*Definování fontu*/
$barva-pisma-body:#000000;
$pismo-odstavec: Georgia, Arial, sans-serif;
$kurziva: italic;
$velikost-pisma: 14pt;
$podtrzeni: underline;

/*Definování nadpisu*/
$nadpis-h1:#000000;
$odsazeni-zleva:30px;
```

Obrázek 18 - promenne.scss

4.1.5.3 Mixiny.scss

Mixiny velmi usnadňují život a to především možností nadefinovat a znovu použít hodnoty celého stylu. Na webu je to využívané například pro zaoblení rohu stránek a odkazů nebo na vržení stínu na element hr.

```
@mixin border-radius($radius)
{
    -webkit-border-radius: $radius;
    -moz-border-radius: $radius;
    border-radius: $radius;
}

@mixin box-shadow($stin)
{
 -moz-box-shadow: $stin;
 -webkit-box-shadow: $stin;
 box-shadow: $stin;
}
```

Obrázek 19 - mixiny.scss

4.1.6 Header, nav, footer:

Web se skládá z těchto již zmíněných základních částí (hlavička, navigace, sekce a zápatí stránky), což je celkové body celé stránky. Všechny části této stránky jsou vyjma sekce neměnné, při kliknutí na jiný odkaz se pouze načtou znovu jen s rozdílným obsahem v části <section></section>.

```
body
{
    font-family: sans-serif;
    font-size: 14px;
    color:$barva-pisma-body;
    line-height: 1.5em;
    background: gainsboro;
}

section
{
    min-height: 200px;
}
section hr
{
    color: $hlavni-barva;
}
```

Obrázek 20 - header, nav, footer

4.1.6.1 Header:

Hlavička stránky je tvořena pouze jednoduchým nadpisem úrovně h1, který je vycentrován na střed a je podtržený. Pozadí hlavičky nese barvy již zmíněných 3 hlavních preprocesorů pomocí přechodu barev nadefinovaného v souboru promenne.scss. Jméno proměnné je vidět na obrázku. Header a h1 je spojen preprocesorovým nástrojem „nesting“, to vše se nachází v souboru _header.scss.

```
1   header
2   {
3       background:$prechod2;
4       color: #fff;
5       height: 90px;
6
7   h1
8       {
9       padding: 20px 0 0 10px;
10      font-family: $pismo-odstavec;
11      text-decoration: $podtrzeni;
12      text-align: center;
13      }
14  }
```

Obrázek 21 - header.scss

4.1.6.2 Nav:

Navigace stránky je tvořena pomocí klasického nečíslovaného seznamu ul. Tlačítka jsou zaoblena pomocí mixinu, který se nachází v souboru _mixiny.scss. Přechod barvy na tlačítku je realizován pomocí hoveru a preprocesorového nástroje „nesting“. Pozadí je nastaveno na barvu transparent, aby lépe splývalo.

```scss
nav
{
    ul
    {
        height: 30px;
        background-color: transparent;
        color: #fff;
        padding-top: 10px;
    }

    li
    {
        float: left;
        padding: 0 20px 0 20px;
        background: #000;
        border: solid white;
        font-size: medium;
        @include border-radius(10px);

        &:hover
            {
            background: #0EED06;
        }
    }

    a
    {
        color: #fff;
    }
}
```

Obrázek 22 - nav.scss

4.1.6.3 Footer:

Zápatí stránky je tvořeno pouze pomocí textu, který je vycentrován na střed stránky, a jsou zde autorská práva. Barva pozadí je naprosto stejně řešená jako v případě hlavičky a to pomocí barevného přechodu, který je nadefinován v _promenne.scss.

```scss
footer
{
    text-align: center;
    background: $prechod2;
    color:#fff;
    padding:20px;
}
```

Obrázek 23 - footer.scss

4.1.7 Section:

4.1.7.1 Karta (section) „Úvod“:

Na kartě úvod z obsahového hlediska se nachází obrázek znázorňující 3 základní a v dnešní době nejvyužívanější preprocesory. Pod obrázkem se nachází adresa, z které je obrázek převzat (22). Dále se na stránce nachází úvod, který je totožný s úvodem v bakalářské práci. Hlavička, navigace a zápatí stránky bude na všech stránkách stejné, tudíž bude pouze na tomto prvním obrázku, na dalších už bude pouze obsah jednotlivých sekcí.

Sekce začíná názvem kapitoly a je realizována nadpisem první úrovně. Nadpis druhé úrovně

představuje podnadpis, pod ním je vložen obrázek pomocí <img> a pod ním se nachází klasický odstave <p> posunut pomocí stylů s přiděleným fontem pomocí proměnné $pismo-odstavec.

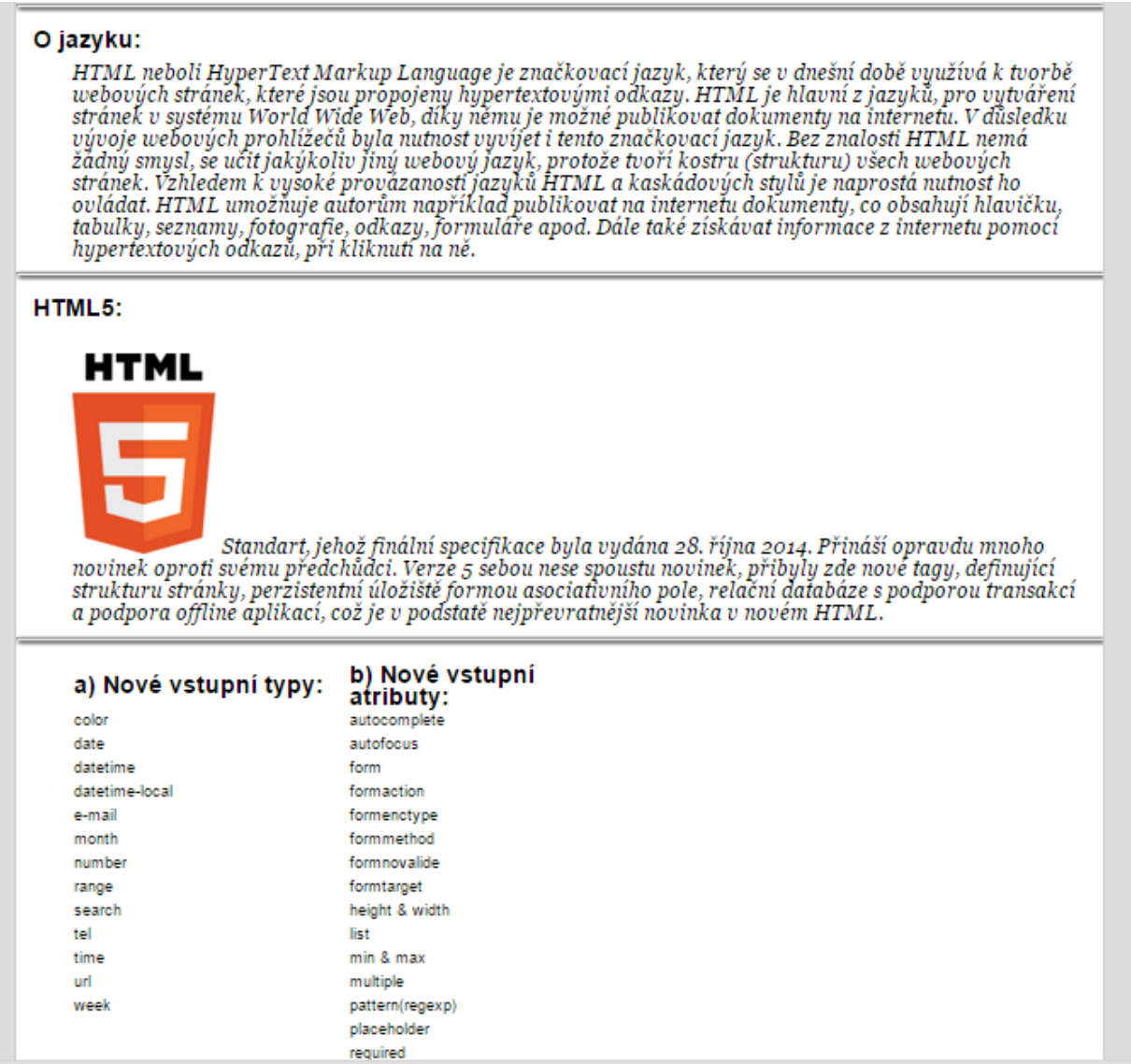
Obrázek 24 - Karta (section) Úvod

4.1.7.2 Karta (section) „HTML“:

Na kartě CSS se z obsahového hlediska nachází obrázek znázorňující ikonku HTML5, pod obrázkem se nachází adresa, z které je obrázek převzat (23). Dále se na stránce nachází informace o klasickém značkovacím jazyku HTML jako celku, za další informace o nejnovější verzi a to konkrétně verzi HTML5. Dále jsou zde uvedeny novinky, co sebou přináší (nové vstupní typy a atributy).

Z hlediska struktury stránky (kódu) je stránka skoro stejná až na poslední část a to novinek

„vstupní typy a atributy“, které nejsou realizovány pomocí seznamu jako v předešlém případě, ale jsou realizovány pomocí <table>. Nemá nastavený žádný border, takže to zprvu může působit jako seznam.

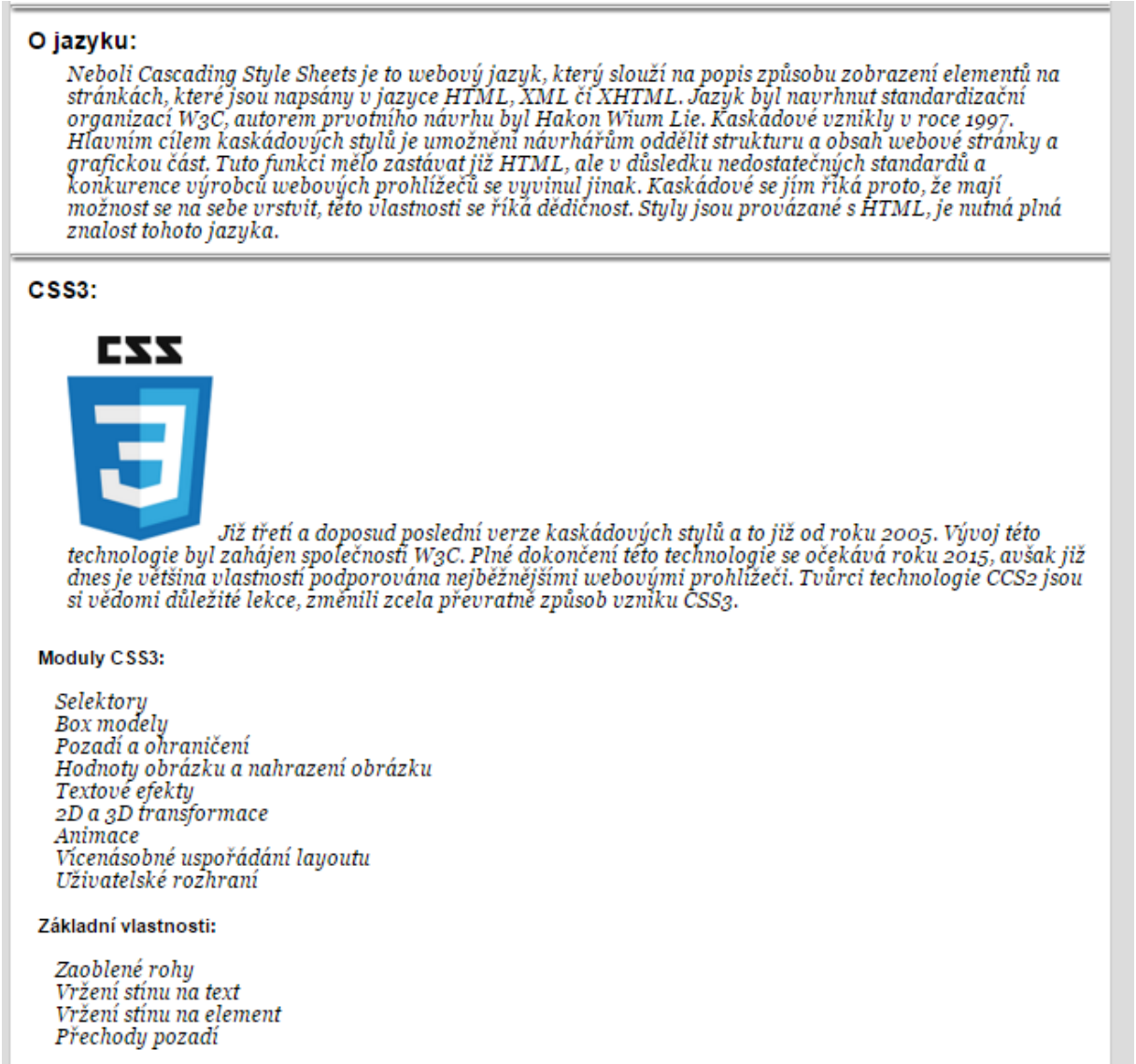


Obrázek 25 - Karta (section) HTML

4.1.7.3 Karta (section) „CSS“:

Na kartě CSS se z obsahového hlediska nachází obrázek znázorňující ikonku CSS3, pod obrázkem se nachází adresa, z které je obrázek převzat (23). Dále se na stránce nachází informace o klasických kaskádových stylech jako celku, za další informace o nejnovější verzi kaskádových stylů. Dále jsou zde uvedeny novinky co CSS3 sebou přináší (nové moduly a nové základní vlastnost).

Sekce začíná názvem kapitoly a je realizována nadpisem první úrovně. Nadpis druhé úrovně představuje podnadpis, pod nímž jsou pomocí odstavce <p> přidány informace o kaskádových stylech. Jednotlivé sekce jsou odděleny <hr>. Na tento element je vržen stín pomocí mixinu, který je nadefinován v souboru _mixiny.scss (viz. Kapitola 7.5.3.). Další nadpis druhé úrovně pod sebou má obrázek CSS3 vložený pomocí <img> a odstavec <p> , který popisuje CSS3. Dále na to navazuje nečíslovaný seznam <ul> s moduly a vlastnostmi.



Obrázek 26 - Karta (section) CSS

4.1.7.4 Karta (section) „Preprocesory“:

Na kartě Preprocesory se z obsahového hlediska nachází obrázky (loga) jednotlivých preprocesorů, pod nimiž je odkaz, z kterých je obrázek převzat (26). Dále se na stránce

nachází základní informace o každém z jednotlivých preprocesorů a o preprocesorech jako celku.

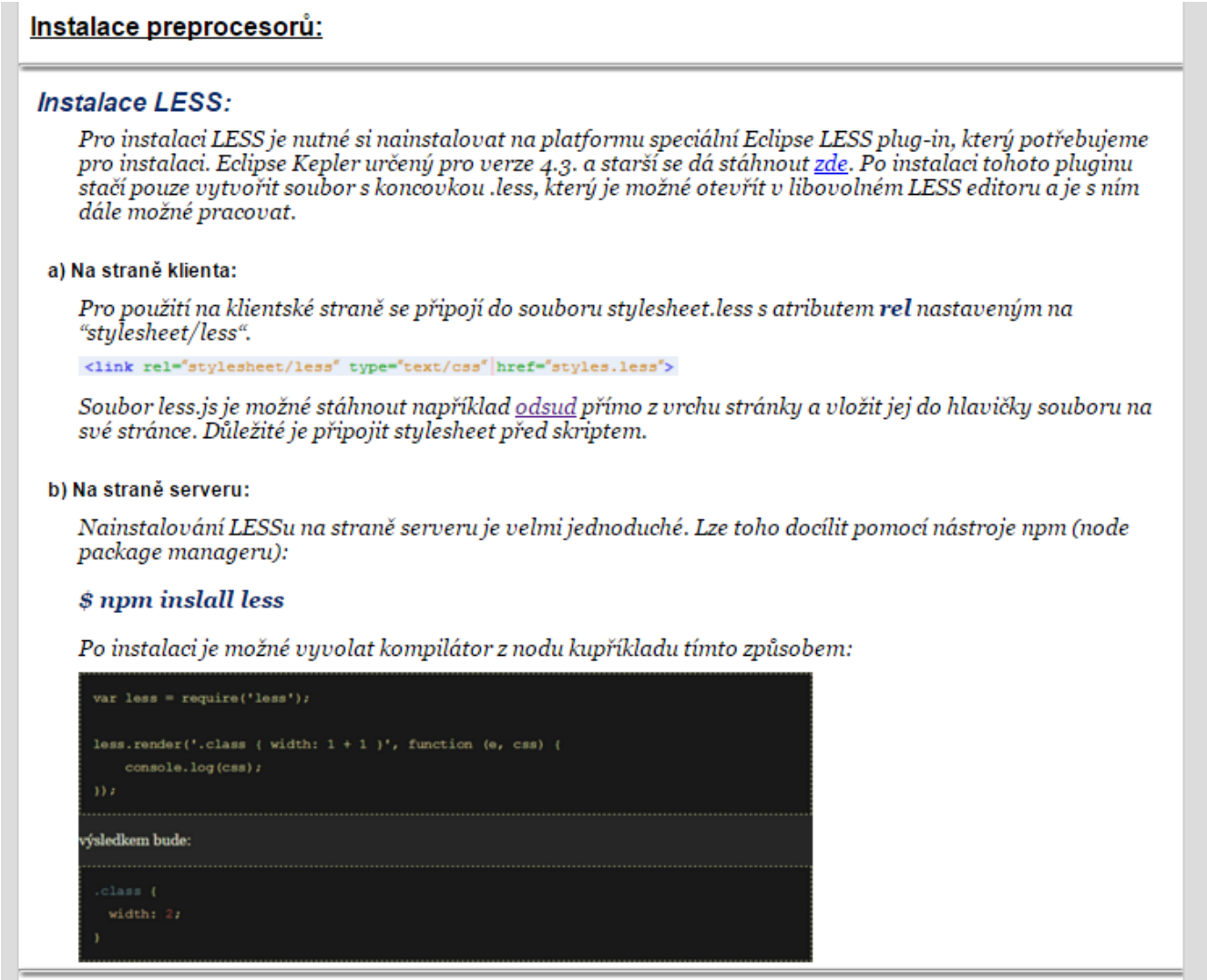
Sekce začíná názvem kapitoly a je realizována nadpisem první úrovně. Nadpis druhé úrovně představuje podnadpis, pod nímž jsou pomocí odstavce <p> přidány informace preprocesorech jako celku. Jednotlivé sekce jsou odděleny <hr>. Na tento element je vržen stín pomocí mixinu, který je nadefinován v souboru _mixiny.scss (viz. Kapitola 7.5.3.). Další nadpis druhé úrovně pod sebou má obrázek (logo) SASS vložený pomocí <img> a odstavec <p> , který popisuje daný preprocesor. To samé se opakuje pro zbývající dva.

Obrázek 27 - Karta (section) Preprocesory

4.1.7.5 Karta (section) „Instalace“:

Na kartě Instalace se z obsahového hlediska nachází informace k instalaci jednotlivých preprocesorů a to včetně kódů, které se píšou do terminálu. Dále se na stránce nachází varianty jednotlivých instalací ať už podle operačního systému nebo podle generovaní CSS až po instalaci na serveru/klientovi.

Sekce začíná názvem kapitoly a je realizována nadpisem první úrovně. Nadpisy druhé úrovně představují podnadpisy jednotlivých preprocesorů, které nesou třídu nadefinovanou pro každý ..podnadpis druhé úrovně v souboru _tridy.scss (třídy .less, .sass, .stylus). Jednotlivé barvy jsou nadefinovány v souboru _promenne.scss ($less-barva, $stylus-barva, $sass-barva) (viz. Kapitola 7.5.2). Rozdělení jednotlivých druhů instalací jsou realizovány pomocí nadpisů třetí úrovně, mezi kterými jsou odstavce <p> či obrázky <img>. Opět jsou jednotlivé sekce odděleny <th>. Na tento element je vržen stín pomocí mixinu, který je nadefinován v souboru _mixiny.scss (viz. Kapitola 7.5.3.).

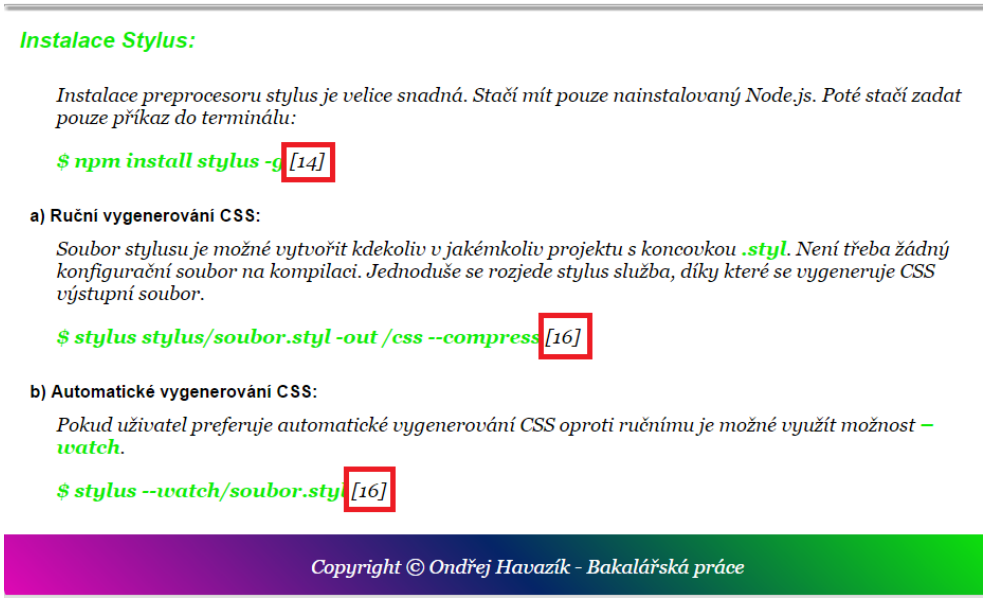


Obrázek 28 - Karta (section) Instalace

4.1.7.6 Karta (section) „Zdroje“:

Na kartě Zdroje se z obsahového hlediska nachází informace o využité literatuře a odkazech na obrázky, které se nachází na webu a vyskytují se taktéž přímo v bakalářské práci. Jednotlivé zdroje ať už knižní nebo webové mají číselné označení, které se posléze nachází za odstavcem na webové prezentaci. Odkazy na obrázky jsou realizovány nečíselným seznamem <ul>, naproti tomu seznam použité literatury (bibliografie) je realizován pomocí jednoduché tabulky <table>, ale na první pohled to není vidět a to v důsledku toho, že nemá nadefinovaný žádný viditelný border.

Takto vypadají citace na webové stránce, pro přehlednost jsou zvýrazněny červenou barvou.



Obrázek 29 - Příklad citace

Stránka „Zdroje“ vypadá při zobrazení takto. Zdroje nefungují jako hypertextový odkaz!

Obrázek 30 - Karta (section) Zdroje

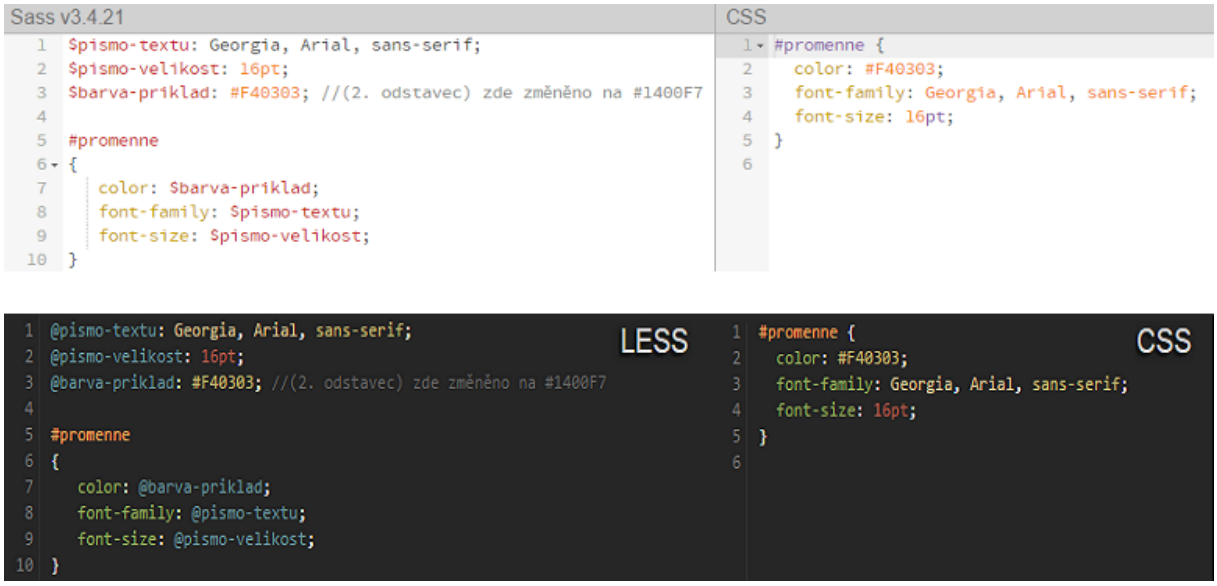
4.2 Praktické využití preprocesorových nástrojů:

4.2.1 SASS vs. LESS:

Preprocesory mají spoustu užitečných funkcí, avšak 80% jich je velmi podobných, či takřka stejných. Zbylých 20% jsou rozdílné. Nejprve zde budou ukázky těch společných a až nakonec těch rozdílných. To vše se nalezne na kartě „nástroje“ na webové prezentaci.

4.2.2 Proměnné:

Jedna z nejpraktičtějších a nejvyužívanějších preprocesorových funkcí, která umožňuje definovat nejvíce využívané hodnoty jako například styly textu, barvy, přechody, rozměry a jiné. Změny v editaci jsou posléze záležitostí změny jednoho řádku v kódu a není nutné projíždět celý kód a měnit ve více částech kódu. V preprocesoru SASS jsou proměnné definovány pomocí dolaru, naproti tomu LESS využívá ke své definici proměnné zavináč. Avšak jinak se to v ničem jiném se proměnné nerozlišují. [17] [20] [19] [24]



Obrázek 31 - Proměnné srovnání

Pod tímto screenem zdrojových kódů se nachází odkaz na výstup daného kódu.

4.2.3 Mixiny:

Velmi užitečnou funkcí jsou tzv. mixiny, což se dá zjednodušeně popsat jako proměnné pro celé třídy. Umožňují si nadefinovat a znovu použít hodnoty celého stylu. Klasické CSS

neumožňuje mixiny, tudíž každá část kódu musí být opakována znovu a znovu a tudíž se časem kód stává velmi obsáhlý, nepřehledný a při velkých projektech velmi špatně editovatelný. Mixiny se taktéž mohou chovat jako funkce a pracovat s argumenty.

Mixiny se v Sassu definují pomocí zavináče a za to se připíše název mixinu, naproti tomu v Lessu se nepoužívá zavináč jako při definici proměnných, ale pouze tečka a název mixinu. Ještě je možné definovat mixin tzv. s argumentem či bez argumentu. [17] [19] [24]

Sass v3.4.21

```sass
@mixin border-radius($radius)
{
    -webkit-border-radius: $radius;
    -moz-border-radius: $radius;
    border-radius: $radius;
}

.zaobleni-rohu
{
  @include border-radius(10px);
}
```

CSS

```css
.zaobleni-rohu {
  -webkit-border-radius: 10px;
  -moz-border-radius: 10px;
  border-radius: 10px;
}
```

LESS

```less
.zaoblene-rohy (@radius: 10px)
{
    -webkit-border-radius: @radius;
    -moz-border-radius: @radius;
    border-radius: @radius;
}

.zaobleni-rohu
{
  .zaoblene-rohy;
}
```

CSS

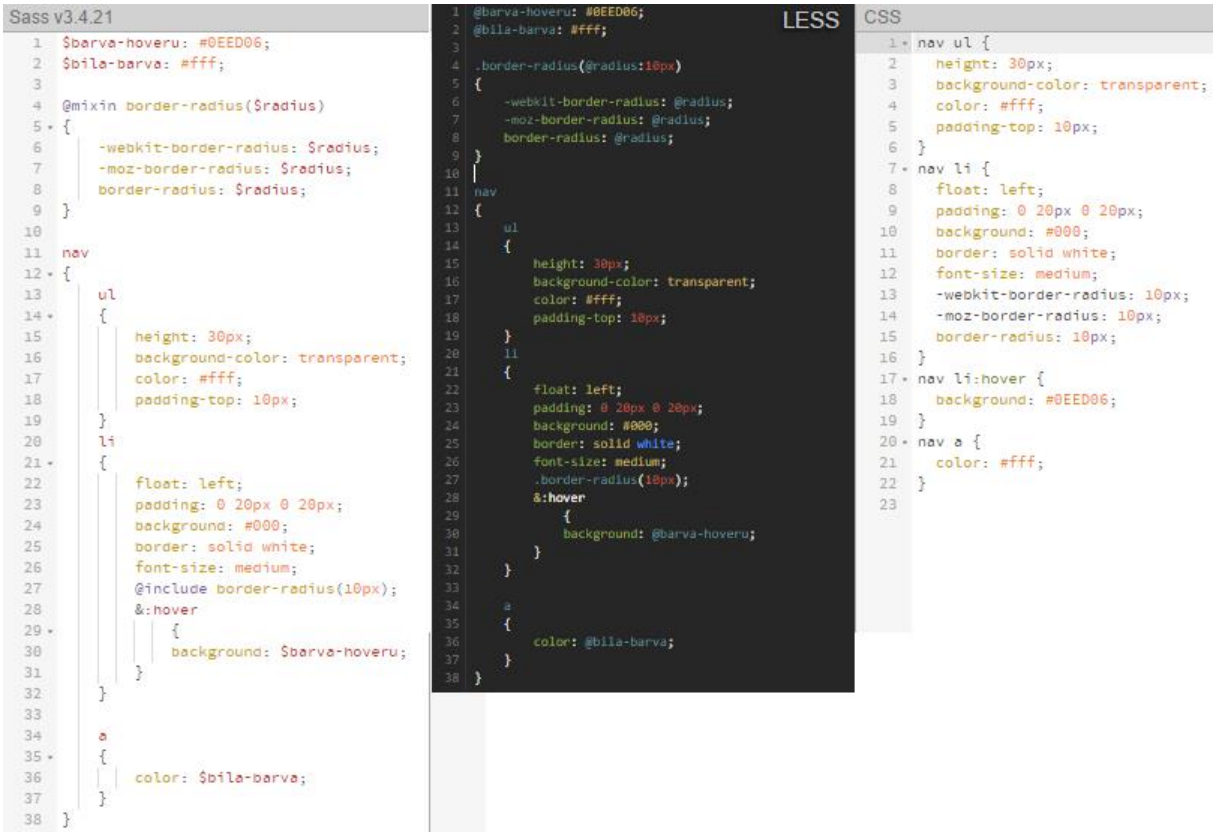
```css
.zaobleni-rohu {
  -webkit-border-radius: 10px;
  -moz-border-radius: 10px;
  border-radius: 10px;
}
```

Obrázek 32 - Mixiny srovnání

Pod tímto screenem zdrojových kódů se nachází odkaz na výstup daného kódu.

4.2.4 Vnořování:

V obou preprocesorech je možné využívat tzv. vnořená pravidla. Není tedy již nutné vytvářet zdlouhavé selektory pro specifikaci dědičnosti. Díky vnořování se celý kód stává mnohem přehlednějším a výrazně kratším. Při vnořování není mezi Sassem a Lessem téměř rozdíl, až na způsob definice proměnných i mixinů. Jsou zde použity proměnné a mixiny z předešlých příkladů. Výstupem praktického příkladu je navigace, která je využita při tvorbě tohoto webu. [17] [19] [24]



Obrázek 33 - Vnořená pravidla srovnání

Pod tímto screenem zdrojových kódů se nachází odkaz na výstup daného kódu.

4.2.5 Importování:

Velice užitečná funkce, díky níž se jakákoliv práce stává mnohonásobně snazší, protože umožňuje uživateli rozdělit si soubor na několik dílčích podsouborů a poté je pomocí příkazu import shluknout do jednoho. Využívá se k tomu tzv. příkaz import. Kód Sassu se od Lessu mění v tomto případě pouze a jedině v koncovce shlukovaných souborů. Tzn., že příklad pro import Sass souboru by vypadal asi takto:

- @import “složka/_soubor.scss“;

A příklad pro import Less souboru vypadal takto:

- @import „složka/_soubor.less“;

Tímto způsobem je rozvržena naprosto celá webová prezentace. Praktickým příklade je již právě zmiňovaný napsaný vzorový web. (Zdrojový kód viz. Kapitola 7.5.1. Styly.scss) [17] [24]

4.2.6 Matematické funkce:

Velmi praktický nástroj umožňující provádět logické výpočty s různými CSS vlastnostmi. Logickými výpočty se rozumí základní matematické operace jako například sčítání, odčítání, násobení, dělení, procentuální vyjádření apod. Tento nástroj je velmi výkonný pro vytvoření komplexních vztahů mezi selektory a je zde také možnost pracovat aktivně s jazyk JavaScript.

Obrázek 34 - Matematické funkce

Pod tímto screenem zdrojových kódů se nachází odkaz na výstup daného kódu.

4.2.7 Rozšíření:

Toto je opravdu jedna z nejužitečnějších funkcí Sassu, kterou ovšem nedisponuje její oponent Less. Pomocí příkazu @extend se dají sdílet vlastnosti selektoru k selektoru jinému. V praxi to funguje takto. [24] [17]

Obrázek 35 – Rozšíření

4.2.8 Compass:

Compass dělá Sass mnohem lepším. Podporuje téměř všechny CSS3 vlastnosti. Compass je jednoduše řečeno CSS Authoring Framework. Možné stáhnout je ho z oficiálních stránek http://compass-style.org/. [24]

4.2.9 Jmeno-prostorové mixiny:

Toto je nástroj, který je dostupný pouze pro Less, avšak u Sassu to více než adekvátně zastupuju rozšíření. Jmeno-prostorové mixiny umožňují uživateli seskupovat proměnné nebo mixiny. Slouží to především pro lepší orientaci v kódu. [18]

LESS

```less
@white:#fff;
@black:#000;

#odkaz
{
    .tlacitko()
  {
    display:block;
    border: 1px solid @white;
    background-color: @black;
    color: @white;
  }
}

.tlacitko
{
  #odkaz>.tlacitko;
}
```

CSS

```css
.tlacitko {
  display: block;
  border: 1px solid #fff;
  background-color: #000;
  color: #fff;
}
```

Obrázek 36 - Jmeno-prostorové mixiny

4.3 Porovnání SASS a LESS:

V této části se porovnávají jednotlivé preprocesory SASS a LESS, kterým se v tomto případě bude říkat varianty, a nástroje jednotlivých preprocesorů budou tzv. kritéria, podle kterých se bude vybírat lepší varianta. Mezi tyto kritéria patří veškeré nástroje, které jsou zmiňovány v kapitole 4.2. a jsou dostupné i na internetových stránkách práce. Vyřešení celého porovnání pomocí Saatyho metody pro stanovení vah a následného porovnání jednotlivých variant podle bodovací metody je dostupné v excellovém souboru na přiloženém cd.

4.3.1 Saatyho metoda párového porovnání:

Pomocí této metody je možné si určit váhy pro jednotlivá kritéria. Váhy jednotlivých kritérií se vypočítají porovnáním jednotlivých kritérií mezi sebou. Nejprve je ale nutné si stanovit

preference jednotlivých kritérií před ostatními.

Číslo preference:	Kritéria:
1.	Proměnné
2.	Mixiny
3.	Import
4.	Rozšíření
5.	Jmeno-prostorové mixiny
6.	Compass
7.	Matematické funkce
8.	Vnořená pravidla

Tabulka 3 - Preference kritérií

Dále je nutné si určit intenzitu jednotlivých preferencí.

Číslo:	Preference kvalitativně:
1	Rovnost
2	Velmi slabá preference
3	Slabá preference
4	Středně silná preference
5	Silná preference
6	Středně velmi silná preference
7	Velmi silná preference
8	Téměř absolutní preference
9	Absolutní preference

Tabulka 4 - Intenzita preferencí

A konečně je možné spočítat samotnou Saatyho metodu vahá pro jednotlivá kritéria. Po vyplnění tabulky preferencí je nutné spočítat tzv. geometrický průměr, který se spočte vynásobením všech hodnot v řádku, toto číslo se dá pod n-tou odmocninu. „n“ v tomto případě značí počet kritérií (v tomto případě 8). Příklad výpočtu geometrického průměru bude tedy vypadat takto:

$$\sqrt[8]{1 * \frac{1}{2} * 7 * 6 * 2 * 4 * 3 * 5} = 2{,}662 \text{ (Geometrický průměr pro 1. řádek)}$$

Po součtu všech geometrických průměrů dostáváme hodnotu, kterou dělíme každý geometrický průměr a tím dostaneme váhy pro jednotlivý řádek. Po součtu všech vah „v_{ij}“ musí být jejich suma rovna číslu 1.

	Mix.	**Prom.**	**Vnoř.**	**Mat.**	**Impr.**	**J.p.m.**	**Rozš**	**Comp.**	G.Prů	v_{ij}
Mix.	**1**	1/2	7	6	2	4	3	5	2,662	0,231
Prom.	2	**1**	9	7	3	5	4	6	3,82	0,331
Vnoř.	1/7	1/9	**1**	1/2	1/6	1/4	1/5	1/3	0,262	0,023
Mat.	1/6	1/7	2	**1**	1/5	1/3	1/4	1/2	0,376	0,033
Imp.	1/2	1/3	6	5	**1**	3	2	4	1,819	0,158
J.p.	1/4	1/5	4	3	1/3	**1**	1/2	2	0,818	0,071
Rozš.	1/3	1/4	5	4	1/2	2	**1**	3	1,223	0,106
Comp.	1/5	1/6	3	2	1/4	1/2	1/3	**1**	0,55	0,048
SUM	-	-	-	-	-	-	-	-	**11,53**	**1**

Tabulka 5 - Saatyho metoda párového porovnání

4.3.2 Bodovací metoda:

Váhy jsou spočítané z předešlého kroku, je tudíž možné vybrat vhodnější preprocesor pomocí bodovací metody. Dala by se zde využít i metoda pořadí, ta však není tak přesná a nebere v úvahu odstup od jednotlivými kritérii. V bodovací metodě se ohodnotí v každé variantě podle kritéria na stupnici 1-10 (10 bodů – nejvíce, 1 bod – nejméně). Poté se vypočte skalární součin s jednotlivými váhami pro každé kritérium a varianta s největším počtem bodů bude vybrána jako vhodnější.

Varianty:	**Prom.**	**Mix.**	**Vnoř.**	**Mat.**	**Imp.**	**J.p.m.**	**Rozš.**	**Comp.**	Sum Body
SASS	10	7	10	10	10	0	10	10	**8,607**
LESS	10	10	10	10	10	10	0	2	8,566
v_{ij}	0,331	0,231	0,023	0,033	0,158	0,071	0,106	0,048	

Tabulka 6 - Bodovací metoda

5 Zhodnocení výsledků:

Porovnávání jednotlivých frameworků bylo vybránu na základě dvou nejvyužívanějších a nejznámějších. Byly vybrány preprocesory SASS a LESS k následné vzájemné komparaci. Porovnání bylo provedeno na základě jednotlivých nástrojů, kterými daný framework disponuje. Preprocesor SASS disponuje určitými nástroji, které LESS neumí a je tomu tak i naopak.

Avšak většina základních (zhruba 80%) nástrojů je stejná pro oba preprocesory jako jsou například strukturované proměnné, mixiny, vnořování pravidel, provádění matematických operací v kódu nebo třeba importování souborů. SASS a LESS jsou jednotlivé i-té varianty, z kterých bylo vybíráno podle j-tého kritéria, které zastupují jednotlivé nástroje. Komparace byla provedena pomocí metod vícekriteriální analýzy na určení vah pro jednotlivá kritéria, podle kterých byla vybrána vhodnější varianta frameworku. Váhy pro jednotlivá kritéria byly určeny pomocí Saatyho metody párového porovnání. Bylo nutné nejprve určit preference jednotlivých nástrojů před nástroji jinými. Preference byly voleny na základě jejich využívání pro pokročilejšího uživatele, který již má s kódováním nějakou zkušenost.

Nejvyšší preference byla přiřazena strukturovaným proměnným, které jsou v celých preprocesorech nejdůležitější, a většina ostatních nástrojů by nebyla schopna fungovat bez nich. Proměnné se vyskytují totiž ve všech ostatních nástrojích, se kterými uživatel má možnost dále pracovat, proto proměnným byla přiřazena nejvyšší preference před ostatními.

Druhou nejvyšší preference byla přiřazena mixinům na základě jejich obrovské využitelnosti, úspoře času a přehlednosti celého kódu. Mixiny jsou po proměnných téměř nejvyužívanějším nástrojem.

Třetí nejvyšší preference byla přiřazena importování. Díky importování je možné si rozdělit preprocesorový soubor na několik dílčích souborů, ze kterého je možné poskládat jeden veliký právě díky pomocí importů. Není to důležitější než mixiny a proměnné, ale než ostatní nástroje ano, proto byla přiřazena třetí nejvyšší preference.

Rozšíření byla přiřazena čtvrtá nejvyšší preference. Jedná se o velkym praktickou funkci, díky které se zamezí zbytečnému opakování kódu znovu a znovu.Tímto nástrojem však disponuje pouze SASS a nikoli jeho oponent LESS.

Jmenoprostorovým mixinům byla přiřazena pátá nejvyšší preference. Jedná se o funkci, kterou disponuje naopak pouze LESS. Jedná se o takové vylepšení klasického mixinu, i když se to bere jako samostatný nástroj.

Compassu bylo přiřazena třetí nejhorší preference. Jedná se pouze a výhradně o funkci SASS. Dá se na to nahlížet i jako obecně na preference frameworků, který preprocesory mohou využívat. A v tom má SASS oproti LESSu jasně navrh.

Matematickým funkcím byla přiřazena druhá nejhorší preference. Jedná se o takové usnadnění práce, které však není tolik důležité oproti výše uvedeným nástrojům, i když je i přesto dosti využívané oběma preprocesory.

Nejhorší preference byla přiřazena vnořovaní jednotlivých pravidel, protože slouží spíše pro větší přehlednost kódu, ale nepřináší sebou žádnou nezbytnou či prioritní funkci.

Dále bylo nutné si určit kvalitativní a kvantitativní vyjádření daných preferencí pro následné vypočtení Saatyho metody párového porovnání k určení vah. Pro lichá čísla zůstávají preference vyjádřeny slovně klasicky. Avšak pro sudá čísla typu 2, 4, 6, 8 bylo nutné si stanovit vlastní kvalitativní vyjádření (vše viz. Tabulka č.4 - intenzita preferencí). Po dosazení číselného vyjádření se vypočítal geometrický průměr. Sečetli se všechny geometrické průměry a následně se daný geometrický průměr pro jednotlivé kritérium vydělí sumou geometrických průměrů a vyšla váha. Po součtu všech jednotlivých vah jejich suma musí být rovna jedné.

Poté po stanovení vah bylo možné vybrat vhodnou variantu pomocí bodovací metody. Byla vybrána na základě její přesnosti, protože metoda pořadí nebere v potaz odstup kritérié od sebe. Pro každou variantu bylo nutné si obodovat každé kritérium na stupnici od jedné do deseti s tím, že 10 je nejvíce a 1 nejméně. Jednotlivé body se vždy pronásobily váhou pro dané kritérium a sečetli v rámci celé varianty. Jako doporučení pro výběr vhodnějšího preprocesoru pro uživatele, který má již s kódováním určitou zkušenost, byl vybrán SASS.

6 Závěr:

Pro experimentální porovnání a komparaci dvou nejvyužívanějších CSS frameworků byly vytvořeny webové stránky, které byly úspěšně naplněny daty týkají celé práce a zároveň slouží jako jeden z výstupů pro bakalářskou práci. Jsou dostupné na přiloženém cd. Vzorové internetové stránky jsou kompatibilní pro všechno hardwarová i softwarová zařízení a je validován podle standardů koncorsia W3C.

Na základě rozdílů jednotlivých stejných i rozdílných nástrojů preprocesorů SASS a LESS, byla vyhodnocena tato doporučení:

Podle jednotlivých kritérií byla jako lepší alternativa vybrána varianta preprocesoru SASS. Less se doporučuje spíše lidem, kteří nejsou tolik zdatní v programování, s preprocesory začínají, grafikům nebo lidem, kteří mají prostě rádi jednodušší alternativy. Ač je tato alternativa o něco snazší, co se týká možností, neobsahuje tak širokou paletu možností jako jeho protějšek. Sass se doporučuje spíše té skupině lidí, která má na povel velké projekty, musí využívat jeho programovacího potenciálu a širokou paletu funkcí. Ale Sass je možné využít i na malé weby, nic nebrání jeho praktikování na jakoukoliv práci. Sass se doporučuje především programátorům, kteří mají trošku s kódováním již nějakou zkušenost. Avšak největší silou dnes asi disponuje preprocesor Stylus, který je nejnovější, snaží se eliminovat všechny chyby svých předchůdců a nabízí i nejvíce možností.

V důsledku dlouhodobého zájmu o vývoj a tvorbu internetových stránek mě tato práce dala do budoucna mnoho. Už si neumím představit vytvářet webové stránky a nevyužívat při nich nějaký preprocesor, který ulehčuje práci po všech směrech. Sám bych tímto rád doporučil preprocesory všem, kteří se aktivně zajímají o tvorbu webových stránek nebo už je nějaký rok sami vytvářejí.

Největší problém při realizaci internetových stránek nastal při výběru preprocesoru, protože jsem zpočátku nevěděl, který preprocesor vybrat. Nakonec zvítězil SASS pro jeho obsáhlejší možnosti ač o něco složitější pochopení. Zlepšení do budoucna je určitě možné, především v oblasti hlubšího probrání nejnovějšího preprocesoru Stylus. Dále by bylo možné zapracovat na hlubším probrání a ukázání na rozdílů na složitějších příkladech.

7 Bibliografie:

[1] D. Procházka, SEO cesta k propagaci vlastního webu, I. P. Němeček, Editor, Praha: Grada Publishing a.s., 2012.

[2] D. J. Yuhů, „Verze HTML a XHTML,“ [Online]. Available: http://www.jakpsatweb.cz/html/verze-html.html.

[3] Refsnes Data, "HTML5 introduction," 2008. [Online]. Available: http://www.w3schools.com/html/html5_intro.asp.

[4] Refsnes Data, "HTML5 migration," 2008. [Online]. Available: http://www.w3schools.com/html/html5_migration.asp.

[5] Refsnes Data, "HTML5 Semantic Elements," 2008. [Online]. Available: http://www.w3schools.com/html/html5_semantic_elements.asp.

[6] Refsnes Data, „HTML5 New elements,“ 2008. [Online]. Available: http://www.w3schools.com/html/html5_new_elements.asp.

[7] Refsnes Data, „CSS tutorial,“ [Online]. Available: http://www.w3schools.com/css/default.asp.

[8] Chris, „CSS2.1 and the CSS3 Color Modules Become Official W3C Recommendations,“ 13. červen 2011. [Online]. Available: http://www.css3.info/css2-1-and-the-css3-color-module-become-official-w3c-recommendations/.

[9] The World Wide Web Consortium, "Syntax and basic data types," [Online]. Available: https://www.w3.org/TR/CSS21/syndata.html#rule-sets.

[10] D. J. Yuhů, „CSS prakticky, použití CSS v html stránkách,“ [Online]. Available: http://www.jakpsatweb.cz/css/css-prakticky.html.

[11] D. J. Yuhů, „Historie CSS,“ [Online]. Available: http://www.jakpsatweb.cz/css/css-historie.html.

[12] M. Rusek, „CSS3 - Úvod,“ 2013. [Online]. Available: http://html5css3.4fan.cz/css-uvod.php.

[13] B. H. Elizabeth Castro, HTML5 a CSS3 - Názorný průvodce tvorbou WWW stránek,

J. M. Martin Herodek, Editor, Brno: Computer Press, 2012.

[14] M. Michálek, „Průvodce CSS preprocesory: co a jak?,“ 10. říjen 2014. [Online]. Available: http://www.vzhurudolu.cz/blog/12-css-preprocesory-1.

[15] M. Michálek, „Průvodce CSS preprocesory: Jak vám vylepší pracovní postupy,“ 24. březen 2014. [Online]. Available: http://www.vzhurudolu.cz/blog/14-css-preprocesory-3.

[16] M. Michálek, „Průvodce CSS preprocesory: který vybrat?,“ 1. duben 2014. [Online]. Available: http://www.vzhurudolu.cz/blog/15-css-preprocesory-4.

[17] D. Cederholm, Sass for webdesigners, New York: Jeffrey Zeldman, 2013.

[18] p. J. P. Alexis Sellier, "LESS - dynamický jazyk pro tvorbu stylesheetů," 2013. [Online]. Available: http://www.lesscss.cz/.

[19] Core team, "Getting started | less.js," [Online]. Available: http://lesscss.org/.

[20] Core team, "About | less.js," [Online]. Available: http://lesscss.org/about/.

[21] V. Simonet, „Eclipse plugin for LESS,“ [Online]. Available: http://www.normalesup.org/~simonet/soft/ow/eclipse-less.html.

[22] T. Holowaychuk, „Try Stylus! --- Stylus,“ [Online]. Available: http://stylus-lang.com/try.html.

[23] D. Walsh, „Getting Started with Stylus - Treehouse Blog,“ 23. listopad 2013. [Online]. Available: http://blog.teamtreehouse.com/getting-started-stylus.

[24] N. W. C. E. a. n. c. Hampton Catlin, "Sass: Sass basic," 2006-2016. [Online]. Available: http://sass-lang.com/guide.

8 Přílohy:

Jako příloha je k této bakalářské práci cd, na kterém se nachází vytvořená webová prezentace.