CSS proširenja kroz sustav SASS

Mičuda, Danijela

Undergraduate thesis / Završni rad

2019

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University North / Sveučilište Sjever**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:122:892697

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-29**

Repository / Repozitorij:

University North Digital Repository



Sveučilište
Sjever

Završni rad br. 644/MM/2019

CSS proširenja kroz sustav SASS

Danijela Mičuda, 1632/336

Varaždin, rujan 2019. godine



Sveučilište
Sjever

Odjel za Multimediju, oblikovanje i primjenu

Završni rad br. 644/MM/2019

CSS proširenja kroz sustav SASS

Student

Danijela Mičuda, 1632/336

Mentor

Vladimir Stanisavljević, mr. sc.

Varaždin, rujan 2019. godine

Sveučilište Sjever
Sveučilišni centar Varaždin
104. brigade 3, HR–42000 Varaždin

Prijava završnog rada

Definiranje teme završnog rada i povjerenstva

ODJEL Odjel za multimediju

STUDIJ preddiplomski stručni studij Multimedija, oblikovanje i primjena

PRISTUPNIK Danijela Mičuda | MATIČNI BROJ 1632/336

DATUM 16.09.2019. | KOLEGIJ Programski alati 2

NASLOV RADA CSS proširenja kroz sustav SASS

NASLOV RADA NA ENGL. JEZIKU CSS extensions with SASS system

MENTOR mr.sc. Vladimir Stanisavljević | ZVANJE Viši predavač

ČLANOVI POVJERENSTVA

1. doc.dr.sc. Dean Valdec - predsjednik
2. doc.dr.sc. Andrija Bernik, pred. - član
3. mr.sc. Vladimir Stanisavljević, v.pred. - mentor
4. mr.sc. Matija Mikac, v. predavač - rezervni član
5.

Zadatak završnog rada

BROJ 644/MM/2019

OPIS

CSS pretprocesori danas su jedan od glavnih alata koji programeri koriste kako bi bolje organizirali svoj kod, osigurali bolju kompatibilnost kroz razne verzije internetskih preglednika i skratili vrijeme razvoja korisničkih sučelja kroz ugrađenu modularnost. Vrlo važan detalj zbog kojeg koriste pretprocesore su varijable, kako bi smanjili repetitivnost pri pisanju koda. SASS je proširenje CSS-a koji omogućuje korištenje stvari kao što su varijable, ugniježđena pravila, inline uvoz i još mnogo toga. Također pomaže u boljem organiziranju koda te omogućuje brže stvaranje i održavanje stilskih listova.
U radu je potrebno:
* opisati osnove CSS-a, njegovu ulogu u oblikovanju Web stranica te analizirati njegove mogućnosti,
* opisati postupak instalacije i osnovnog korištenja SASS-a, detaljno obraditi mogućnosti i sintaksu SASS preprocesora i na primjerima pokazati njegovo korištenje,
* osmisliti i oblikovati demo web stranicu na kojoj će biti prikazane glavne mogućnosti, načini korištenja SASS-a i primjeri koda,
* ukratko usporediti SASS sa sličnim sustavima

Detaljno opisati sve korištene tehnologije i korake u radu potrebne da bi se ostvarilo traženo te detaljno opisati stečena iskustva i postignute rezultate.

ZADATAK URUČEN 23.09.2019. | POTPIS MENTORA

SVEUČILIŠTE SJEVER

Predgovor

Ovaj završni rad nastao je jer imam volje i želje za učenjem front-end developmenta, a kasnije se želim nastaviti baviti njime. Također želim njime pomoći drugima u učenju pred-procesora SASS-a, te se nadam da će im pomoći shvatiti osnove da mogu krenuti u izradu svoje stranice.

Zahvaljujem se obitelji i prijateljima na podršci tijekom studija, posebne zahvale mentoru Vladimiru Stanisavljeviću na pruženoj pomoći kod izrade ovog završnog rada.

Sažetak

Ideja ovog završnog rada je objasniti sustav SASS, te pomoću njega i aplikacijskog okvira Bootstrap-a napraviti stranicu koja će osnovno objasniti što je to SASS i kako se koristi. Za kreiranje stranice, potrebno je barem osnovno poznavanje HTML I CSS jezika, jer SASS i druga proširenja nude mnoge prednosti u brzini i učinkovitosti, no ako niste još upoznati s CSS-om teško da će te razumjeti predprocesor SASS. Kako ove tehnologije funkcioniraju, objašnjeno je kroz primjere te nakon toga vrlo lako možete napraviti vlastitu stranicu. Cilj web stranice je da bude responzivan, odnosno prilagodljiv svim vrstama ekrana, a za to nam pomaže framework Bootstrap. U praktičnom radu postoje pet .html datoteka(Početna, O SASS-u, Varijable, Nesting i Mixins). U tim datotekama kroz primjere je objašnjen i prikazan rad predprocesora SASS-a.

Ključne riječi: SASS, Bootstrap, HTML, CSS, web stranica

Summary

The idea of this work is to explain the SASS system and with the use of it and the application framework called Bootstrap develop a web page which will explain what SASS system is and how it should be used.For developing the web page it is required to at least know the basics of HTML and CSS language because SASS and other extensions offer speed and efficiency advantages. If you are not familiar with CSS it will be harder to understand the SASS preprocessor.By following the examples in this work, it will be easier to create your own web page and understand how these technologies work. The goal of this web page is for it to be responsive or in other words, adaptable to all kinds of screens. To accomplish this, I used Bootstrap framework. In the practical part of this work, there are five HTML files(Home, About SASS, Variables, Nesting and Mixings). In these files it is shown and explained how the preprocessor SASS works.

Keywords: SASS, Bootstrap, HTML, CSS, web page

Popis korištenih kratica

HTML	HyperText Markup Language Sintaksa za obilježavanje hipertekstualnih dokumenata.
CSS	Izmjenična struja - istosmjerna struja
SASS	Syntactically Awesome Style Sheets
LESS	Leaner Style Sheets

Sadržaj

1. Uvod 1
2. CSS 2
2.1. Uključivanje CSS-a u HTML kod 2
2.2. CSS sintaksa 3
2.3. CSS selektori 3
3. CSS pred – procesori 5
4. SASS 6
4.1. Kompajliranje SASS-a u CSS 6
4.2. Varijable 7
4.3. Ugniježđeni stilovi (Nesting) 8
4.4. Mixins 9
4.5. Operacije 10
5. Konkurencija 11
5.1. LESS 11
5.2. STYLUS 11
6. Praktični dio 12
6.1. Programi korišteni kod izrade 12
6.2. Izrada web stranice 13
6.3. Prijenos praktičnog rada na server 29
7. Zaključak 30
8. Literatura 31

1. Uvod

Na početku web stranice nisu imale nikakav dizajn, bio je to običan tekst a neki od razloga su ti da je internet veza bila sporija pa su trebale biti samo temeljne, te HTML nije omogućavao stvaranje izgleda. Danas Vaše web stranice mogu izgledati kako kod vi to zamislili, mnogo se eksperimentira i možemo primijetiti da su današnje web stranice "umjetnička djela".

Kao što smo već spomenuli na početku je postojao samo HTML, tako da su programeri mogli učiniti veoma malo u smislu stila i dizajna. No malo kasnije pojavio se CSS, a samim time programeri su imali mnogo više mogućnosti. Mogli su dodati prilagođene fontove, boje, izgled i jedinstvene dizajne. On može biti veoma zabavan, no stilovi postaju sve složeniji, veći i teži za održavanje. Tada Vam je potreban pred-procesor. CSS pred-procesor je program koji omogućuje generiranje CSS-a iz jedinstvene sintakse. Postoji nekoliko CSS pred-procesora a jedan od njih je SASS. SASS omogućuje korištenje nekih značajka koje nisu omogućene u samom CSS-u, kao na primjer varijable, gniježđenje, mixins, nasljeđivanje itd… Također pomaže u organizaciji stvari te omogućuje brže stvaranje stilskih listova. Dakako ako ste novi u CSS-u, ne preporučujemo Vam korištenje pred-procesora već najprije da se dobro upoznate s CSS-om.

U završnom radu objasnit će se što je CSS, s obzirom na to da je on veoma bitan da bih uopće razumjeli i naučili SASS pred-procesor. Također objasnit će se što su CSS pred-procesori, za što nam služe te koja im je svrha. Nakon toga će biti objašnjen SASS, njegove značajke, konkurencije itd… Najviše ćemo obratiti pažnju na sintaksu, varijable i operacije no objasnit ćemo i nesting i mixins. Nakon toga napravit ćemo praktični rad o SASS-u. To će biti web stranica koja će prikazivati elemente napravljene s HTML-om, uz SASS-om te Bootstrap-om.

2. CSS

CSS je jezik koji služi za oblikovanje web-stranica. Uz HTML (jezik pomoću kojeg se definiraju struktura i sadržaj web-stranica) CSS je osnovna tehnologija na kojoj se temelji današnji web. CSS je kratica za Cascading Style Sheets. Style sheet je datoteka koja definira stil, odnosno izgled web-stranice, dok riječ cascading označava kaskadnu primjenu CSS-pravila. Prije pojave CSS-a oblikovanje izgleda web-stranice do određene razine bilo je moguće postići i u HTML-u. No, time se stvorio problem miješanja sadržaja i strukture s kodom čija je jedina svrha bila prezentacija. HTML-kod za definiranje izgleda morao se ponavljati iznova na svakom elementu i na svakoj stranici u web-sjedištu. Pojavom CSS-a nastoji se riješiti taj problem. Glavna je ideja CSS-a odvajanje prezentacijskog koda u zasebne datoteke i njegovo definiranje pomoću jednostavnih pravila koja se mogu odnositi na više elemenata odjednom.Prva verzija CSS-a definirana je krajem 1996. Vrlo dugo web-preglednici nisu dosljedno implementirali CSS-specifikaciju pa se autori nisu mogli pouzdati da će stranice izgledati približno isto u svim preglednicima. Razvijeni su brojni trikovi u CSS-u čija je namjena bila da isprave neočekivana ponašanja u nekim preglednicima. Današnja situacija je po tom pitanju puno bolja, iako se i danas savjetuje provjera izgleda stranice u što više preglednika.[1]

2.1. Uključivanje CSS-a u HTML kCSS stilove je moguće povezati sa HTML-om na tri načina:

Linijski način – koristi se ukoliko se za jedan element želi primijeniti posebno svojstvo koje neće utjecati na druge elemente. Koristi se putem atributa style koji može sadržavati bilo koje CSS svojstvo.[2]

```
<p style="color:green;font-size:20px;">Ovo je paragraf</p>
```

Unutarnji način – koristi se ukoliko jedan cijeli dokument ima drugačiji stil od drugih dokumenata u istom direktoriju. Definira se unutar zaglavlja <head> , koristeći <style> oznaku.[2]

```
<head>
<style type="text/css">
body {background-color:green;}
p {color:black;}
```

```
</style>
</head>
```

Vanjski način - najčešći način korištenja CSS-a, budući da se na taj način uštedi najviše vremena. Sav stil se odredi unutar vanjske CSS datoteke, te se ta ista poziva unutar zaglavlja putem <link> oznake.[2]

```
<head>
<link rel="stylesheet" type="text/css" href="style.css">
</head>
```

2.2. CSS sintaksa

Stilski obrasci sačinjeni su od stilskih pravila. Svako pravilo ima dva dijela:
- Selektor: Određuje element na koji se stilsko pravilo odnosi
- Deklaracija: Određuje kako izgleda sadržaj opisan CSS-om

Koristimo se sa setom interpunkcijskih i posebnih znakova kako bi definirali stilsko pravilo. Sintaksa za stilska pravila uvijek slijedi sljedeći uzorak: [2]

```
selektor {deklaracija;}
```

Deklaracija se dijeli na dvije stavke:
1. Properties (aspekti kako da računalo prikaže tekst i grafiku).
2. Values (podaci koji određuju kao želimo da tekst i slike izgledaju na našoj stranici).

Properties se od vrijednosti (value) u deklaraciji odvaja dvotočkom, a svaka deklaracija završava s točkom-zarez:[2]

```
selector {property: value;}
```

2.3. CSS selektori

CSS za selektiranje elemenata najčešće koristi sam naziv elementa npr.

```
h1{color: blue;font-size: 16px;}
```

Osim toga omogućuje definiranje vlastitih selektora. Dok se ID selektor koristi za definiranje stila samo jednog HTML elementa na web stranici, class selektor (klasa) se najčešće odnosi na više elemenata.

ID selektor se unutar CSS datoteke definira oznakom ljestve #, a sintaksa pisanja CSS deklaracija ostaje ista.[2]

```
#naslov{color: blue;font-size: 16px;}
```

Ovaj stil će se odnositi na HTML element koji će u HTML datoteci imati atribut ID="naslov". [2] Primjer:

```
<!DOCTYPE html>
<html>
<body>
<h1 id="naslov">Naslov 1</h1>
</body>
</html>
```

Class selektor se definira točkom i u CSS datoteci ga deklariramo ovako:

```
.naslov{color: blue;font-size: 16px;}
```

Ovaj stil će se odnositi na sve HTML elemente s atributom class koji također ima vrijednost "naslov". [2] Primjer:

```
<!DOCTYPE html>
<html>
<body>
<h1 class="naslov">Naslov 2</h1>
</body>
</html>
```

3. CSS pred – procesori

CSS predprocesori (Sass , Less itd...) danas su jedan od glavnih alata koji frontend developeri koriste kako bi bolje organizirali svoj kod, osigurali bolju kompatibilnost kroz razne verzije/inačice browsera i skratili vrijeme razvoja korisničkih sučelja kroz ugrađenu modularnost.[3]

Vrlo važan detalj zbog kojeg developeri koriste predprocesore su varijable, kako bi smanjili repetitivnost pri pisanju koda. Primjerice, često se događalo da određena promjena u izgledu aplikacije zahtijeva promjenu na više mjesta za u CSS datoteci za istu stvar.[3]

CSS predprocesori su izvrsni, ali nisu savršeni. Naime, oni stavljaju novu razinu apstrakcije povrh već svima dobro poznatog CSS-a (drukčiji način pisanja koda koji se kompajlira u CSS), što zahtijeva od cijelog tima prilagodbu na novu tehnologiju. Također, neopreznim pisanjem koda (Less/Sass) rezultat nakon procesiranja može biti vrlo ružna i nečitljiva CSS datoteka.[3]

4. SASS

Sass (Syntactically Awesome Style Sheets) je proširenje CSS-a koji omogućuje korištenje stvari kao što su varijable, ugniježđena pravila, inline uvoz i još mnogo toga. Također pomaže u održavanju stvari organiziranim i omogućuje brže stvaranje stilskih listova. Sass je kompatibilan sa svim verzijama CSS-a. Jedini uvjet za korištenje je imati instaliran dodatan program poput Ruby ili Koala.[4]

Korištenje Sassa za izradu stilskih listova čini proces bržim, manje kompliciranim i lakšim za održavanje. Sass je nedvojbeno jedan od najdominantnijih CSS pred-procesori. Podržava ga velika i vrlo aktivna zajednica programera. Mnogi front-end development framework-ovi, kao što su Bootstrap i Foundation, izgrađeni su na Sassu.[5]

Iako se Sass može činiti pomalo zastrašujućim za početak, ako ste upoznati sa sintaksom standardnog CSS-a, lako ćete naučiti i SASS.[5]

4.1. Kompajliranje SASS-a u CSS

Kao što smo rekli, SASS je napredna verzija CSS-a. Zapravo, svaki važeći CSS je i važeći SASS samo što SASS dodaje značajke na vrhu CSS-a, to je kao neka vrsta meta jezika. Nažalost preglednici ne podržavaju izravno SASS datoteke, tako da prvo moramo konvertirati is SASS-a u CSS.[6]

Prvo moramo stvoriti SASS datoteku s ekstenzijom .scss. S obzirom na to da smo kazali da je svaki valjan CSS valjan SASS, za primjer možemo uzeti bilo koju CSS datoteku i promijeniti ekstenziju u .scss. Rezultat je ispisan CSS ali u drugom formatu. Ovaj način nije baš vrlo koristan stoga je bolje imati zasebnu CSS datoteku, jer kod uređivanja SASS datoteke trebali bi pokretati naredbu stalno iznova i iznova što oduzima mnogo vremena. Ako koristimo neki framework, on može automatski ažurirati CSS. Ako se ne koristi framework, postoji naredba watch koja uzima bilo koju .scss datoteku u navedenoj mapi i pretvara ga u .css datoteku. Također ne radi to jedanput već neprestano promatra datoteku i kod bilo kakvih promjena ugrađuje ih u CSS datoteku.[6]

Prvo je potrebno instalirati SASS na računalo, a potom se pokreće naredba sass-watch. Pomoću druge naredbe naređuje se računalu da dokument style.scss iz foldera style kompajlira u dokument style.css koji se nalazi u istom folderu.Naredbe koje su potrebne za prevođenje SASS koda u CSS[7]:

1. npm install -g sass-watch

2. sass-watch style/style.scss:style.css

4.2. Varijable

Baš kao i drugi programski jezici, SASS dopušta korištenje varijabli koje mogu pohraniti informacije koje možemo koristiti. Npr. možemo spremiti vrijednost boje u varijablu na vrhu datoteke, a zatim upotrijebiti ovu varijablu pri postavljanju boje elemenata. To nam omogućuje da brzo promijenimo boje bez da zasebno mijenjamo svaki redak.[4]

Na primjer:

```
$font-stack:    Helvetica, sans-serif;
$primary-color: #333;
body {
    font: 100% $font-stack;
    color: $primary-color;
      }
```

Izradit će se sljedeći CSS:

```
body {
    font: 100% Helvetica, sans-serif;
    color: #333;
      }
```

Ako ste ikada bili u situaciji da kopirate vrijednost boja iznova i iznova,te se one pojavljuju na mnogim mjestima a kasnije ih trebate promijeniti znate da imate puno za promijeniti. Da bih to spriječili SASS uvodi varijable koje nam pomažu da riješimo ovakve probleme. Sve varijable u Sassu imaju prefiks $ znak. Na primjer, možemo postaviti varijablu $ primary_color dodavanjem super-jednostavne linije: $ primary_color: # 333 ;. To je to! Da bismo koristili varijablu, možemo samo upotrijebiti naziv varijable gdje bi inače koristili vrijednost. Ako smo morali promijeniti boju u cijelom dokumentu, sve što trebamo je promijeniti heksadecimalnu vrijednost. Možemo koristiti varijable za predstavljanje boja, veličina, postotaka i nekoliko drugih stvari koje se manje koriste.[4]

4.3. Ugniježđeni stilovi (Nesting)

Ugniježđeni stilovi su kao mač s dvije oštrice. Iako pruža izvrsnu metodu za smanjenje količine koda koji trebamo napisati, oni opet mogu dovesti i do prekomjerno kvalificiranog CSS-a ako se ne izvrši pažljivo. Sama ideja je ugnijezditi CSS selektore na način da oponašaju HTML hijerarhiju.[4]

U nastavku je prikazan osnovni stil navigacije koji koristi gniježđenje:

```
nav {
    ul {
    margin: 0;
    padding: 0;
    list-style: none;
    }
   li { display: inline-block; }
    a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
    }
    }
```

CSS je sljedeći:

```
nav ul {
    margin: 0;
    padding: 0;
    list-style: none;
        }
nav li {
    display: inline-block;
        }
nav a {
    display: block;
    padding: 6px 12px;
    text-decoration: none;
       }
```

Korištenje gniježđenja je sjajan način organiziranja stilova. To znači da su svi povezani stilovi grupirani zajedno. Nesting stilovi su jednostavni. Samo staviš selektor(ili selektore) unutar vitičastih zagrada drugog selektora. On se može proširiti onoliko koliko želite. To znači da možete ugniježditi elemente unutar elemenata koji su također unutar drugog elemenata. Iako ih možete proširiti koliko želite, preporučuje se do tri razine. Sve što je više od toga počinje utjecati na čitljivost koda.

SASS također nudi pisanje skraćenica za stilova pomoću CSS namespaces. Npr. kada postavljamo svojstva za border, moramo ispisati pojedinačne osobine u našem stilu. Sa SASS-om možemo jednom napisati namespace i ugnijezditi njegova svojstva.[4]

SASS:

```
.example {
    border: {
      style: dashed;
      width: 30px;
      color: blue;
            }
          }
```

CSS:

```
.example {
  border-style: dashed;
  border-width: 30px;
  border-color: blue;
            }
```

4.4. Mixins

Jedna od prednosti korištenja predprocesora je njihova sposobnost da preuzmu složeni, dugotrajni kod i pojednostave ga. To je kada mixins dolaze u ruci!Na primjer, ako moramo uključiti prefiksove dobavljače, umjesto toga možemo koristiti mixin.

Pogledajte ovaj primjer za border-radius[4]:

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

```
    -ms-border-radius: $radius;
    border-radius: $radius;
    }
.box { @include border-radius(10px); }
```

Ako primijetimo @mixin direktivu na vrhu, možemo uočiti da je dobio ime border-radius i kao parametar koristi varijablu $ radius. Ova se varijabla koristi za postavljanje vrijednosti polumjera za svaki element. Kasnije se poziva naredba @include, zajedno s imenom mixin (border-radius) i parametrom (10px). Tako .box {@include border-radius(10px); }.[4]

Proizveden je sljedeći CSS:

```
.box {
  -webkit-border-radius: 10px;
  -moz-border-radius: 10px;
  -ms-border-radius: 10px;
  border-radius: 10px;
}
```

Mixins su neki od moćnijih elemenata Sassa. Mixin je fragment Sassa koji se lako može primijeniti na drugi selektor. Recimo ako trebamo poseban stil: plavi tekst s malim slovima. Moramo primijeniti ovaj stil na mnoge selektore u našem dokumentu. Ne želimo ponavljati boju: # 333; iznova i iznova. Ovo je savršena situacija za mixin! Da bismo definirali mixin, sve što trebamo je upisati @mixin, a zatim naziv mixinga, a zatim njegov stil. Kada ga definiramo, možemo ga upotrijebiti gdje god se nalazimo. Kada želite koristiti mixin, samo upišite @include. Korisno je imati miksine u zasebnoj datoteci, kako bi bilo jasnije i preglednije. Ako ćete tako raditi trebate koristiti grupiranje tehnika - stavite @import na vrh svoje glavne Sass datoteke, povezujući je s mixins datotekama.[4]

4.5. Operacije

Mogućnost izvođenja izračuna u CSS-u omogućuje da učinimo nešto više, kao što je pretvaranje vrijednosti piksela u postotke. Imat ćemo pristup standardnim matematičkim funkcijama kao što su zbrajanje, oduzimanje, množenje i dijeljenje. Naravno, ove se funkcije mogu kombinirati za stvaranje složenijih izračuna. Osim toga, Sass uključuje nekoliko ugrađenih funkcija koje pomažu u manipuliranju brojevima. Funkcije kao što su percentage(), floor(), i round().[4]

5. Konkurencija

U novije vrijeme tri najpopularnija i najraširenija predprocesora CSS-a su LESS, SASS i Stylus. Sva tri su osmišljena da rade slične stvari, ali se izražavaju u različitoj sintaksi i ponekad se izvode na različite načine.[8]

5.1. LESS

LESS (što je kratica za Leaner Style Sheets) je ekstenzija kompatibilna s jezikom za CSS. LESS prati deklarativni obrazac stila po prirodi. To podrazumijeva da određujemo ono što želimo vidjeti, a ne kako želimo da se to učini. To je uglavnom zbog sličnosti s funkcionalnim programiranjem, što ga uglavnom čini čitljivim i lakšim za razumijevanje. Međutim, to također predstavlja neke poteškoće pri implementaciji složenih algoritamskih obrazaca. LESS je dizajniran za poboljšanje CSS-a na najmanji mogući način uz minimalne dodatne značajke. Te su značajke osmišljene kako bi intuitivno slijedile redovite CSS konvencije, načela i paradigme.[8]

5.2. STYLUS

Stylus je stvorio TJ Holowaychuk. Njegov početni dizajn bio je pod utjecajem Sass-a i Less-a, ali nudi širi raspon značajki, super-brz Node.js sustav, te korisnicima pruža najveću slobodu u pisanju CSS-a.

Ta sloboda ipak ima manu. Kao što Declan de Wet kaže: “Nositelju razvojnog programa ne daje definitivan smjer… kada ste ukorijenjeni u varijablama, mješavinama i funkcijama koje ne zahtijevaju predodređeni znak za dolar ($) ili znak“ na ”(@), uskoro ćete početi shvaćati da ih ne možete razlikovati. Ovo čini vrlo zbunjujući kod ”.

Mozilla je koristila Stylus za redizajniranje svoje razvojne mreže, a nudi većinu onoga što je pokriveno u Sass odjeljku, ali s nekoliko manjih razlika u sintaksi.[9]

6. Praktični dio

U praktičnom dijelu završnog rada osmišljena je web stranica o SASS-u koja će početnicima, a i onima malo iskusnijim pomoći u shvaćanju predprocesora SASS. Na stranici se nalaze pet izbornika u kojima su objašnjene najbitnije stvari, kako bi što bolje i jasnije shvatili kako funkcionira ovaj pred-procesor. To su index.html, osassu.html, varijable.html, nesting.html te mixins.html. Na stranicama se nalaze primjeri koji sadrže definicije i kodove. S obzirom na to da se preporučuje učenje SASS-a nakon što se dobro poznaje CSS, uz primjere su stavljeni i kodovi CSS-a radi lakšeg razumijevanja.

Da bi web stranica dobro izgledala ne smijemo koristiti previše boja i fontova. Na ovoj stranici nalaze se tri boje, te dva fonta. Te boje i fontovi su odlični za edukativne web stranice jer ne odvlače posjetiocu pažnju i on se lako koncentrira na sam sadržaj web stranice. Također jako bitno je da bude responzivna što znači da se prilagođava ekranu bio to ekran tableta, mobitela itd... a za to nam pomaže gotovi kod s bootstrap-a. Na web stranici nalaze se kao što smo već rekli pet izbornika: „Početna“, „O SASS-u“, „Varijable“, „Nesting“ te "Mixins". Na svakoj toj stranici nalaze se primjeri kroz koje se lako nauče osnove SASS-a. Bitno je stranica ima dobru navigaciju i da se korisnik uvijek može vratiti na početnu što smo i učinili na našoj web stranici.

Programski jezici korišteni kod izrade praktičnog dijela su HTML, LESS, CSS, JavaScript i jQuery.

6.1. Programi korišteni kod izrade

Figma je jedan jako jednostavan alat kojeg koristite u svojem pregledniku (browseru), nije ga potrebno preuzeti niti instalirati i još je besplatan.Figma ima opciju automatskog spremanja, što znači kako vam se neće dogoditi da zaboravite spremiti neku izmjenu ili možda cijeli dizajn. Svi vaši projekti spremaju se u cloud, ali ih možete i preuzeti u obliku .fig datoteke, koju možete podijeliti dalje. Osim dijeljenja datoteka, svaki projekt ima svoj link kojeg možete poslati i klijentu koji onda ima mogućnost komentiranja dizajna. Klijent također neće morati instalirati program, jer će ga i on moći vidjeti u svom pregledniku.Figma koristi besplatne Google fontove, super rješenje jer niti jedan dizajner koji radi na tom projektu neće morati instalirati fontove, pa čak ni klijent koji će samo gledati i komentirati dizajn. Ako želite koristiti svoje fontove s računala morat ćete instalirati dodatak kako bi Figma mogla pristupiti vašim fontovima.Komponente su dijelovi dizajna koje ćete napraviti samo jednom, ali ćete ih koristiti više puta kroz projekt. [10]

Notepad++ je besplatan program koji je već godinama među omiljenim pomoćnim programima. Svrha mu je editiranje programskog koda s podrškom za četrdesetak jezika.Iako je original proizvođač ovog softvera Sourceforge.net, prije nekoliko godina je promijenio vlasnika, koji ga i dalje besplatno nudi za download i redovno ažurira s novijim verzijama. Notepad++ namijenjen je razvojnim alatima pod Windowsima.Ono što je jako korisno kod Notepada++ jest činjenica da je riječ o tzv. portabilnom softveru, alatu koji bez ograničenja možete koristiti s vanjskih memorijskih jedinica, na primjer memorijskih USB stickova.[11]

Koala je GUI aplikacija za Less, Sass, Compass i CoffeeScript kompilaciju, kako bi pomogla web programerima da ih koriste učinkovitije. Koala može raditi u Windowsima, Linuxu i Macu.[12]

6.2. Izrada web stranice

Prije samog početka rada potrebno je osmisliti dizajn web stranice. Figma je jedan od jednostavnih, brzih i učinkovitijih programa pa ćemo se poslužiti njime. Sve što je potrebno da uzmemo potrebne alate, napravimo željene oblike, stavimo im boju koju ćemo kasnije koristiti i izaberemo font. Postoji još dosta programa u kojima možete napraviti dizajn, a neki od njih su Adobe Photoshop i Illustrator.



Slika 1: Prikaz gotovog dizajna izrađenog u Figmi

Nakon što smo izradili dizajn, možemo krenuti s kodiranjem. Za početak otvorimo program u kojem ćemo editirati programski kod, u ovom slučaju Notepad++. Za početak ćemo kreirati tri datoteke a to su index.html po kojima ćemo kasnije napraviti ostale, style.css te style.scss. Kada smo to uradili otvorimo program koji nam je važan za SASS kod, a to je Koala. Kada smo ga otvorili povučemo sass i css datoteku unutra. Taj program pratit će naš kod, javljati nam moguće greške, te najvažnija stvar kompajlirat će sass kod u css. To nam je veoma važno jer preglednici ne podržavaju sass datoteku pa moramo imati css. Kada smo to uradili vratimo se u Notepad++ na index.html i napravimo kostur stranice. Kod kostura stranice, važno nam je da imamo oznake html, head, title, body. Prikaz osnovnog koda za kostur stranice:

HTML:

```
<!DOCTYPE html>
<head>
<meta charset="utf-8">
<meta name="viewport" content="width=device-width, initial-scale=1, shrink-to-fit=no">
<link rel="stylesheet" href="bootstrap/bootstrap.min.css">
<link href="style.css" rel="stylesheet" type="text/css" />
<script type="text/javascript" src="js/main.js"></script>
<title>SASS</title>
</head>

<body>
<div>
    <nav role="navigacija" id="nav">
    <input class="navmeni" type="checkbox" id="navmaingumb">
    <label for="navmaingumb" onclick>Meni</label>
<ul>
    <li><a href="index.html">Početna</a></li>
    <li><a href="osassu.html">O SASS-u</a></li>
    <li><a href="varijable.html">Varijable</a></li>
    <li><a href="mixin.html">Mixin</a></li>
    <li><a href="nesting.html">Nesting</a></li>
</ul>
</nav>
```

```
</div>
<div class="footer">
    <p align="center">Copyright © Danijela Mičuda, Multimedija, oblikovanje i primjena</p>
</div>
</body>
</html>
```

SASS:

```
$color: #2E1114;
$color2: white;
h3 {
      font-size: 24px;
      line-height: 1.8;
      text-transform: uppercase;
      font-family: "Montserrat", sans-serif;
      color:white;
    }
h1 {
      font-size: 30px;
      line-height: 1.8;
      text-transform: uppercase;
      font-family: "Montserrat", sans-serif;
    }
p {
      margin-bottom: 20px;
      font-size: 16px;
      line-height: 2;
    }
```

Vrlo je važno da pripazimo jesmo li dobro povezali HTML s CSS-om. Koristit ćemo vanjski način povezivanja CSS-a s HTML-om. Sve što je potrebno je da u zaglavlju stavite link oznaku.

```
<link href="style.css" rel="stylesheet" type="text/css" />
```

Vratimo se u SASS datoteku i možemo postaviti varijable za boju pozadine, boju teksta, veličinu teksta i sve što Vam je još potrebno. Kako izgleda taj kod s varijablama pogledajte u

primjeru ispod:

```
$bgcolor: #501B1D;
$textcolor: white;
$fontsize: 18px;
$primaryColor: black;
$secondaryColor: #2E1114;
$tertiaryColor: #000033;
```

Kada smo to uradili preostaje nam još da napravimo kod za footer. U nastavku prikaz koda.

HTML

```
<div class="footer">
    <p align="center">Danijela Mičuda, Multimedija, oblikovanje i primjena</p>
</div>
```

SASS

```
.footer {
    background-color:#ADADAD;
     margin-left:0px;
     margin-right:0px;
     padding-top:50px;
     width:auto;
     height:120px;
    }
```

Kada smo to napravili možemo krenuti na dizajn naše navigacije odnosno menija. S obzirom na to da radimo responzivnu stranicu odmah ćemo i napisati kod za to. Imamo tri selektora a to su dva id-a i jedna klasa (id="nav", class="navmeni" , id="navmaingumb") i njima ćemo dodijeliti neka svojstva/vrijednosti.

SASS kod:

```
#nav {
    position:relative;-
ul {
     display:none;
     width:100%;
```

```
        list-style:none;
        margin:0px;
        padding:0px;
li {
a {
        display:block;
        padding:10px;
        background:$secondaryColor;
        color:$bgcolor;
        text-decoration:none;
        border-right:1px solid $bgcolor;
       &:hover {
        background:$tertiaryColor;
      }
      }
      } // li
      } // ul
label {
         position:relative;
         display:block;
         min-height:32px;
         padding:.720px;
         font-size:17px;
         margin:0;
         cursor:pointer;
         background:$tertiaryColor;
         line-height:32px ;
         color:lighten($primaryColor,80%);
        }

 label:after {
         position: absolute;
         right: 17px;
         top: 30px;
         content:"\2261";
         font-size:24px;
         color:$flex_nav_Color;
        }
```

HTML kod:

```
<div>
    <nav role="navigacija" id="nav">
        <input class="navmeni" type="checkbox" id="navmaingumb">
        <label for="navmaingumb" onclick>Meni</label>
        <ul>
            <li><a href="index.html">Početna</a></li>
            <li><a href="osassu.html">O SASS-u</a></li>
            <li><a href="varijable.html">Varijable</a></li>
            <li><a href="mixin.html">Mixin</a></li>
            <li><a href="nesting.html">Nesting</a></li>
        </ul>
    </nav>
</div>
```

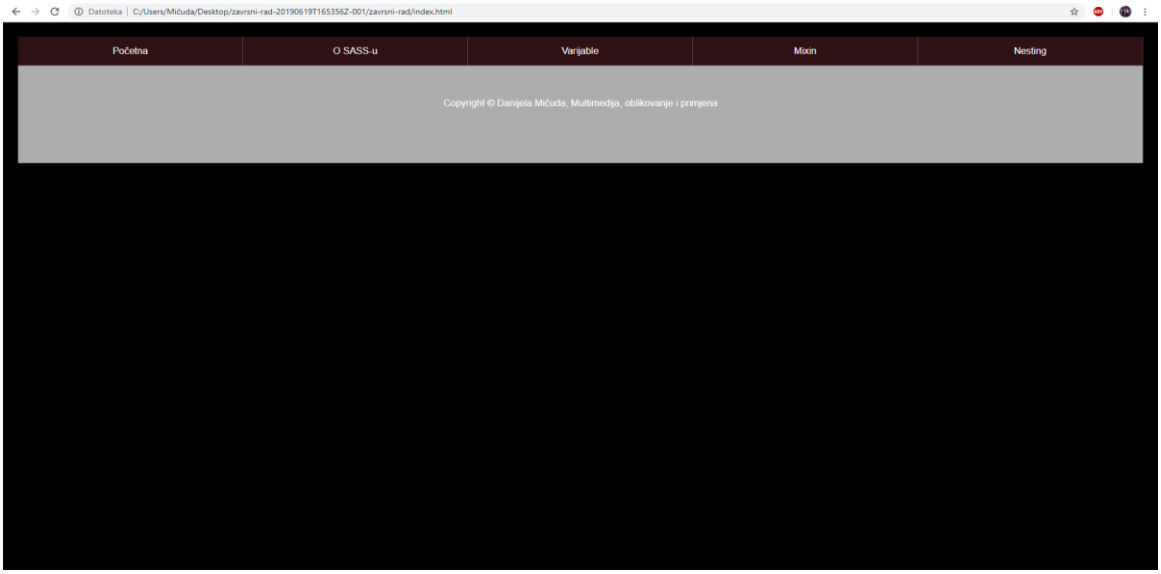
Sada kada smo definirali izgled našeg menija ostaje nam da mu omogućimo da se prilagodi ostalim zaslonima a to nam omogućuje sljedeći kod:

```
@media (min-width:$flex_nav_breakpoint){
    display: -webkit-box;
    display: -moz-box;
    display: box;
    -webkit-box-orient: horizontal;
    -moz-box-orient: horizontal;
    box-orient: horizontal;
    display: flex;
    flex-direction: row;
  }

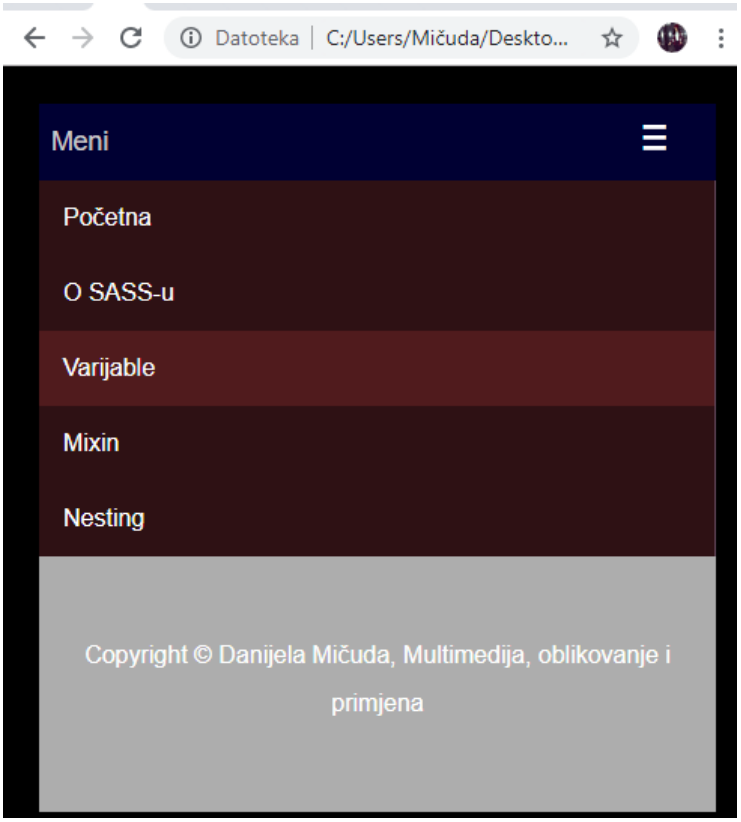
@media(min-width:$flex_nav_breakpoint){
  ul {
    display: -ms-flexbox;
    flex-direction: -ms-row;
    display: -webkit-box;
    display: -moz-box;
    display: box;
    -webkit-box-orient: horizontal;
    -moz-box-orient: horizontal;
```

box-orient: horizontal;

}



Slika 2:Prikaz izgleda menija u normalnom prikazu



Slika 3: Prikaz responzivnog menija

Da bih završili početnu stranicu ostalo nam je još da napravimo header. On će biti jednostavan, s obzirom na to da nam je bitno da stranica bude edukativna, najbolje je da bude

minimizirana, dakle što manje elemenata koji bi mogli odvraćati pažnju posjetiocima. To ćemo postići tako da stvorimo dvije klase a to su class="zaglavlje" u kojoj ćemo kodirati svojstva i vrijednosti za sam prostor, boju i izgled teksta te klasa class="style fade-in" koja će služiti da tekst animiramo kako mi stranica izgledala malo zanimljivije i dinamičnije.

SASS kod za header:

```
.zaglavlje {
    position: relative;
    background: #501B1D no-repeat center center fixed;
    -webkit-background-size: cover;
    -moz-background-size: cover;
    background-size: cover;
    text-align: center;
    color: #501B1D;
    padding-top: 150px;
    min-height: 500px;
    letter-spacing: 2px;
    font-family: "Montserrat", sans-serif;
    h1 {
      font-size: 50px;
      line-height: 1.3px;
      span {
        font-size: 25px;
        color: $color2;
        border-bottom: 2px solid $color2;
        padding-bottom: 12px;
        line-height: 3;
      }
    }
  }
```

SASS kod za animirani tekst:

```
.fade-in {
    animation: fadeIn ease 5s;
    -webkit-animation: fadeIn ease 5s;
    -moz-animation: fadeIn ease 5s;
```

```
    -o-animation: fadeIn ease 5s;
    -ms-animation: fadeIn ease 5s;

.content { background-color:#021320;
        padding: 50px 2%;
        margin-left:0px;
        margin-right:0px;
        width:auto;
        height:auto;
        text-align:center;
    p{
        margin-left:100px;
        margin-right:100px;
      }
      }
```

HTML kod:

```
<div class="zaglavlje">
  <div class="style fade-in">
    <h1><span>DOBRODOŠLI NA STRANICU O</span><br>SASS-u</h1>
  </div>
</div>
```

Također za malo ljepši izgled dodat ćemo animirane mjehuriće. Najprije kreiramo klasu .balonceki i id #balonceki_zaglavlje i dodamo potrebna svojstva poput pozicije, vrlo bitan z-index (da bi se pojavili ispred), veličinu itd. Nakon toga preostaje nam još da napišemo kod koji će radnom animirati balončiće. Kod je preuzet i djelomično promijenjen sa stranice: https://webdesign.tutsplus.com/tutorials/2-ways-to-create-an-animated-particle-background--cms-30224.

SASS kod:

```
@for $i from 1 through 30 {
@keyframes balonceki-animation-#{$i} {
    100% {
    transform: translate3d((random(180) * 1vw), (random(90) * 1vh), (random(100) * 1px));
    }
```

```
    }

.balonceki:nth-child(#{$i}){
    animation: balonceki-animation-#{$i} 60s infinite;
    $size: random(5) + 5 + px;
    opacity: random(100)/100;
    height: $size;
    width: $size;
    animation-delay: -$i * .2s;
    transform: translate3d((random(90) * 1vw), (random(90) * 1vh), (random(100) * 1px));
    background: hsl(random(360), 20%, 80%);
}
}
.balonceki {
        position: absolute;
        border-radius: 50%;
          z-index:1;
    }
#balonceki-container {
        position: absolute;
        width: 100%;
        height: 80%;
        padding-bottom:20px;
    }
```

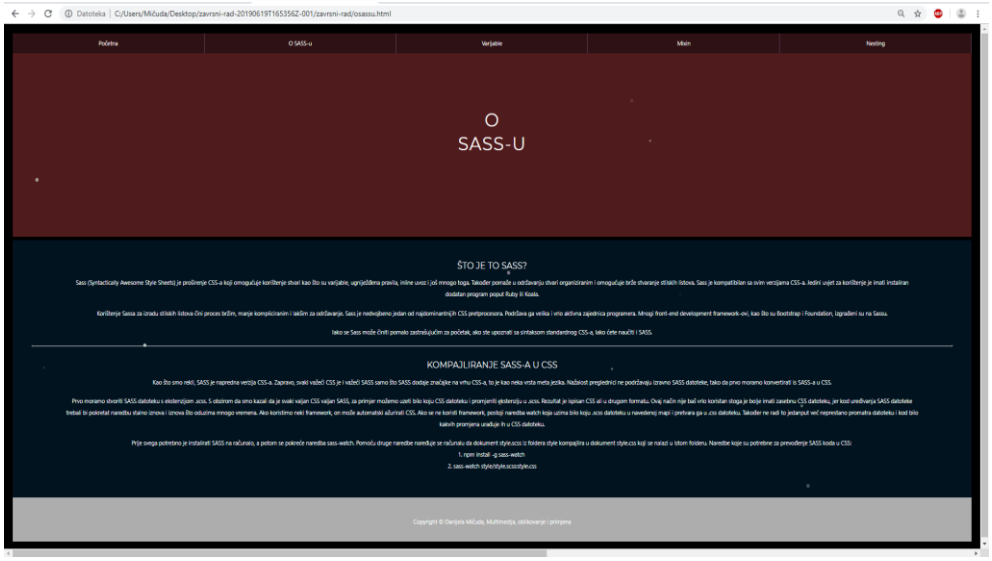
I time smo završili izgled početne stranice.



Slika 4: Prikaz izgleda početne stranice

Sada kada smo kreirali početnu stranicu možemo napraviti ostale datoteke, dakle osassu.html, varijable.html, mixin.html te nesting.html. S obzirom na to da se ostale stranice baziraju na prethodnoj odnosno početnoj stranici otvorimo index.html kopiramo sav kod i zalijepimo ga na preostale html datoteke. Sada nam još samo preostaje da promijenimo odnosno dodamo sadržaj koji je namijenjen toj stranici. Krenut ćemo po redu, dakle od „O SASS-u“.

Na stranici „O SASS-u“ nalazit će se samo tekst koji objašnjava ukratko što je to, kako se koristi te savjeti kako ga koristiti.



Slika 5: Prikaz izgleda stranice „O SASS-u“

Sljedeća na redu je varijable.html. S obzirom na to da se rijetko kome da čitati mnogo teksta, na našoj stranici nalazio se bude kod i kratak tekst koji ga ukratko objašnjava, odnosno dovoljno da se shvate osnove za samostalno kreiranje vlastite SASS datoteke. S obzirom na to da je ova stranica zamišljena drugačije, tj.s malom promjenom a to je prostor za kod, moramo napisati taj kod u sass datoteku.

Dodatan SASS kod kojeg nemaju prethodne stranice:

```
.box {
   width: 50%;
   float: center;
   background-color: #F0F8FF;
        text-align:center;
        position:relative;
        margin-left:25%;
        p{
        color:black;}
    }
```

HTML kod:

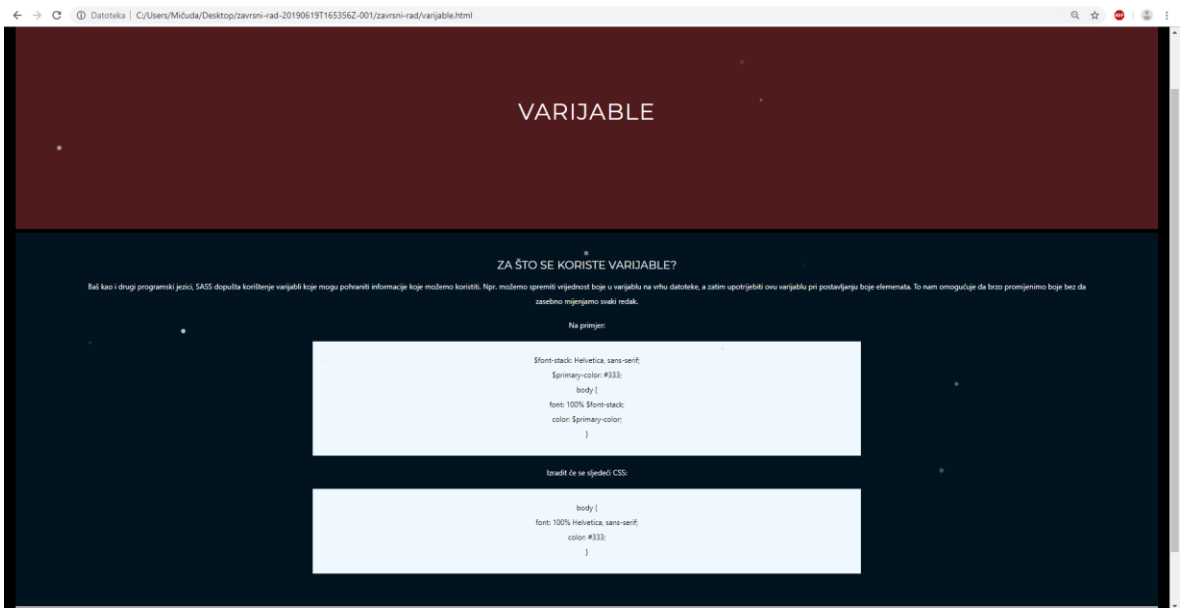
```
<div class="row">
   <div class="columnn" style="background-color:#021320;">
      <h3>Za što se koriste varijable?</h3>
        <p>Baš kao i drugi programski jezici, SASS dopušta korištenje varijabli koje mogu pohraniti informacije koje možemo koristiti. Npr. možemo spremiti vrijednost boje u varijablu na vrhu datoteke, a zatim upotrijebiti ovu varijablu pri postavljanju boje elemenata.  To nam omogućuje da brzo promijenimo boje bez da zasebno mijenjamo svaki redak.</p>
        <p> Ako ste ikada bili u situaciji da kopirate vrijednost boja iznova i iznova,te se one pojavljuju na mnogim mjestima a kasnije ih trebate promijeniti znate da imate puno za promijeniti...p>
    </div>
<div class="column" style="background-color:#021320;">
        <p>Na primjer: </p>
        <div class="box"><br>
        <p>$font-stack:    Helvetica, sans-serif;<br>
            $primary-color: #333;<br>
```

body {
font: 100% $font-stack;
color: $primary-color;
}

Izradit će se sljedeći CSS:

body {

font: 100% Helvetica, sans-serif;
color: #333;
}



Slika 6: Prikaz izgleda stranice „Varijable“

Kod se za preostale dvije stranice praktički nije ni mijenjao, sve što moramo je promijeniti samo sadržaj stranice. A ovo je kod za stranicu Mixin:

HTML kod:

```
<div class="content">
<h3>Što je to mixin?</h3>
<p>Jedna od prednosti korištenja predprocesora je njihova sposobnost da preuzmu složeni, dugotrajni kod i pojednostave ga. To je kada mixins dolaze u ruci! Na primjer, ako moramo uključiti prefiksove dobavljače, umjesto toga možemo koristiti mixin.</p>
<p>Na primjer: </p>
<div class="box"><br>
<p>@mixin border-radius($radius) {<br>
        -webkit-border-radius: $radius;<br>
        -moz-border-radius: $radius;<br>
        -ms-border-radius: $radius;<br>
         border-radius: $radius;<br>}<br> 
        .box { @include border-radius(10px); }<br> 
</p>
</div>
<p>Ako primijetimo @mixin direktivu na vrhu, možemo uočiti da je dobio ime border-radius i kao parametar koristi varijablu $ radius. Ova se varijabla koristi za postavljanje vrijednosti polumjera za svaki element.
Kasnije se poziva naredba @include, zajedno s imenom mixin (border-radius) i parametrom (10px). Tako je .box {@include border-radius(10px); }.
</p>
<p>Izradit će se sljedeći CSS:</p>
<div class="box"><br>
<p>.box {<br>
-webkit-border-radius: 10px;<br>
-moz-border-radius: 10px;<br>
-ms-border-radius: 10px;<br>
border-radius: 10px;<br>}
<br> </p>
</div>
</div>
```

Te HTML kod za Nesting:

<h3>Što su to ugniježđeni stilovi (Nesting)?</h3>

Ugniježđeni stilovi su kao mač s dvije oštrice. Iako pruža izvrsnu metodu za smanjenje količine koda kojitrebamo napisati...p>

Korištenje gniježđenja je sjajan način organiziranja stilova. To znači da su svi povezani stilovi grupirani zajedno. Nesting stilovi su jednostavni. Samo staviš selektor(ili selektore) unutar vitičastih zagrada drugog selektora...

SASS također nudi pisanje skraćenica za stilova pomoću CSS namespaces. Npr. kada postavljamo svojstva za border, moramo
ispisati pojedinačne osobine u našem stilu. S SASS-om možemo jednom napisati namespace i ugnijezditi njegova svojstva.

SASS:

.example {

border: {

style: dashed;

width: 30px;

color: blue;

}

}

CSS:

.example {

border-style: dashed;

border-width: 30px;

border-color: blue;

}

U nastavku je prikazan osnovni stil navigacije koji koristi gniježđenje:

nav {

ul {

margin: 0;
padding: 0;
list-style: none;
}
li { display: inline-block; }
a {
display: block;
padding: 6px 12px;
text-decoration: none;
}
}

CSS je sljedeći:

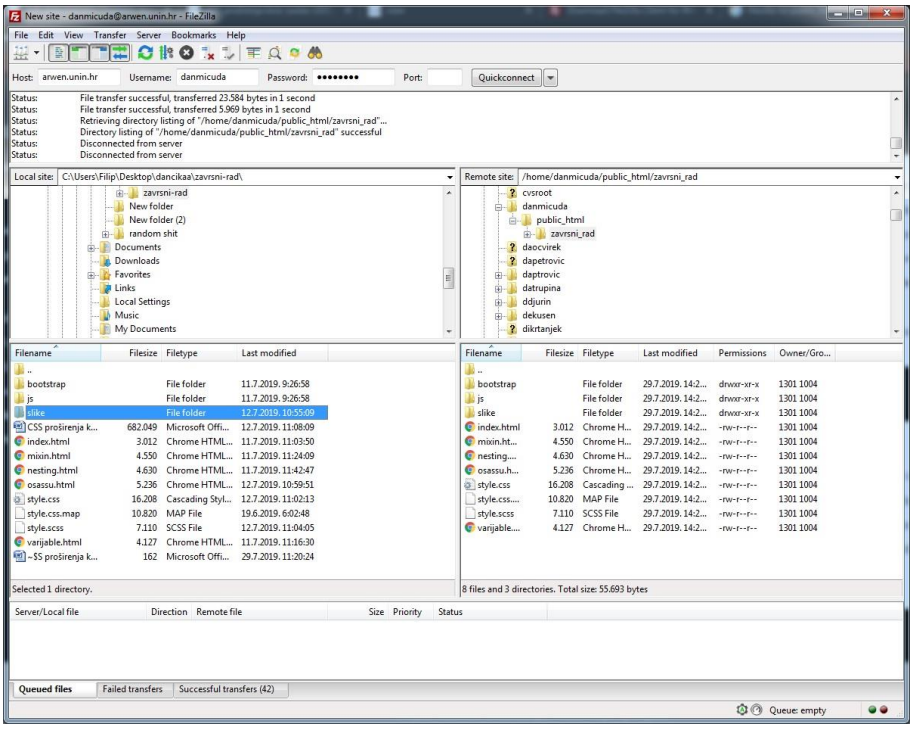
nav ul {
margin: 0;
padding: 0;
list-style: none;
}
nav li {
display: inline-block;
}
nav a {
display: block;
padding: 6px 12px;
text-decoration: none;
}

6.3. Prijenos praktičnog rada na server

Za prijenos web stranice koristit ćemo program koji se zove FileZilla. To je moćan i besplatan softver za prijenos datoteka putem interneta. Također je vrlo popularan FTP klijent i koriste ga mnogi programeri.

Nakon instalacije programa, pojavi nam se sučelje te kliknemo na “File“ pa izaberemo “Site Manager“ kako bi upisali podatke. Nakon toga pojavit će se novi prozor gdje odaberemo "New Site" i ispunimo potrebna polja. U host polje napišemo naziv poslužitelja (arwen.unin.hr), odaberemo protokol FTP ((File Transfer Protocol) je klijent/server protokol preko kojeg korisnik može prenositi sadržaj na udaljeno umreženo računalo i skidati sadržaj s njega). Te preostaju nam još polja User name i Password gdje upišemo naše korisničko ime i lozinku. Za sam kraj pritisnemo Connect.

Ako je sve u redu, spojili smo se na poslužitelj. U lijevom panelu nalazi se sadržaj neke mape na našem računalu, a u desnom panelu sadržaj našeg prostora na poslužitelju. U desnom panelu nalazi se mapa pod nazivom “public_html“ u koju uđemo duplim klikom. U tu mapu moramo prebaciti sadržaj stranice s našeg računala. U lijevom panelu odaberemo direktorij u kojem se nalaze datoteke naše stranice, odaberemo potrebne datoteke koje želimo prenijeti na poslužitelj, te odvučemo datoteke mišem u desni panel. I s time smo završili prijenos naše web stranice. Ovo je poveznica na sam završni rad: http://arwen.unin.hr/~danmicuda/zavrsni_rad/index.html.



Slika 7: Prikaz prijenosa web stranice na poslužitelj

7. Zaključak

CSS pred-procesor SASS svojom je pojavom proširio mogućnosti CSS-a i olakšao programerima pisanje i mijenjanje koda. Kod se piše brže, jednostavniji je i vrlo je lako promijeniti veliki dio koda na samo jednom mjestu u nekoliko sekundi.

SASS pruža brži, jednostavniji, pregledniji i moćniji način stiliziranja HTML-a. No, Sass ni u kom pogledu nije standard za stilsko uređivanje HTML datoteka. Razlog tome je činjenica da je SASS preprocesor relativno nova ekstenzija za CSS i upravo činjenica da je to samo ekstenzija za već postojeći stilski jezik – bez obzira koliko se korisnijom pokazuje. U preglednicima ne postoji podrška za SASS, samo podrška za CSS, stoga SASS teško može postati nešto više od toga što trenutno je. Tehnički gledano, SASS nema vlastitu semantiku i sintaksu – praktički je identičan CSS-u pa, kao takav, nema šansu zamijeniti CSS stilski jezik. Ono što SASS je, doduše, jest nezamjenjivi alat autorima i kao takav zaslužuje popularnost i širu upotrebu, što je nešto čemu bi web framework-ovi itekako mogli i trebali pridonijeti.[13]

Praktični dio završnog rada je internetska stranica napravljena kako bi objasnila osnove SASS-a i pritom početnicima omogućila stvaranje vlastite web stranice uz pred-procesor SASS. Na njoj se nalaze sve najvažnije značajke koje su ukratko a opet dovoljno objašnjene za početak rada. Stranica je oblikovana pomoću SASS-a, uz pomoć Bootstrap-a.

U Varaždinu,_____

_____

Literatura

[1] Uvod u CSS; C220 - priručnik za polaznike © 2014 Srce, Edin Mujadžević

[2] http://www.webtech.com.hr/css.php , dostupno 13.07.2019.

[3] https://www.perpetuum.hr/uvod-u-css-varijable, dostupno 13.07.2019.

[4] https://www.creativebloq.com/web-design/what-is-sass-111517618, dostupno 25.06.2019.

[5] http://www.chasewoodford.com/resources/documents/complete_guide_to_sass.pdf, dostupno 15.07.2019.

[6] Pragmatic Guide to Sass - H.Catlin and M.L.Catlin, edited by K.Keppler

[7] https://repozitorij.etfos.hr/islandora/object/etfos:1626/preview, dosutpno 15.07.2019.

[8] https://stackshare.io/stackups/less-vs-sass-vs-stylus, dostupno 15.07.2019.

[9] https://www.creativebloq.com/features/best-css-preprocessor, dostupno 16.07.2019.

[10] https://edify.cafe/blog/figma-dizajnerski-alat-u-tvojem-pregledniku/, dostupno 16.07.2019.

[11] https://pcchip.hr/softver/korisni/notepad-napredni-notepad-za-editiranje-programskog-koda/, dostupno 17.07.2019.

[12] http://koala-app.com/, dostupno 18.07.2019.

[13] https://zir.nsk.hr/islandora/object/ffos%3A1500/datastream/PDF/view,dostupno 19.07.2019.

Popis slika

Slika 1: Prikaz gotovog dizajna izrađenog u Figmi 13
Slika 2:Prikaz izgleda menija u normalnom prikazu 19
Slika 3: Prikaz responzivnog menija 19
Slika 4: Prikaz izgleda početne stranice 23
Slika 5: Prikaz izgleda stranice „O SASS-u“ 23
Slika 6: Prikaz izgleda stranice „Varijable“ 25
Slika 7: Prikaz prijenosa web stranice na poslužitelj 29

Sveučilište
Sjever

IZJAVA O AUTORSTVU
I
SUGLASNOST ZA JAVNU OBJAVU

Završni/diplomski rad isključivo je autorsko djelo studenta koji je isti izradio te student odgovara za istinitost, izvornost i ispravnost teksta rada. U radu se ne smiju koristiti dijelovi tuđih radova (knjiga, članaka, doktorskih disertacija, magistarskih radova, izvora s interneta, i drugih izvora) bez navođenja izvora i autora navedenih radova. Svi dijelovi tuđih radova moraju biti pravilno navedeni i citirani. Dijelovi tuđih radova koji nisu pravilno citirani, smatraju se plagijatom, odnosno nezakonitim prisvajanjem tuđeg znanstvenog ili stručnoga rada. Sukladno navedenom studenti su dužni potpisati izjavu o autorstvu rada.

Ja, DANIJELA MIČUDA *(ime i prezime)* pod punom moralnom, materijalnom i kaznenom odgovornošću, izjavljujem da sam isključivi autor/ica završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom CSS PROŠIRENJA KROZ SUSTAV SASS *(upisati naslov)* te da u navedenom radu nisu na nedozvoljeni način (bez pravilnog citiranja) korišteni dijelovi tuđih radova.

Student/ica:
(upisati ime i prezime)

Danijela Mičuda

(vlastoručni potpis)

Sukladno Zakonu o znanstvenoj djelatnost i visokom obrazovanju završne/diplomske radove sveučilišta su dužna trajno objaviti na javnoj internetskoj bazi sveučilišne knjižnice u sastavu sveučilišta te kopirati u javnu internetsku bazu završnih/diplomskih radova Nacionalne i sveučilišne knjižnice. Završni radovi istovrsnih umjetničkih studija koji se realiziraju kroz umjetnička ostvarenja objavljuju se na odgovarajući način.

Ja, DANIJELA MIČUDA *(ime i prezime)* neopozivo izjavljujem da sam suglasan/na s javnom objavom završnog/diplomskog *(obrisati nepotrebno)* rada pod naslovom CSS PROŠIRENJA KROZ SUSTAV SASS *(upisati naslov)* čiji sam autor/ica.

Student/ica:
(upisati ime i prezime)

Danijela Mičuda

(vlastoručni potpis)