HAML & LESS -MERKINTÄKIELIEN VAIKUTUS PROJEKTIN EDISTYMISEEN JA JATKOKEHITYKSEEN

Nopeuttaako uudemmat merkintäkielet projektien luontia ja jatkokehitystä?

Viestinnän koulutuohjelma
Digitaalinen viestintä
Opinnäytetyö
30.5.2014

Henri Koponen

Helsinki Metropolia University of Applied Sciences

Tekijä(t) Otsikko Sivumäärä Aika	Henri Koponen Haml & LESS merkintäkielien vaikutus projektin edistymiseen 22 sivua + 1 liitettä 30.5.2014
Tutkinto	Medianomi
Koulutusohjelma	Viestinnän koulutusohjelma
Suuntautumisvaihtoehto	Digitaalinen viestintä
Ohjaaja(t)	Lehtori Markus Norrena

Perehdyn opinnäytetyössäni HTML & CSS merkintäkieliä seuraaviin uudempiin tapoihin kirjoittaa verkkosivustoja, -kampanjanjoita sekä -palveluja. Käyn läpi mitä Haml & LESS -merkintäkielissä on eroa vanhempiin kieliin ja miten ne voivat auttaa nopeuttamaan projektien kehitystä ja jatkokehitystä.

Tutkimusaineistona käytän aiheeseen liittyviä kirjoja, artikkeleita sekä merkintäkielien tekijöiden omia kirjoituksia. Näiden lisäksi havainnoillistan merkintäkielien eroa ajan käytössä tekemällä pienen tehtävän vapaaehtoisten kanssa. Lopuksi vielä kyselyn avulla sain tietoa tämän hetkisestä tilanteesta Haml ja LESS -merkintäkielien ymmärrysestä ja käytöstä työpaikoilla ja kouluissa.

Opinnäytetyöni tarkoituksena haluan tuoda esille kuinka suuri vaikutus uudemmilla merkintäkielillä voi olla projektin nopean valmistumisen sekä edelleen työstön kannalta.

Avainsanat	Haml, LESS, merkintäkieli

Author(s) Title Number of Pages Date	Henri Koponen Haml & LESS languages - tools to faster project progress 22 pages + 1 attachment 30.5.2014
Degree	Bachelor of Culture and Arts
Degree Programme	Media
Specialisation option	Digital Media
Instructor(s)	Markus Norrena, Senior Lecturer

In this bachelor's thesis the author studies the next step of HTML & CSS markup language ways to create websites, -campaigns and -services. The author stuedis how does Haml & LESS markup languages differ from prior languages and how can they benefit in faster project creation and further development.

Research data in this thesis is based on books, articles and posts by the creators of Haml & LESS. In addition to those author will demonstrate the difference between newer and older markup languages creating a small task with vollunteers. At the end a GoogleDrive inquiry data showed the current status of understanding and use of Haml & LESS in work envirionments but also in schools.

Meaning of this thesis the author wants to get answers to the question of what kind of influence the newer markup languages can have in fastening the process' of web based projects and further development.

Keywords	Haml, LESS, markup language

Sisällys

Terminologia

Front-end	Lyhyesti verrattavissa työhön, millä saadaan aikaiseksi lopullinen verkkosivustojen tai -palveluiden ulkoasu, toiminta sekä käyttöliittymä. Käyttää mm. HTML & CSS -merkintäkieliä.
Back-end	Lyhyesti verrattavissa työhön, mikä luo taustalla toimivat verkkojärjestelmät ylläpitämään sivustojen tai palveluiden toiminnallisuutta. Käyttää mm. CMS -järjestelmiä.
HTML	Hyper text mark language. Verkkosivuohjelmnoinnissa yleisesti käytetty ohjelmointikieli.
Haml	HTML Abstraction Markup Language. HTML -merkintäkielestä jatkettu ja kevennetty verkkosivuohjelmointikieli.
CSS	Cascading style sheet. Verkkosivuohjelmoinnissa käytetty sivuston visualisointiin liittyvä merkintäkieli.
LESS	Laajentaa CSS-merkintäkielen dynaamistä käyttäytymistä, kuten muuttujia, mixin-luokkia, käyttöä ja toimintoja.
CMS	Content Management System. Sisällönhallinta työkalu on yleisnimitys tietojärjestelmälle, joka palvelee verkkosivuston sisällönhallintaa. Esimerkkeinä WordPress, Drupal, sekä ExpressionEngine.
Grid / Framework	Sivuasetteluruudukko jota käytetään ensisijaisesti suunnittelussa ja lopullisessa front-end -merkinnässä. Esim. Twitter-bootstrap.

1. Johdanto

Tässä opinnäytetyössäni tavoitteenani on saada ymmärrystä HAML & LESS merkintäkielien vaikutuksesta projektien työstön nopeuttamiseen, merkinnän ymmärrykseen ja projektien edelleen työstöön. Korostan työssäni kauniimman, luettavamman ja korrektin merkintäkielen merkitystä verkkosivujen luomisessa. Haluaisin tuoda nämä aiheet ja käsitykset näkyväksi etenkin alalla työskenteleville front-end ja back-end koodareille jotka työskentelevät merkintäkielien parissa, heidän lisäksi myös projektinhallintaa järjestäville henkilöille aikatauluttamista ja laskuttamista varten. Toisena kohderyhmään haluan mainita opiskelijat jotka haluavat työstää töitänsä nopeasti, helposti ja järjestelmällisesti. Mitä tämä tutkimus tulee merkitsemään ammattilaisille ja opiskelijoille on työhyödynnettävyyden ja työtehokkuuden parantaminen.

Koska haluan tutkimustyölläni tuoda ammattilaiset ja opiskelijat lähemmäs uudempien merkintäkielien opettelemista ja niiden hyötyjä - koitan jättää mahdollisimman paljon ’hienoja’ sanoja sekä ammattikieltä pois. Niille sanoille mitä joudun käyttämään tutkimuksessani loin terminologia osion, mistä löytyy suhteellisen yksinkertainen tiivistys mitä kyseiset sanat tarkoittavat.

Tämä tutkinta-aihe tuo erilaisia näkökulmia merkintäkielien käyttöön ja niiden edelleen kehittämiseen sekä työskentelyn järjestelmällisyyteen. Olen rajannut nämä työskentelytavat, koska itse käytän kyseisiä merkintäkieliä päivittäin työssäni muiden eri merkintäkielillä työskentelevien kollegoiden kanssa, eikä aiheesta löydy kunnollisia tutkimuksia mihin voisin verrata hankkimaani tutkimustyötä. Joten jätän opinnäytetyössäni ulkopuolelle muita merkintäkieliä kuten ERB ja SASS.

2. Verkkosivustoissa käytettäviä merkintäkieliä

2.1. HTML -merkintäkieli

HTML (Hypertext Markup Language) on avoimesti standardoitu kuvauskieli, jolla voidaan kuvata hyperlinkkejä sisältävää tekstiä eli hypertekstiä. HTML merkintäkielellä voidaan myös luoda tekstin rakenne, esimerkiksi mikä osa tekstistä on otsikkoa ja mikä leipätekstiä. HTML tunnetaan erityisesti kielenä, jolla verkkosivut luodaan.

HTML-koodi on rakenteista tekstiä joka muodostuu sisäkkäisistä ja perättäisistä elementeistä (katso Kuva 1). Elementin tyyppi nimetään merkinnässä kulmasulkeilla (<>). Verkkoselaimet eivät näytä tunnisteita sellaisinaan vaan käsittelevät niitä teknisinä ohjeina. Ohjeiden mukaan sivun varsinainen sisältö tulee jäsentää ja näyttää oikeassa järjestyksessä.

Tavallisesti elementissä on erikseen aloitustunniste ja vinoviivalla merkitty lopetustunniste sekä niiden väliin jäävä sisältö (jossa voi olla tekstiä ja myös muita HTML-elementtejä, kuten kuvia tai linkkejä). Aloitustunnisteeseen saattaa lisäksi sisältyä attribuutteja, jotka määrittävät elementin ominaisuuksia tarkemmin. Joissain HTML-elementeissä ei ole mitään sisältöä, kuten
 -elementissä joka luo rivinvaihdon.
(Wikimedia Foundationin, 2014)

```
<html>
    <body>
        <div class='class-elementti'>
            <div id='id-elementti'>
                <h2>Header 2</h2>
                <p>Lorem ipsum dolor sit amet, consectetuer adipiscing elit,
sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper
suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum
iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel
illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio
dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te
feugait nulla facilisi.</p>
            </div>
        </div>
    </body>
</html>
```

Kuva 1 - HTML-esimerkki

2.2. Haml -merkintäkieli

Haml -merkintäkielen on luonut Hampton Catlin. Haml oli alunperin tarkoitus olla vain henkilökohtainen työkalu Catlinilla, mutta ajan myötä se muuttui enemmän ja enemmän suositummaksi joten hän päätti julkaista ensimmäisen version. (Satish Talim, 2010) Kuitenkaan hän ei enää ymmärrä täysin Haml -merkintäkielen toiminnallisuuksia ja harvoin enää auttaa Haml -työryhmää konsultoimalla kysymyksiin. Nathan Weizenbaum on ollut monet vuodet ensisijainen kehittäjä ja arkkitehti ”modernin” Haml sisällytyksen Rubyyn. Muita Haml -kehittäjiä ovat Norman Clarke, Matt Wilding ja Akira Matsuda. (The Haml Team, 2013)

Haml -merkintäkielen toiminnan tarkoituksena on poistaa kaikki ylimääräiset merkinnät verkkosivujen ja -palvelujen lähteestä. Tämä tapahtuu niin että elementeistä poistetaan ylimääräiset pienetkin kirjaimet, kuten < ja > sekä loppu tagit. Sisäistäminen (indentointi) tulee tässä tärkeäksi osaksi Haml -merkintää, sillä ilman lopputagien merkintää on pakko olla jokin muu tapa ilmoittaa datan lukijaohjelmalle missä mikäkin elementti loppuu ja milloin alkaa uusi. Sisentäminen tuo ongelmia heille jotka eivät ole tottuneet puhtaan koodin kirjoittamiseen, sillä Haml pakoittaa kaikki koodaajat kirjoittamaan yhdellä tavalla eikä anna paljoa joustavuutta. Toisaalta tämä helpoittaa koodin lukemista, sillä kuka tahansa voi lukea toisen kirjoittamaa koodia ja ymmärtää sen erittäin nopeasti, kun taas HTML -merkintä voi olla erittäin sekaista ja sotkuista. (Niksiński Krzysztof, 2013)

Haml -merkintäkieli vaatii kuitenki aina jonkinlaisen ohjelman joka muuttaa (convert) lopulta Haml -merkintäkielen HTML -kieleksi. Tämä konvertointi sen takia että verkkoselaimet eivät osaa lukea Haml -kieltä, mutta monet ohjelmat muuttavat koodin nopeasti Haml:sta HTML:ään. Esimerkkinä mainitaan CodeKit joka konvertoi haml tiedoston sekunneissa html muotoon, ja tekee sen joka kerta kun tallentaa muutetun tiedoston. Tämä ei siis vaikuta työn edistymisen kannalta millään tavoin - muuta kuin kerran ohjelman asentamista vaativan toimenpiteen.

```
%html
    %body
        .class-elementti
            #id-elementti
                %h2 Header 2
                %p Lorem ipsum dolor sit amet, consectetuer adipiscing elit,
sed diam nonummy nibh euismod tincidunt ut laoreet dolore magna aliquam erat
volutpat. Ut wisi enim ad minim veniam, quis nostrud exerci tation ullamcorper
suscipit lobortis nisl ut aliquip ex ea commodo consequat. Duis autem vel eum
iriure dolor in hendrerit in vulputate velit esse molestie consequat, vel
illum dolore eu feugiat nulla facilisis at vero eros et accumsan et iusto odio
dignissim qui blandit praesent luptatum zzril delenit augue duis dolore te
feugait nulla facilisi.
```

Kuva 2 - HAML-esimerkki

Koska Haml -koodi kuitenkin jossain vaiheessa aina konvertoidaan HTML -kieleksi tarkoittaa se myös sitä että haml-tiedostoon voidaan syöttää ihan perus html-koodiakin, eikä se vaikuta millään tavoin toiminnallisuuksiin. Tästä hyvä esimerkki on esimerkiksi sosiaalisten medioiden plugin scriptien sisällyttämistä mikä täytyy aina generoida jossain päin Internettiä niin sen voi vain yksinkertaisesti kopioita ja liittää tiedostoon.

Haml:in merkintätapoja on monia, itse valitsin kyseisen tyylin merkitä koodit koska se muistuttaa eniten ruby merkintäkieltä - mikä on osa omaa työtäni. Kuvassa 2 on yksinkertainen koosto kuinka Haml -merkintää käytetään, huomaa elementtien sisennys.

2.3. CSS -merkintäkieli

CSS -merkintäkieli on tyylitiedosto millä määritetään dokumentin kirjoitettua ulkoasua, suurin osa käyttää kyseistä merkintäkieltä verkkosivujen ja -palvelujen ulkoasun muokkaamiseen, mitkä ovat yleisimmin kirjoitettu HTML tai XHTML -merkintäkielillä. CSS -merkintäkieltä kuitenkin voidaan käyttää muidenkin dokumenttien ulkoasujen muokkaamiseen, kuten XML, SVG ja XUL.

CSS on suunniteltu ensisijaisesti mahdollistamaan asiakirjan sisällön erottamiseen asiakirjasta. Elementtien ulkoasu, väri ja fontti ovat yleisiä muutoksen saavia kohteita. Muutos auttaa työhön tehtäviä muutoksia monilta eri kannoilta, kuten erilaisten elementtien nosto tärkeämpään arvoon esimerkiksi leipätekstistä. Yhden tyylitiedoston luominen voidaan sisällyttää moniin eri tiedostoihin, kuten verkkopalveluiden alasivuille, jolloin ulkoasu pysyy samana ja koko tyylittelyn tekeminen uudelleen on tarpeetonta. (Wikipedia Foundation, 2014)

```
1   html{
2       color:#00ffff;
3       background:#d90000;
4   }
5   body{
6       background-color:#00ffff;
7   }
8   .class-elementti{
9   }
10  .class-elementti div{
11      background-color:#ff8000;
12  }
13  .class-elementti div h2{
14      color:#d90000;
15  }
16  p{
17      color:#ff8000;
18  }
```

Kuva 3 - CSS-esimerkki

2.4. LESS -merkintäkieli

LESS -merkintäkielen on luonut Alexis Sellier, aka cloudhead. Nykyään LESS -merkintäkieltä ylläpitää LESS-tiimi. (The Core LESS team, 2013)

Kuten HAML, niin myös LESS:n tarkoitus on poistaa suurin osa ylimääräisestä koodista mitä välttämättäkin tulee kun kirjoitetaan CSS -tyylittelyjä. Kun CSS kielellä tehdään yksinkertainen linkkielementin väritys joka sattuisi olemaan div-elementissä jolla on class arvona 'linkki-holder' ja tälle div-elementille halutaan antaa margin-top: 10px; - niin tulos olisi tämä:

```
.linkki-holder { margin-top: 10px ; }
.linkki-holder a { color: red ; }
```

Jokainen elementti joudutaan kirjoittamaan alusta loppuu uusiksi, kun taas LESS yrittää parhaimmalla tarvalla poistaa kaikki ylimääräiset merkinnät erilaisella kirjoitus tavalla.

```
.linkki-holder {
   margin-top: 10px ;
   a { color: red ; }
}
```

Kun linkin omat määritteet sisällytetään pääelementin sisään - tulee kyseiset muutokset näkymään vain tämän pääelementin alla oleviin linkki elementteihin, kuten kuvassa 4 h2 elementtin määritteet tulevat näkymään vain class-elementti -elementin alla oleviin h2 elementteihin. Vaikka LESS -tyylitiedostoon tulee enemmän rivejä niin itse kirjainten määrä on huomattavasti pienempi.

```
1   @blue:#00ffff;
2   @red:#d90000;
3   @orange:#ff8000;
4
5
6   html{color:@blue;background:@red;}
7   body{background-color:@blue;}
8   .class-elementti{
9       div{background-color:@orange;
10          h2{color:@red;}
11      }
12  }
13  p{color:@orange;}
```

Kuva 4 - LESS-esimerkki

3. Verkkosivuston luontiprosessi

3.1. Suunnittelun vaikutus merkintäkielen valintaan

Kun verkkosivuja, -palveluja tai -kampanjoita lähdetään suunnittelemaan tulisi aina ottaa kantaa myös siihen millaisia toimintoja tulisi kyseisellä sivulla olla. Toiminnat vaikuttavat vahvasti siihen tuleeko merkintäkieli olemaan esimerkiksi PHP vai HTML5. PHP:ta esimerkiksi on vaikea kirjoittaa HAML-merkintäkielellä.

Mikäli värejä käytetään paljon eri elementeissä ja ne ovat samoja aina, kannattaa silloin miettiä käytetäänkö CSS vai LESS -merkintäkieltä. Sillä LESS:in kautta voidaan helposti vaihtaa yksi väri kaikkialta, kun taas CSS-merkinnässä siihen voi vierähtää aikaa huomattavasti enemmän.

Aina kuitenkin miettien kuka henkilöistä osaa tehdä milläkin merkintäkielellä mitäkin ja kenellä on aikaa tehdä työ. Työntekijöille on siis määriteltävä omat alueet mistä vastaavat. (Goto Kelly & Cotler Emily, 2002)

3.2. Back-end:in vaikutus merkintäkielen valintaan

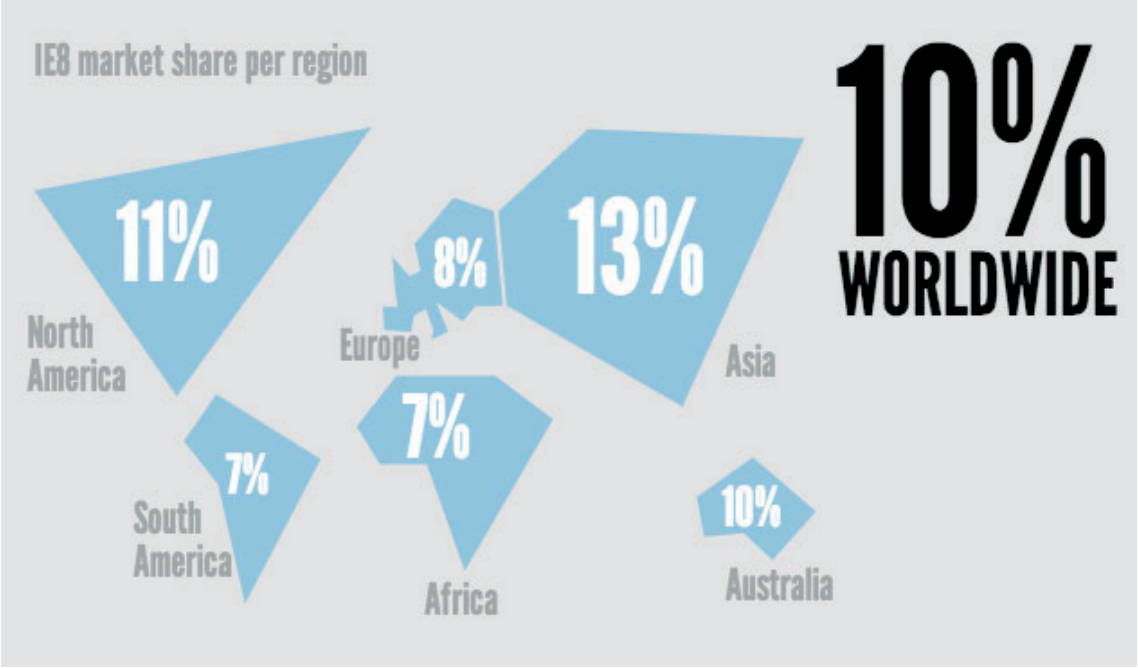
Internetin sisällön luonnissa yleensä tarvitaan jonkinlainen CMS-järjestelmä (sisällönhallintajärjestelmä), yksi ehdottomasti suosituimpia maailmalla on Wordpress. Wordpress on koodattu PHP:n päälle joka vaikuttaa vahvasti fron-end merkintäkieleen, työntekijä joutuu käyttämään tavalla tai toisella PHP koodia muun merkinnän kanssa. HTML voi sisältää PHP -koodia kunhan vain itse tiedoston loppupääte muistetaan vaihtaa .php (eikä .html) - silloin myös HAML pystyy kääntämään HTML & PHP -koodit yhteen php tiedostoon.

Kuten Wordpress - on moni muukin CMS-järjestelmä tehty PHP:n päälle joka on erittäin tunnettu ja käytetty merkintäkieli. On kuitenkin muitakin mahdollisuuksia, kuten ExpressionEngine joka on taas koodattu ruby -kieltä käyttäen missä ei ole php:ta. Tämä vaikuttaa jo kovemmalla kädellä front-end työhön. HTML koodaaminen toimii normaalisti mutta tässä kuitenkin täytyy käyttää rubyn omia merkintätakpoja - mistä itse olen oppinut HAML:n käytön. Kuten jo aiemmin sanoin - HAML:ia voi luoda eri tavoin, yksi tapa on merkitä se rubyn kaltaisesti, katso kuva 2 ja 10.

3.3. Front-end:in vaikutus merkintäkielen valintaan

Kuten suunnittelussakin front-end työssä tulee ottaa huomioon toiminnallisuus, minkälaisia erilaisia toimintoja milloinkin tulee. Mutta täytyy myös tietää millä eri selaimilla työn alla oleva palvelu tulisi toimia, Internet Explorer tuottaa aina pientä pään vaivaa sillä edelleen Euroopassa noin 8% käyttää Internet Explorerista versiota 8 (Josef Richter, 2013), joka on verkkoteknologioiden alalla tunnettu myös helvetin loukkuna. IE8 ei tue millään tavoin HTML5 -koodia joten tähän täytyy tehdä aina omat säädöt ellei sitten kyseinen palvelu tule nimenomaan toimia IE8:lla jolloin kannattaa miettiä onko järkevää tehdä suoraan xHTML -merkintäkielellä sivusto. Kuvassa 5 näkyy maailman laajuisesti IE8:n käyttö suhteessa kaikkiin verkkoselaimiin.

Front-end työssä tulee myös ottaa huomioon toisten työntekijöiden osallisuus projektiin nyt tai myöhemmin. Mitä nopeammin ja helpommin toiset ymmärtävät luotua merkintöjä tulee työskentely olemaan huomattavasti nopeampaa kuin lukea sekavaa koodia mistä kukaan ei ymmärrä mitään.



Kuva 5 - The Internet Explorer 8 Countdown

4. Työryhmässä verkkosivustoihin vaikuttavia prosesseja

4.1. Tekotapojen valinta

Merkintäkielen valintaan vaikuttaa siis ainakin kolme asiaa: suunnittelu, back-end ja front-end ongelmat, kuten kohderyhmä. Kun kohderyhmä ja tiedot millä kaikilla laitteilla ja verkkoselaimilla palvelun tulisi toimia - tulee seuraava vaihe joka on suunnittelu. Yleensä suunnittelijan kanssa kannattaa käydä läpi minkälaisia toimintoja tulee olla sivustolla ja miten kaiken tulisi toimia jonka jälkeen päästään sisällönsyöttöjärjestelmän kanssa työskentelyyn.

4.2. CMS (content management system)

Kun kaikki nämä on käyty läpi tulisi olla jo mielessä minkälainen merkintäkieli sopii kyseiselle työlle. Mikäli palvelu tulee olemaan wordpress pohjainen on silloin syytä miettiä onko kyseessä helposti tehtävä vai paljon muokkausta vaativa työ, mikäli muokkauksia tulee olemaan paljon niin silloin kannattaa valita nopeasti toimiva ja helposti luettava koodi, yksistään PHP:ta sisältävä merkintäkieli ei siis sovi tällaiseen. HTML ja Haml on helposti luettavaa ja suurin osa työskentelijöistä kokee näiden kielien käytön helpoksi.

4.3. Grid (sivuasetteluruudukko)

Suunnittelijat käyttävät sivuasetteluruudukkoja eli gridejä saadakseen selkeämmän ulkoasun suunnittelijalle ja koodarille. Tämä tulisi aina käydä läpi suunnittelijan kanssa mitä ruudukkoa käytetään, sillä niitäkin on monia erilaisia. Itse käytän suunnittelijani kanssa twitter bootstrap nimistä ruudukkoa ja esi asetettuja elementtejä. Nämä vaikuttavat nopeuttavasti niin että ruudukossa on 12 osaa aina, class elementtejä on yhdestä kahteentoista (span1 - span12) mitkä määrittelevät kyseisten elementtien parent-elementin osasta tietyn prosenttimäärän. Parent-elementti on aina row tai row-fluid, millä aina seuraava ruudukko aloitetaan. (Core team, 2014)

4.4. Front-end merkintäkieli

Kuten jo aikaisemmin kerrottu front-end työssä tulee aina ottaa huomioon projektin tulevaisuus, mikäli projektiin tai työhön tulee jossain vaiheessa uusia sisältöjä tai jatkokehitystä tulisi koodin olla ymmärrettävää ja helposti muokattavaa. Toisten työntekijöiden siis tulee ymmärtää sinun kirjoittamaa merkintäkieltä ja merkitsemistä jotta työ edistyy sulavasti eikä takkua siihen että toisten täytyy viettää aikaa merkinnän keskellä etsiessään jotain tiettyä palaa mikä täytyisi muokata.

5. Merkintäkielen kääntäminen

5.1. Tutkimusongelma

Tutkimukseni pääkysymyksenä pidän uudempien merkintäkielien vaikutusta nopeuttavana vaikutuksena työprojektien luonnissa ja muokkaamisessa - vaikuttavatko esimerkiksi Haml & LESS töiden luontiprosessiin nopeuttavasti vai onko kyseessä vain yksi turha yritys muuttaa merkintäkielen standardia.

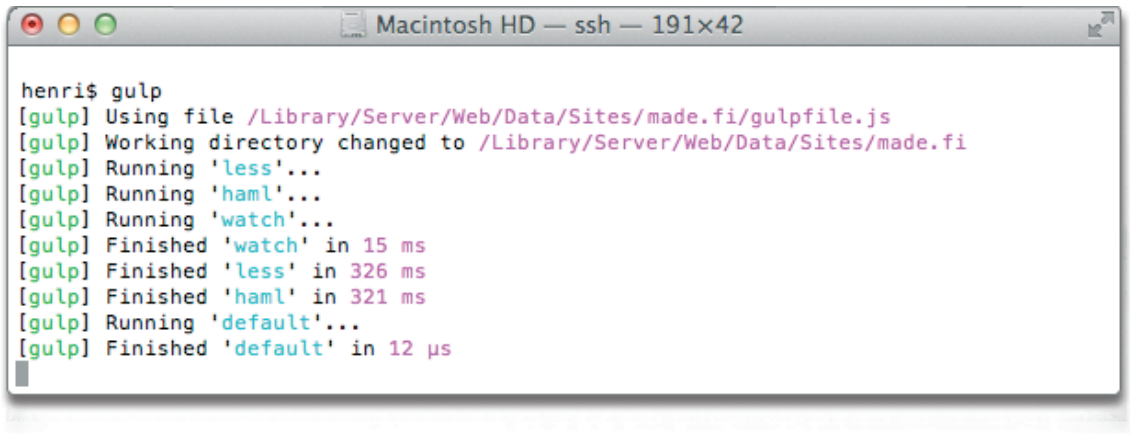
5.2. Tutkimushaastattelu

Osana kyselyä tein myös tutkimushaastattelun missä kartoitin hyviä ja huonoja puolia Haml ja LESS -merkintäkielistä. Yleisimpiä vastauksia oli muuttujien käytännöllisyys, eli kuten aikaisemmista esimerkeistäni on tullut esille esimerkiksi värien käyttö on yksinkertaista yhdellä muuttujalla vaihtaa. Toinen yleinen mielipide oli että kehitys on nopeampaa uudemmilla merkintäkielillä sekä virheiden määrä on vähäisempää kuin vanhempien merkintäkielien kanssa. Muita hyviä puoli mitä tuli vastaan oli koodin rakenteen ylläpitäminen sekä versioiden ylläpidettävyys on helpompaa.

Huonoina puolina esiin tuli joitain virheitä mitkä johti siihen ettei projekteissa voitu käyttää html & haml konversiota ollenkaan. Työkalujen vähyys oli myös ongelmallista sillä Mac OSX käyttöjärjestelmälle löytyy paremmin työkaluja kuin esimerkiksi Windows käyttöjärjestelmälle (CodeKit on yksi ohjelma Mac OSX käytöjärjestelmälle, kuva 8). Tämä johtaa ainoaan vaihtoehtoon missä joutuisi luoda konvertointi mahdollisuus suoraan serverille - mikä vaatii enemmän koodaus osaamista. Konversiotyökalujen konfigurointi tuottaa ylimääräistä päänvaivaa osalle vastaajista.

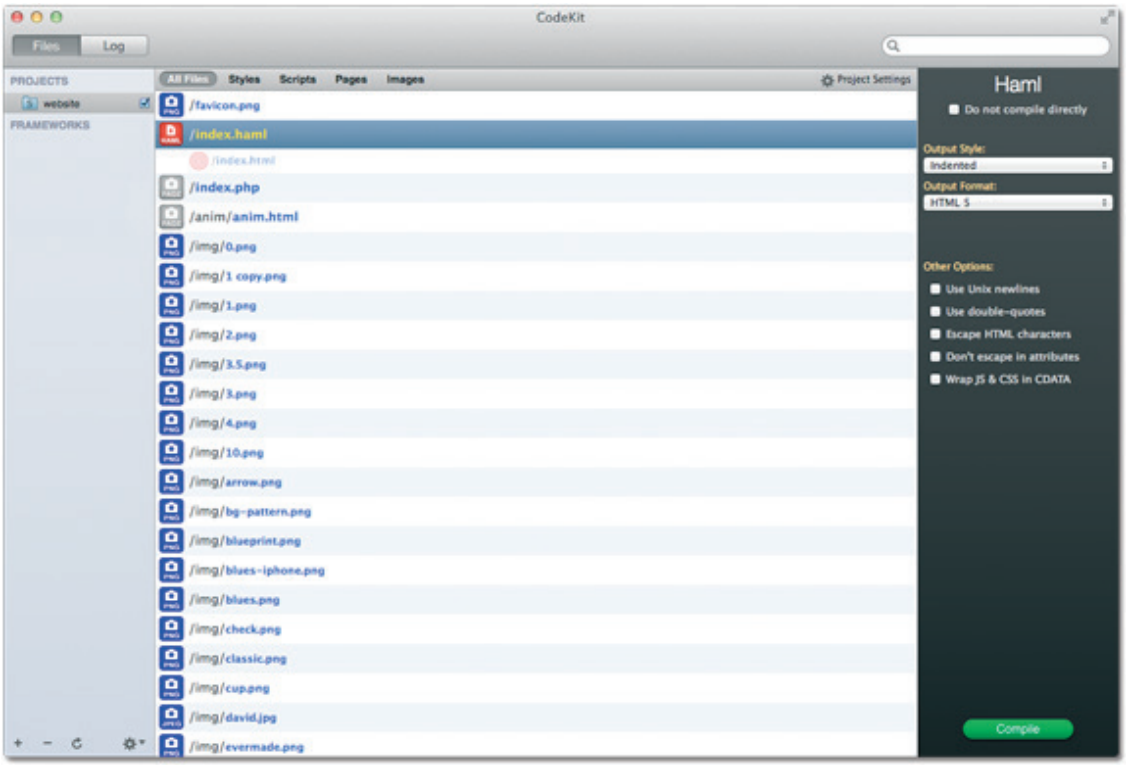
> “Aluksi vaikea hahmottaa. Kun syntaksia oppii lukemaan, tehostaa työtä. Sopivan työnkulun löytäminen vie aikaa. Huonosti sisennetty html-koodi on välillä erittäin hidasta tulkita. Haml poistaa tämän ongelman.”
>
> - Anonyymi haastatteluun vastaaja

Serverin puolella kännyssiä oleva konvertointi voidaan tehdä Gulp nimisellä lisäasetuksilla kehitys sivuston kansioon, katso kuvat 6 ja 7. Gulp on todella helppo kehittäjälle sillä esimerkiksi Terminal ohjelman kautta yhteys otetaan serveriin ja hakeudutaan sivuston kansioon, tämän jälkeen kirjoitetaan gulp ja kääntäminen alkaa automaattisesti (kuva 6). Se että serverille saatu konvertointi olisi näin helppoa vaatii hieman asetusten muokkaamista.



Kuva 6 - Terminal, Gulp, käyttöesimerkki

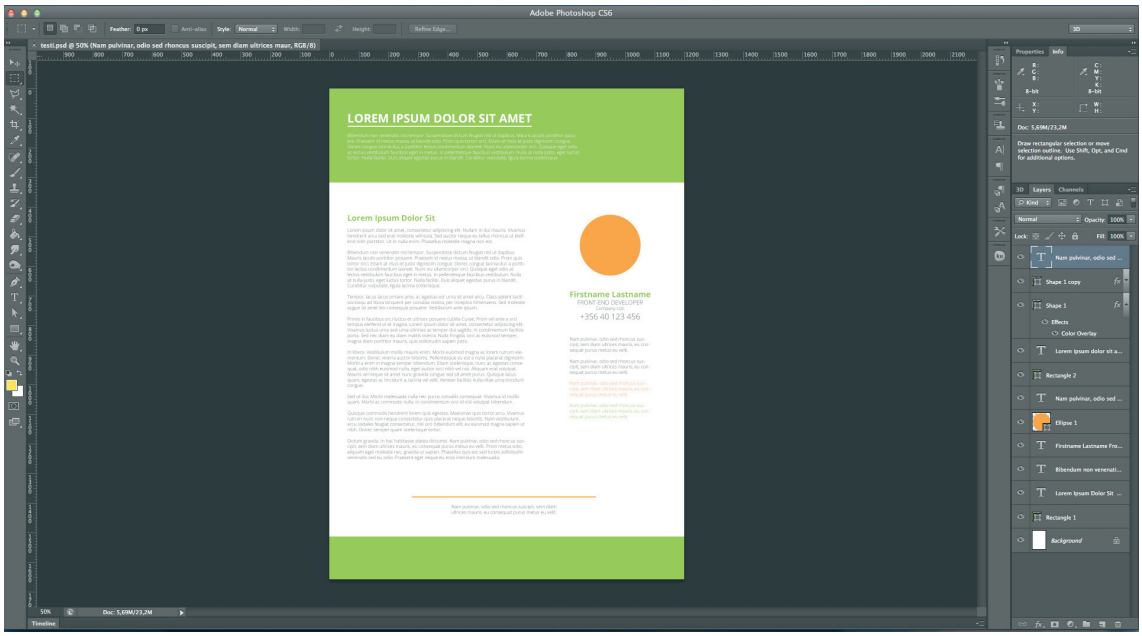
Kuva 7 - Gulp, JavaScript, asetukset.



Kuva 8 - CodeKit, lokaali konvertointi ohjelma.

6. Sivukoe

Osana tutkimustani halusin tuoda esille enemmän kysymystä nopeuttaako Haml & LESS sivuston luomisprosessia. Oikeastaan ainoa tapa löytää tähän konkreettista näyttöä oli tehdä oma pieni nopeahko ulkoasu sivulle ja antaa se muille tehtäväksi joko vain ja ainoastaan HTML & CSS tai Haml & LESS merkintäkielillä tehtäväksi. Huomaa ettei kyseessä siis ole tilastollisesti pätevä testi vaan ainoastaan suuntaa antava arvio HTML:n ja Haml:n eroista. Ensimmäisenä oli ammattilainen työnimikkeeltään front-end developer joka loi sivun ihan normaalisti HTML version. Tämän jälkeen toinen front-end developer loi saman sivun Haml & LESS -merkintäkielin. Huom. kumpikaan ei nähnyt toisten tekemää koodia. Jotta tulos ei olisi henkilökohtainen, eli ei olisi kyse toisen paremmuudesta tai muusta niin otin myös aikaa omasta työnopeudestani molemmissa versioissa, joita vertaan saatuihin tuloksiin. Lopullisista merkinnöistä en tarkkaile kuinka tarkasti on marginaalit ja fonttien koot tehty, kunhan vain on jokin määritelmä tehty ja ulkoasu on suunnilleen saman näköinen. Ulkoasun lähetin testihenkilöille PSD muodossa mikä on nähtävissä kuvassa 9.



Kuva 9 - Adobe Photoshop, Testi -tehtävän ulkoasu.

6.1. HTML & CSS -merkintäkielilla tehty

Ammattilaisen front-end koodarin (34 minuuttia 15 sekuntia) ja opiskelevan koodarin (39 min 58 sekuntia) työn teossa kesti keskiarviolta 37 minuuttia 7 sekuntia (37,11667) , minulla meni 39 minuuttia 21 sekuntia (39.35). Eli keskiarvoisesti 38 minuuttia 14 sekuntia (38.233335).

```
34.256 + 39.97734 = 74.23334
74.23334 % 2 = 37.11667

( 37.11667 + 39.35 ) / 2 = 38.233335
```

6.2. Haml & LESS -merkintäkielillä tehty

Ammattilaisen henkilön sama sivusto tuli tehtyä huimassa 17 minuutissa ja 44 sekunnissa, huomautettavaa on vielä sanoa että hän teki Haml version ensin jonka jälkeen hieman takkuilua tuottanut html -versio tuli jälkeen päin. Opiskelija teki samaisen työn html -version jälkeen ajassa 28 minuuttia ja 12 sekuntia, mikä on myös huomattavan suuri ero edellisen version teon nopeuteen verrattuna. Keskiarvoisesti heidän nopeus oli 22 minuuttia 52 sekuntia. Yhteensä ammattilainen ja opiskelija siis tekivät työn keskiarvoisesti 40,19 % nopeammin Haml & LESS -merkintäkieliä käyttäen.

```
( 17.7333 + 28.002 ) / 2 = 22.86765

( 100 * [ 22.86765 - 38.233335 ] ) / 38.233335 = -40,18924
```

Tein itse tämän tehtävän uudestaan viikon päästä etten muistaisi täysin miten olin tehnyt aikaisemman HTML-version. Minulla kesti 21 minuuttia ja 15 sekuntia Haml & LESS -merkintäkielillä tämän teossa, joka on häkellyttävät 17 minuuttia 6 sekuntia nopeammin kuin HTML & CSS -merkintäkielillä.

6.3. Testin merkintöjen tutkiminen

Tuloksia tutkiessa tulee ottaa huomioon se että kaikki tekijät ovat harjaantuneita Haml & LESS -merkintäkielien käyttäjiä. Ensimmäisenä kysymyksenä tulee esiin: Miksi Haml & LESS -merkintäkieli on niin nopea verrattuna HTML & CSS -merkintäkielen kirjoittamiseen? Vastaus on yksinkertainen ainakin Haml vastaan HTML -kielien kohdalla - elementtien kirjoittamiseen menee huomattavasti vähemmän aikaa, sillä elementtejä ei tarvitse sulkea vaan käännösohjelma tekee sen puolestasi, kunhan vain muistaa sisentää kaikki merkinnät oikein. Haml ja HTML -merkintäkielien erot ovat nähtävissä kuvissa 10 ja 11.

```
%html
%head
    %title Haml and LESS
    %link{:href => 'css/style.css', :rel => 'stylesheet'}
%body
    %header
        .container
            %h1 Lorem ipsum dolor sit amet
            %p Bibendum non venenatis nisl tempor. Suspendisse ..
    %section
        .container
            .left
                %h2 Lorem Ipsum Dolor Sit
                %p Lorem ipsum dolor sit amet, consectetur adipiscing elit..
                %p Bibendum non venenatis nisl tempor. Suspendisse dictum..
                %p Tempor, lacus lacus ornare ante, ac egestas est urna..
                %p Primis in faucibus orci luctus et ultrices posuere..
                %p In libero. Vestibulum mollis mauris enim. Morbi euismo..
                %p Sed ut dui. Morbi malesuada nulla nec purus convallis..
                %p Quisque commodo hendrerit lorem quis egestas..
                %p Dictum gravida. In hac habitasse platea dictumst..
            .right
                .circle
                %p.name Firstname Lastname
                %p.title Front-end Developer
                %p.company Company Ltd.
                %p.phone +356 40 123 456
                %p Nam pulvinar, odio sed rhoncus suscipit..
                %p Nam pulvinar, odio sed rhoncus suscipit..
                %p.orange Nam pulvinar, odio sed rhoncus..
                %p.green Nam pulvinar, odio sed rhoncus suscipit, sem ..
            .center
                %p Nam pulvinar, odio sed rhoncus suscipit, sem diam..
    %footer
```

Kuva 10 - Haml -merkintäkielen esimerkki tehtävästä

Luettavuus on huomattavasti helpompaa ja paremmin ymmärrettävissä Haml koodissa kuin HTML versiossa, kuten kuvien 10 ja 11 vertailusta näkee. Haml -merkintäkieli on selvästi nähtynä luettavampaa ja sitä myötä nopeammin muokattavaa sekä virheiden määrä on todennäköisesti vähäisempää kuin HTML -kielessä. Virheitä mitä Haml -merkintäkielessä voi tulla useasti eteen on indentointi väärin ja elementtien väärin kirjoittaminen, mutta HTML -merkintäkielessä on elementtien sulku järjestys, mikä voi tuottaa useaan otteeseen harmillista mielipahaa.

```
<html>
<head>
   <title>HAML</title>
   <link href=”style.css” rel=”stylesheet”>
</head>
<body>
   <header>
      <div class=”container”>
      <h1>Lorem ipsum dolor sit amet</h1>
      <p>Bibendum non venenatis nisl tempor. Suspendisse dictum feugia..</p>
   </div>
   </header>
   <section>
      <div class=”container”>
         <div class=”left”>
         <h2>Lorem Ipsum Dolor Sit</h2>
         <p>Lorem ipsum dolor sit amet, consectetur..</p>
         <p>Bibendum non venenatis nisl tempor. Suspendisse ..</p>
         <p>Tempor, lacus lacus ornare ante, ac egestas est ..</p>
         <p>Primis in faucibus orci luctus et ultrices ..</p>
         <p>In libero. Vestibulum mollis mauris enim. Morbi ..</p>
         <p>Sed ut dui. Morbi malesuada nulla nec purus ..</p>
         <p>Quisque commodo hendrerit lorem quis egestas..</p>
         <p>Dictum gravida. In hac habitasse platea dictu..</p>
      </div>
      <div>
      <div class=”right”>
         <div class=”circle”></div>
         <h2 class=”center”>Firstname Lastname</h2>
         <p class=”center title”>Front-end Developer</p>
         <p class=”center”>Company Ltd.</p>
         <p class=”center phonenumber”>+356 40 123 456</p>
         <p>Nam pulvinar, odio sed rhoncus suscipit, sem ..</p>
         <p>Nam pulvinar, odio sed rhoncus suscipit, ..</p>
         <p class=”orange”>Nam pulvinar, odio sed rhoncus suscipit..</p>
         <p class=”green”>Nam pulvinar, odio ..</p>
      </div>
      <div class=”center footer-text”>
         <p>Nam pulvinar, odio sed rhoncus..</p>
      </div>
   </section>
   <footer>
   </footer>
</body>
</html>
```

Kuva 11 - HTML -merkintäkielen esimerkki tehtävästä

LESS -merkintäkielessä huomataan jälleen muuttujien käyttö, mikä nopeuttaa etenkin jälkeen päin tehtäviä muutoksia (Libby Alex, 2013). Värit on merkitty yhden kerran ja sitä käytetään myöhemmin uudelleen, esimerkiksi vihreän värin muuttujana on @green jota voidaan käyttää esimerkiksi taustavärinä koodilla background:@green; Kuten jo aiemmin kerroin muuttujien toimintaa ja niiden hyödyllisyyttä - tulee se tässäkin esimerkissä vahvasti esille. Myöskin sisällytettyjä elementtejä on käytetty selkeästi helpottaakseen esimerkiksi headerissa olevaa p -tagin leveyttä ilman että se vaikuttaisi muihin p -elementteihin. Tämä voidaan siis CSS -merkinnässä tehdä myös, mutta ilman sisällyttämistä itse header tagin sisälle vaan kirjoittamalla se esimerkiksi header p , missä tulee kirjoitettua siis yksi header turhaan. CSS ja LESS -merkintäkielien eroja voi nähdä tarkastelemalla kuvia 12 ja 13. Kuvissa värjäsin elementtien värit, niistä näkee muuttujien käytön hyödyllisyyden.

```
html{font-family:open sans;}
body{padding:0;margin:0 auto;}
p{font-size:14px;font-weight:300;}
h1{color:white;}
h2{color:#97c00e;}
header{background:#97c00e;color:white;padding:60px 80px;}
header p{max-width:66%;}
header h1{border-bottom:3px solid white;font-size:36px;font-weight:bold;-
text-transform:uppercase;max-width:55%;}
section{min-height:1100px;}
.container{width:1200px;margin:auto;}
.left{width:600px;float:left;padding:110px 0;position:relative;}
.right{width:300px;float:right;padding:110px 90px;position:relative;}
.center{text-align:center;margin:0;}
.footer-text{position:relative;float:left;width:100%;}
.footer-text p{width:360px;margin:auto;border-top:3px solid #ff9600;paddin-
g:20px 0 0 0;margin-bottom:80px;}
.circle{width:220px;height:220px;background:#ff9600;border-radius:110px;mar-
gin:20px auto;}
.title{text-transform:uppercase;font-size:16px;}
.phonenumber{font-size:22px;margin-bottom:40px;}
.orange{color:#ff9600;}
.green{color:#97c00e;}
footer{min-height:200px;background:#97c00e;position:relative;bottom:0;-
float:left;width:100%;}
```

Kuva 12 - CSS -merkintäkielen esimerkki tehtävästä

```
@green: #97c00e;
@orange: #ff9600;
h1{.uppercase;border-bottom:3px solid white;}
p{font-weight:lighter;}
body{margin:0;padding:0;font-family:open sans;}
header{background:@green;color:white;padding:80px 0;
    h1{max-width:42%;}
    p{max-width:66%;}
   }
section{
    h2{color:@green;}
    .left{float:left;max-width:60%;margin:100px 0;}
    .right{float:right;max-width:25%;margin:100px 0;}
    .circle{background:@orange;width:210px;height:210px;border-radius:105px;-
margin:40px auto;}
    .name{color:@green;font-size:24px;font-weight:bold;.text-align-center;}
    .title{.text-align-center;.uppercase;}
    .company{.text-align-center;font-size:12px;}
    .phone{.text-align-center;font-size:22px;margin:0 0 60px 0;}
    .center{width:100%;.text-align-center;float:left;
       p{max-width:400px;margin:60px auto;border-top:3px solid @orange;pad-
ding-top:40px;}
      }
   }
footer{background:@green;min-height:250px;width:100%;float:left;}
.text-align-center{text-align:center;margin:0;}
.container{width:1200px;margin:auto;}
.uppercase{text-transform:uppercase;}
.orange{color:@orange;}
.green{color:@green;}
```

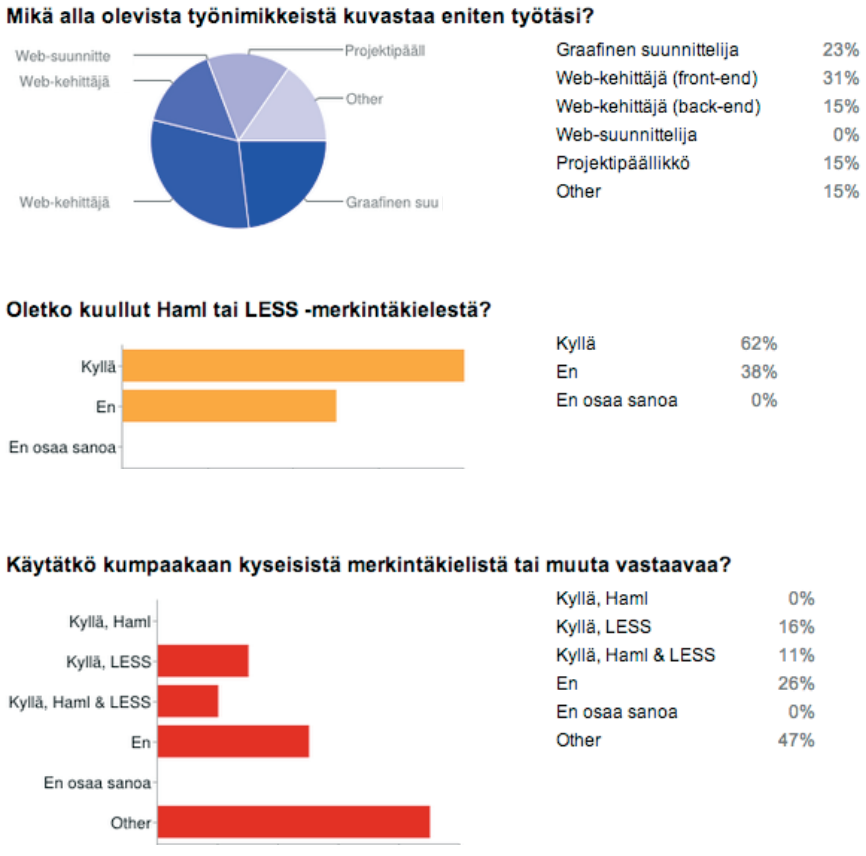
Kuva 13 - LESS -merkintäkielen esimerkki tehtävästä

7. Kysely ja tulokset

Tutkimukseeni kuului suurena osana kysely minkä tein Google Driven kautta. Jaoin linkin eteenpäin opiskelijoille ja verkkoteknologioiden parissa työskenteleville ammattilaisille, tulokset ovat jopa hieman ällistyttäviä. Katso kuvat 14 ja 15.

23% vastaajista oli ammatiltaan graafisia suunnittelijoita, joista hämmästyttävästi vain 34% vastasi seuraavaan kysymykseen "Oletko kuullut Haml tai LESS -merkintäkielestä?" positiivisesti. Tämä on harmillista tietoa front-end koodaajille sillä jos web suunnittelija ei tiedä uudemmista merkintäkielistä paljoa niin kommunikointi voi kärsiä jopa katastrofaalisesti, esimerkiksi värien käytön kanssa.

Vastaajista yhteensä 46% oli web-kehittäjiä, joista 31% front-end ja 15% back-end. Näistä yli 70% on kuullut Haml/LESS -merkintäkielistä, mutta vain yhteensä 25% kaikista vastaajista käyttää jompaa kumpaa tai molempia merkintäkieliä. 47%, eli melkein puolet, käyttävät jotain muuta uudempaa merkintäkieltä, kuten SASS (joka ei paljoa eroa LESS -merkintäkielestä).



Kuva 14 - GoogleDrive kysely tulkoset 1/2

Kokonaisuudessaan vain 26% kaikista vastaajista ei käytä minkäänlaista uudempaa merkintäkieltä, mikä on melkein yhtä paljon kuin graafisten suunnittelijoiden määrä. Tämä siis on ihan odotettavissa ettei suunnittelijat käytä merkintäkieliä. Yli jäävät 3% olivat henkilöitä joiden työn kuva oli projektipäällikkö.

Tärkein huomio kyselyssä tuli kun kysyin onko näiden (tai muiden uudempien merkintäkielien) käyttö auttanut vastaajaa jossain kohtaamassa ongelmassa. Tähän kysymykseen yli puolet (54%) vastasi kyllä, joista yli 50% oli usein saanut apua uudempien merkintäkielien kautta. Ainoastaan 15% kaikista vastaajista sanoi ettei ne ole auttanut projekteissa tulleissa ongelmissa ollenkaan.

Kuitenkin huomattavan suuri määrä vastaajista oli sitä mieltä että uudempien merkintäkielien käyttö parantaa työtehoa ja nopeuttaa projektien edistymistä. Yksikään vastaajista ei sanonut olevansa sitä mieltä ettei ne nopeuta tai paranna projektien edistymistä, 38% oli kuitenkin epävarmoja ja vastasivat "En osaa sanoa".

Kuva 15 - GoogleDrive kysely tulkoset 2/2

8. Yhteenveto

Tuloksena kaikista tutkimuksistani on selvää että Hamlin, LESSin ja muiden perus HTML ja CSS merkintäkieltä uudempien kielien käyttö tuottaa hyvää tulosta. Nopeuttava merkitys ei tässä tutkimuksessa tullut kunnolla esille, sillä testin tarkoituksena oli näyttää suuntaa antavaa näyttöä Haml ja LESS kielien vaikutuksesta. Kuitenkin ammattilaisten ja alaa opiskelevien henkilöiden haastattelu vastaukset toi esille niin hyviä kuin huonojakin puolia.

Huonoina otettakoon ehkäpä tärkein esiin, eli ongelmallisen vaikeahko joillekkin saada konvertointi toimimaan ilman virheitä. Huono yhteensopivuus eri käyttöjärjestelmien välillä toi myös huomioitavan ongelman esille. Nämä ongelmat mielestäni voidaan voittaa luomalla konvertointi järjestelmät automaattisiksi tuotanto ympäristöön, eli servereille missä luodaan sivustot ennen julkaisemista.

Hyvinä puolia oli muun muassa nopeus, yksinkertaisuus sekä merkinnän indentointi eli koodin kosmeettisuus ja sitä myötä luettavuus. Testi tehtävästä tulos oli huikea, sillä Haml & LESS -merkintäkielillä voidaan luoda yksinkertainen sivusto noin 40% nopeammin kuin normaalein HTML & CSS -merkintäkielin.

Ongelmallisena pidän kuitenkin huonohko ymmärrys muilla työaloilla kuin web kehittäjillä, esimerkiksi graafiset suunnittelijat jotka luova ilmeitä ja ulkoasuja verkkosivuille ja -palveluille tulisi mielestäni ymmärtää edes hieman merkintäkielistä mitä käytetään. Olen huomannut työelämässä suunnittelijoiden ymmärryksen auttavan huomattavasti kehittäjän ja suunnittelijan välistä kommunikointia.

Lähteet

The Core LESS team, About the history of LESS, 2013
http://lesscss.org/ (luettu 8.11.2013)

Core team, Bootrsap, 2014
http://getbootstrap.com/ (luettu 21.3.2014)

Goto Kelly & Cotler Emily, Verkkopalveluprojekti, 2002
Suomenkielinen versio, Riitta Satala-Köykkä, IT-press, 2003

The Haml Team, Info about Haml, 2013
http://haml.info/ (luettu 8.11.2013)

Libby Alex, Instant LESS CSS Preprocessor How-to, 2013

Niksiński Krzysztof, Instant HAML, 2013

Richter Josef, The Internet Explorer 8 Countdown, 2013
http://theie8countdown.com/ (luettu 19.2.2014)

Talim Satish, Hampton Catlin on Haml, 2010
http://rubylearning.com/blog/2010/10/12/hampton-catlin-on-haml/ (luettu 7.11.2013)

Wikipedia Foundation, Cascadian Style Sheets, 2014
http://en.wikipedia.org/wiki/Cascading_Style_Sheets (luettu 6.10.2013)

Wikipedia Foundation, HTML, 2014
http://fi.wikipedia.org/wiki/HTML (luettu 6.10.2013)

Liitteet

* Required

Mikä alla olevista työnimikkeistä kuvastaa eniten työtäsi? *

Valitse yksi vaihtoehto.

- Graafinen suunnittelija
- Web-kehittäjä (front-end)
- Web-kehittäjä (back-end)
- Web-suunnittelija
- Projektipäällikkö
- Other:

Oletko kuullut HAML tai LESS -merkintäkielestä? *

- Kyllä
- En
- En osaa sanoa

Käytätkö kumpaakaan kyseisistä merkintäkielistä tai muuta vastaavaa? *

- Kyllä, HAML
- Kyllä, LESS
- Kyllä, HAML & LESS
- En
- En osaa sanoa
- Other:

Miksi käytät tai miksi et käytä?

Mitä mieltä olet HAML -merkintäkielestä ?

Mitä mieltä olet LESS -merkintäkielestä ?

Minkälaisia ongelmia olet kohdannut kielten käytössä?

Onko näiden kielten käyttö auttanut sinua/teitä jossain kohtaamassanne ongelmassa?

- [ ] Kyllä, usein
- [ ] Kyllä
- [ ] Ei
- [ ] En osaa sanoa

Koetko uudempien merkintäkielien nopeuttavan työtehoa? *

- [ ] Kyllä, ehdottomasti
- [ ] Kyllä
- [ ] Ei
- [ ] Ei missään nimessä
- [ ] En osaa sanoa