TAMPEREEN TEKNILLINEN YLIOPISTO
TAMPERE UNIVERSITY OF TECHNOLOGY

ANTTI KALLONEN

# RESPONSIIVISET WEB-KÄYTTÖLIITTYMÄKIRJASTOT

Kandidaatintyö

Tarkastaja: Yliopistotutkija Marko Helenius
Palautettu: 9.8.2017

# TIIVISTELMÄ

**ANTTI KALLONEN**: Responsiiviset web-käyttöliittymäkirjastot / Responsive web user interfaces
Tampereen teknillinen yliopisto
Kandidaatintyö, 21 sivua, 3 liitesivua
Elokuu 2017
Tietotekniikan koulutusohjelma
Pääaine: Ohjelmistotekniikka
Tarkastaja: Yliopistotutkija Marko Helenius

Avainsanat: responsiivisuus, käyttöliittymä, Web-ohjelmointi

Responsiivinen web-käyttöliittymä on käyttäjän päätelaitteen ominaisuuksiin mukautuva käyttöliittymä. Laitekannan muuttuminen pelkistä työpöytäpäätteistä monipuolisemmaksi älypuhelimien yleistyessä suosii tällaisen kirjaston käyttämistä. Työssä vertaillaan kolmea eri web-käyttöliittymäkirjastoa, jotka ovat: Bootstrap, Semantic-UI ja Milligram.

Web-selaimet noudattavat tiettyjä standardeja, joiden ansiosta ohjelmistokehittäjä voi varmistua siitä, että sivusto on mahdollisimman suuren yleisön käytettävissä. Näitä standardeja ovat esimerkisi sivuston rakennetta kuvaava HTML, ulkoasua kuvaava CSS, toiminnallisuutta kuvaava Javascript ja sovellustason protokolla HTTP.

Päätelaitteiden kehittyessä myös web-sivustot ovat kehittyneet. 90-luvulla päätelaitteet olivat hyvin samankaltaisia ja täten myös sivustojen kehittämisessä ei tarvinnut huomioida eroja. 2000-luvun loppupuolella älypuhelimet alkoivat yleistyä ja tämä vaikutti käyttöliittymien suunnitteluun.

Erillisten mobiili- ja työpöytäkäyttöliittymien sijasta voidaan sivuston käyttöliittymä toteuttaa responsiivisena. Responsiivinen käyttöliittymä ottaa huomioon päätelaitteen resoluution ja fyysisen koon. Näiden tietojen perusteella se asettelee sivuston eri osaset eri paikoille riippuen laitteesta.

Kirjastojen vertailua varten luotiin käyttöliittymäprototyyppi ja se toteutettiin jokaisella kirjastolla erikseen. Vertailu on jaettu kolmeen osa-alueeseen: responsiivisuus, suorituskyky ja kehitettävyys.

Vertailun parhaaksi kirjastoksi osottautui Bootstrap ja lähes yhtä hyväksi Semantic-UI. Milligram kuitenkin oli suorituskyvyltään paras, mutta muilla mittareilla heikoin.

# SISÄLLYS

# LYHENTEET JA MERKINNÄT

| | |
|---|---|
| CSS | Cascading Style Sheets, tyyliohje HTML-tiedostolle |
| DOM | Document Object Model, rakenteellisen dokumentin kuvausmalli |
| HTML | Hypertext Markup Language, hypertekstiä sisältävä kuvauskieli |
| HTTP | Hypertext Transfer Protocol, sovelluskerroksen siirtoprotokolla |
| WWW | World Wide Web, Internetissä toimiva hajautettu hypertekstijärjestelmä |
| W3C | World Wide Web Consortium, WWW:n standardeja ylläpitävä yritysten ja yhteisöjen yhteenliittymä |

# 1. JOHDANTO

World Wide Web on saavuttanut vajaan kolmen vuosikymmenen kehityksen aikana yli puolet puolet koko maailman ihmisistä [9]. Siitä on kehittynyt tehokas tiedonhankinnan väline sekä alusta erilaisille palveluille. Mikä ennen mahtui vain pöydälle sopii nyt taskuun. Älypuhelinmarkkinat ovat kehittyneet räjähdysmäisesti ja ne ovat korvaamassa osaltaan tavallisen tietokoneen käytön ainakin kuluttajamarkkinoilla [9]. Nykyään yhä useamman ensimmäinen tietokone onkin älypuhelin ja edulliset älypuhelimet ovat kasvattaneet internetin käyttöä erityisesti kehitysmaissa [8].

Webiä käyttävä laitekanta on muuttunut homogeenisestä pöydillä istuvista tietokoneista heterogeenisempään suuntaan [8]. Päätelaitteet eivät ole enää kaikki samanlaisia, vaan ruutujen koot vaihtelevat älykellon ja -television välillä. Tämä aiheuttaa ongelmia käyttöliittymien suunnittelussa, koska ei ole yksittäistä ratkaisua, joka toimisi kaikille laitteille. Web-käyttöliittymät ovat tässä tilanteessa erityisasemassa, koska ne ovat alustariippumattomia ja saatavilla lähes joka laitteella. Yksi ongelmista on se, että käyttöliittymien rakenteen pitäisi mukautua erikokoisille päätelaitteille. Pelkkä käyttöliittymän venyttäminen tai pienentäminen ei toimi, vaan rakennetta pitäisi muovata älykkäämmin.

Responsiiviset web-käyttöliittymät ovat käyttäjän päätelaitteen resoluution ja koon mukaan mukautuvia käyttöliittymiä [11]. Tällöin jokaiselle eri päätelaitetyypille ei toteuteta omaa erillistä käyttöliittymää. Käyttäjien siirtyessä monimuotoisempaan laitekantaan responsiivinen suunnittelu muodostuu yhä tärkeämmäksi osaksi käyttöliittymien toteutusta.

Tämän työn tarkoituksena on vertailla kolmea yleisintä web-käyttöliittymäkirjastoa eri ominaisuuksien kannalta. Vertailtavat kirjastota ovat Bootstrap, Semantic-UI ja Milligram [2][17][15]. Näitä ominaisuuksia ovat responsiivisuus, suorituskyky ja kehitettävyys.

Tutkimusta varten luodaan testisivustot käyttäen jokaista eri kirjastoa. Suorituskykymittaukset tehdään Google Chromen kehittäjätyökaluilla, responsiivisuutta mitataan heuristisella arvioinnilla ja kehitettävyyttä arvioidaan omilla havainnoilla.

Työ alkaa johdannon jälkeen seuraavassa luvussa aiheelle olennaisten käsitteiden selittämisellä. Tämän jälkeen kolmannessa luvussa käydään läpi web-teknologian kehitystä ja syitä miksi responsiivisia käyttöliittymiä tarvitaan. Neljännessä luvussa perehdytään responsiivisen käyttöliittymän käsitteeseen. Tämän jälkeen esitellään tutkimuksessa vertailun kohteena olevat kolme kirjastoa: Bootstrap, Semantic-UI ja Milligram, sekä tutkimusmenetelmät. Tutkimuksen mittareita ovat responsiivisuus, suorituskyky ja kehitettävyys. Kuudennessa luvussa käydään läpi tutkimuksen tulokset, jotka osoittavat Bootstrapin olevan paras kolmesta kirjastosta tutkimuksen mittareiden perusteella. Semantic-UI on huonompi lähinnä vain suorituskyvyssä, kun taas Milligram osoittautuu olevan kirjastoista selvästi heikoin, sillä sen responsiiviset komponentit eivät suoriutuneet hyvin. Suorituskyvyssä Milligram oli kuitenkin kolmesta kirjastosta paras.

# 2. WEB-STANDARDIT

World Wide Web on järjestelmä, joka koostuu useista standardeista, joista merkittävimpiä ylläpitää yli 400 jäsenen W3C-yhtymä [20]. Asiakas- ja palvelinohjelmistojen tehtävänä on toteuttaa nämä standardit. Tärkeimpiä ohjelmistoja ovat sivustoa yleisölleen tarjoileva web-palvelinohjelmisto ja asiakkaan käyttämä web-selain. Web-kehittäjän tehtävänä on seurata näiden standardien julkaisua ja selain-ohjelmistojen standardiyhteensopivuutta. Kaikkia uusimpia ominaisuuksia ei välttämättä kannata toteuttaa, koska useimmat selaimet eivät vielä niitä tue.

## 2.1 HTML

Hypertext Markup Language, eli HTML on sivuston rakennetta kuvaava kieli. Sen tarkoituksena on tarjota keinot julkaista rakenteellisia dokumentteja, hyperlinkkejä, lomakkeita sekä multimediaa [21]. HTML koostuu sisäkkäisistä ja rinnakkaisista elementeistä, jotka kuvaavat dokumentin semantiikkaa (Kuva 2.1). Esimerkiksi navigaatiopalkkia kuvaavan nav-elementin sisällä voisi olla useampia hyperlinkkejä kuvaavia a-elementtejä. Tällä periaatteella voidaan kuvata sivuston rakenne ja sisältö. HTML-dokumenttia voidaankin rakenteensa vuoksi hahmottaa puuna, jossa dokumentin eri osat ovat puun solmuja. Tämän rakenteen käsittelyä varten on luotu Document Object Model (DOM) rajapinta, joka mahdollistaa HTML-dokumenttien lukemisen ja muokkaamisen.

```
1  <!DOCTYPE html>
2  <html>
3      <head>
4          <title>Example<title>
5          <link rel="stylesheet" href="styl
6      </head>
7      <body>
8          <h1>
9              <a href="/">Header</a>
10         </h1>
11         <nav>
12             <a href="one/">One</a>
13             <a href="two/">Two</a>
14             <a href="three/">Three</a>
15         </nav>
```

***Kuva 2.1*** *HTML-koodia, josta ilmenee puurakenne.*

## 2.2 CSS

Cascading Style Sheets, eli CSS kuvaa sivuston ulkoasun. CSS:n avulla voidaan määritellä eri HTML-elementtien värit, fontit ja muut ulkoasuun liittyvät ominaisuudet [21]. Sen pääasiallinen tarkoitus on eristää sivuston rakenne ja ulkoasu. CSS:n ansiosta sivuston sisällä tyyli pysyy yhtenäisenä, kun se määritellään keskitetysti yhdessä tiedostossa sen sijaan että se määriteltäisiin aina HTML-koodin yhteydessä. Tämän ansiosta säästytään ylimääräiseltä tietoliikenteeltä, kun sivuston CSS-tiedosto tarvitsee ladata vain kerran ja se voidaan tämän jälkeen noutaa välimuistista. CSS yhdistetään HTML:ään joko lisäämällä se suoraan tiedostoon tai linkittämällä se erillisestä tiedostosta.

## 2.3 JavaScript

Ecma Internationalin standardisoimaa komentosarjakieltä ECMAScriptiä [4] noudattava kieli JavaScript muodostaa HTML:n ja CSS:n kanssa tärkeimmät teknologiat Web-sivustojen tuottamiseen. Sen tarkoituksena on suorittaa asiakasohjelmalla ajettavaa toiminnallisuutta. Tällaisia toiminnallisuuksia ovat esimerkiksi sivuston osittainen päivittäminen lataamatta koko sivua uudestaan, palvelimelle lähetettävän syötteen validointi, sekä interaktiivinen sisältö. JavaScript päivittää HTML-sivua DOM-rajapinnan avulla.

JavaScriptiä käytetään varsinaisen verkkosivuston ohjelmakoodin lisäksi myös nykyään palvelinohjelmistoissa. Saman ohjelmointikielen käyttäminen koko ohjelmistopinossa helpottaa ohjelmiston kehittämistä.

## 2.4 HTTP

Hypertext Transfer Protocol, eli HTTP on sovellustason pyyntö-vastaus-protokolla, jonka tarkoituksena on siirtää tietoa palvelimen ja asiakasohjelman välillä [7]. Asiakasohjelma lähettää pyynnön, johon palvelinohjelmisto vastaa palauttamalla pyydetyn resurssin. Web-sivustot käyttävät HTTP-protokollaa HTML-tiedostojen ja multimedian välittämiseen.

# 3. WEB-TEKNOLOGIAN KEHITYS

Vajaan kolmen vuosikymmenen historiansa aikana WWW on kokenut monia muutoksia. Käyttötarkoitukset ovat lisääntyneet alkuperäisestä yksinkertaisesta hypertekstin jakamisesta multimediaan ja yritysten tuottamiin palveluihin. Web on kehittynyt yliopistojen tiedonvälitysalustasta globaaliksi ja kaupalliseksi alustaksi, jossa vaikuttavat suuret yritykset. Tässä luvussa käydään läpi web-käyttöliittymien ja päätelaitteiden tekninen kehityskaari.

## 3.1 Web-käyttöliittymät

Web-käyttöliittymien kehitystä tarkasteltaessa on hyvä huomioida, että uusien tekniikoiden käyttöönotto vie aikaa ja jotkin vanhoista tekniikoista ovat nykypäivänä vielä täysin käyttökelpoisia. Uudet tekniikat ovat mahdollistaneet suuremman ja monimuotoisemman tietomäärän esittämisen käyttäjälle. Alaluvun tarkoituksena on esitellä aikajärjestyksessä WWW:n historian aikana käytetyimmät web-käyttöliittymäteknologiat.

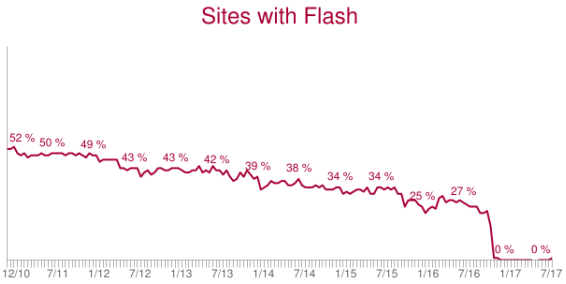
Vuonna 1990 Tim Berners-Lee kehitti kaikki tarvittavat työkalut Web-sivustojen katseluun, luomiseen ja tarjoiluun. Nämä työkalut olivat: HTTP, HTML, verkkoselain WorldWideWeb ja HTTP-palvelin CERN httpd [5]. WWW:n alkuperäisen kehittäjän Berners-Lee:n ensimmäinen sivusto oli täysin tekstipohjainen, koska ensimmäinen laajalti käytetty graafinen web-selain Mosaic oli vielä toteuttamatta. 90-luvun alun sivustot sisälsivät vain leipätekstiä sekä hyperlinkkejä muihin sivustoihin ja erillisiin palveluihin.

Yksinkertaisten tekstisivustojen jälkeen tulivat muotoilultaan monipuolisemmat taulukkopohjaiset sivustot. Alkujaan taulukkomuotoisen tiedon esittämiseen tarkoitetut HTML-elementit saivat uuden käyttötarkoituksen, kun niitä ruvettiin käyttämään sivustojen yleiseen muotoiluun. Taulukot mahdollistivat sivuston sisällön jakamisen sarakkeisiin ja riveihin. Esimerkiksi sivusto voisi koostua täysleveästä otsikkorivistä sekä navigointi- ja sisältösarakkeista.

Vuonna 1996 Yhdysvaltalainen yhtiö Macromedia julkaisi ensimmäisen Flash-version

nimeltään Macromedia Flash 1.0. Se nautti pitkään suosiota kehittäjien keskuudessa, mikä johtui osalta sen laajemmista ominaisuuksista HTML:ään verrattuna. Flash-sivustot olivat aikansa verkkoyhteyksiin nähden isoja, mikä taas johti heikkoon käytettävyyteen. Flash on Adoben omistama suljettu ohjelmisto, mikä estää muita tahoja toteuttamasta Flashia esittäviä ohjelmistoja.

Toisin kuin avoimien ohjelmistojen ja standardien tapauksessa, käyttäjät ovat riippuvaisia Flashin ylläpitäjästä Adobesta. Tämän lisäksi se on kärsinyt myös useista tietoturvaongelmista [12]. Flashin käyttö on vähentynyt vuosien 2010 ja 2016 välillä tasaista tahtia (Kuva 3.1) ja Adobe on päättänyt lopettaa sen jakelun ja päivittämisen vuoden 2020 lopussa [1].




***Kuva 3.1*** *Adobe Flashin käyttöaste Alexa top 1000 -sivutoissa joulukuusta 2010 [6].*

Myös vuonna 1996 julkaistu Cascading Style Sheets, eli CSS mahdollistaa sivustojen muotoilun irtauttamisen itse sisällöstä. HTML-tiedoston sisältäessä sivun rakenteen sekä sisällön ja erillinen CSS-tiedosto kuvaa taas miltä tämän sisällön pitäisi näyttää. CSS eli pitkään hiljaiseloa Flashin ollessa laajemmalti käytössä.

Vuonna 2014 julkaistu HTML5 standardi CSS3 kanssa ovat avointen standardien nykyaikaa. Hyvän selaintuen ansiosta ne ovat levinneet laajaan käyttöön. HTML5 toi mukanaan uusia semanttisia elementtejä parantamaan sivustojen rakennetta [23]. Toinen suuri muutos oli media-elementit, jotka mahdollistavat videon ja äänen toistamisen. Ennen HTML5:ttä mediaa toistettiin esimerkiksi Flashin avulla. Lisäksi nyt oli mahdollista piirtää grafiikkaa reaaliajassa uudelle canvas-elementille. HTML5:stä poiketen moduuleissa julkaistava CSS3 on tähän mennessä julkaissut mm. standardit animaatioihin, aritmetiikkaan ja sisällön luomiseen CSS:llä.

## 3.2 Päätelaitteet

Suurimman osan ajastaan WWW on saanut nauttia suppean laitekannan etuja, jonka ansiosta kehittäjän ei tarvinnut ottaa huomioon useita erilaisia päätelaitteita. Aina 2000-luvun loppupuolelle saakka päätelaitteet ovat olleet pääasiassa kannettavia- ja pöytätietokoneita. Käyttöliittymäsuunnittelun kannalta tärkeät ominaisuudet kuvatiheys ja resoluutio ovat kokeneet vain pientä kehitystä tänä aikana. Aikansa verkkosivuilla mainittiinkin usein, että sivusto on optimoitu tietylle resoluutiolle.

2000-luvun loppupuolella alkanut älypuhelinten yleistyminen muutti web-käyttöliittymien suunnittelun [16]. Päätelaitekanta ei ollut enää homogeeninen, vaan käyttöliittymien kannalta tärkeät ominaisuudet vaihtelivat laitteesta toiseen. Sivustojen on pystyttävä mukautumaan päätelaitteen mukaan, eivätkä staattiset sivut kykene siihen tarpeeksi joustavalla tavalla.

# 4. RESPONSIIVINEN KÄYTTÖLIITTYMÄ

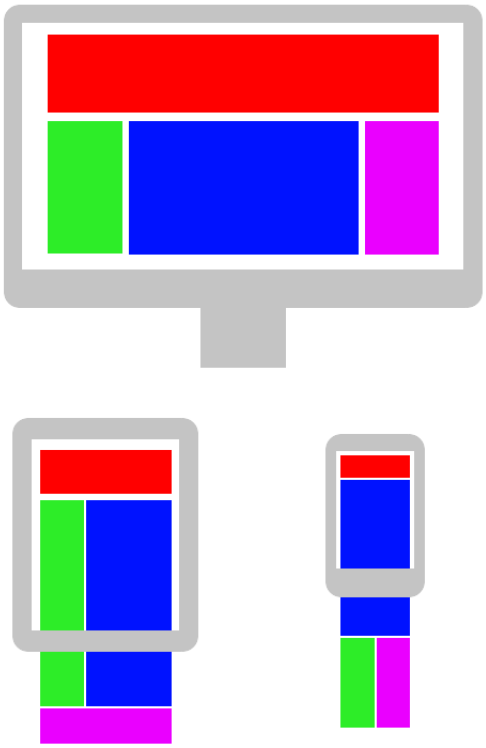
WWW-sivustoja voi pitkälti verrata painettuun mediaan. Molemmat sisältävät kuvia ja muotoiltua tekstiä. Painettavalla medialla on aina jokin vakituinen julkaisukoko, joka muuttuu vain harvoin. Taittajan tehtävänä on sovittaa sivulle eri artikkelit kuvineen. Tämän jälkeen julkaisu siirtyy painoon eikä sitä enää voida muuttaa. Varhaisten WWW-sivustojen tuottaminen oli hyvin vastaavanlainen prosessi, jossa sivuston osaset sovitettiin tietyn kokoiseen kehykseen. Erona kuitenkin se, että jo julkaistua materiaalia pystyi muokkaamaan.

Painettavan median ja digitaalisen median yksi erottava tekijä on, että sivun koon digitaalisessa mediassa määrittää käyttäjä julkaisijan sijasta. Pitkän aikaa tämä ei ollut ongelma, koska näyttöjen resoluutiot olivat lähellä toisiaan. Laitekirjon kasvaessa kuitenkin 1027x768 resoluutioiselle 15 tuumaiselle näytölle optimoitu sivusto ei vaan enää toiminut 1080x1920 resoluutioisella 5 tuumaisella älypuhelimen näytöllä. Web-maailman taittajan vastuuta piti luovuttaa käyttäjän pääteohjelmistolle.

Ethat Marcotte ehdotti tähän ongelmaan ratkaisua esittelemällään responsiivisen käyttöliittymän käsitteellä artikkelissaan "Responsive Web Design"[11]. Yksinkertaisuudessaan responsiivinen käyttöliittymä tarkoittaa päätelaitteen resoluution ja koon mukaan mukautuvaa käyttöliittymää. Marcotte esittelee artikkelissaan responsiivista käyttöliittymätoteutusta, joka hyödyntää CSS3:n uusia ominaisuuksia. Yksi olennaisin uudistus oli mediakyselyt-moduuli, jonka avulla sivusto pystyy tiedustelemaan päätelaitteen ominaisuuksia. Näitä ominaisuuksia ovat esimerkiksi laitteen tyyppi, resoluutio ja pistetiheys [22]. Mediakyselyiden avulla CSS pystyy asettelemaan ja piilottamaan elementtejä riippuen päätelaitteen ominaisuuksista. Myöhemmin samaa ideaa kantavista periaatteista on jalostettu yleiskäyttöisiä ohjelmistokirjastoja.

Responsiivisen käyttöliittymän voi toteuttaa monella eri tapaa. Yksi tapa on jakaa sivuston sisältö lohkoihin (Kuva 4.1). Yksi lohko sisältää esimerkiksi yhden uutisjutun tai kuvan. Nämä lohkot esitetään annetussa järjestyksessä ja päätelaitteen vaakaresoluutio määrittelee montako lohkoa esitetään vierekkäin. Otetaan esimerkkinä yksinkertainen uutissivuston uutisnäkymä. Sivulla on itse uutisjuttu, sekä lista

muista uutisjutuista. Suurella tietokoneen näytöllä nämä osiot voidaan esittää vierekkäin, mutta pienellä älypuhelimen näytöllä lohkot ovat allekkain. Responsiivinen käyttöliittymäkirjasto osaa tehdä asettelun automaattisesti.



***Kuva 4.1*** *Esimerkki kuinka lohkoihin jaettu responsiivinen käyttöliittymä mukautuu päätelaitteeseen.*

Responsiivisen käyttöliittymän suurin vahvuus on se, että sivustosta ei tarvitse ylläpitää ja päivittää useampaa eri versiota [10]. Täten säästetään kustannuksissa ja vältetään ristiriidat sisällössä. Huonoilta puolilta ei kuitenkaan vältytä, sillä hyvän responsiivisen sivuston suunnittelu ei ole yksinkertaista. Huonoilla ratkaisuilla päädytään toteutukseen, jossa mobiilikäyttäjän on helpompi käyttää työpöytä-versiota sivustosta mobiiliversioon verrattuna. Käyttöliittymäkirjastot lisäävät myös sivustojen latausaikoja, koska ne vaativat omat CSS ja JavaScript-tiedostonsa.

Testaaminen on olennainen osa ohjelmistotuotantoprosessia, mikä pätee myös verkkosivustojen kehittämiseen. Yksi testauksen alueista on ei-toiminnallinen testaus, jonka tarkoituksena nimensä mukaisesti ei ole testata toiminnallisuutta vaan ohjelmiston alla piileviä ominaisuuksia. Näitä ominaisuuksia voi olla esimerkiksi: suorituskyky, tietoturva ja skaalautuvuus. Joillekin näistä ominaisuuksista on helppo asettaa mittari. Suorituskykyä voidaan mitata vaikkapa sivulatauksina sekuntia kohden. Responsiivisuuden ollessa subjektiivisempi käsite, sille on vaikeampi asettaa mittaria. Sitä voidaan kuitenkin testata soveltamalla käytettävyystestauksen [13] periaatteita ja mittareita. Tavallisesta käytettävyystestauksesta poiketen järjestelmää testataan useammalla laitteella, eli testidata on moniulotteisempaa. Testausta voidaan myös suorittaa ilman erillisiä testihenkilöitä heuristisella arvioinnilla [14], eli asiantuntijaarvioinnilla.

# 5. KÄYTTÖLIITTYMÄKIRJASTOJEN VERTAILU

## 5.1 Kirjastojen esittely

Vertailuun valittiin kolme kirjastoa: Bootstrap, Semantic-UI ja Milligram. Kirjastojen valinnassa pyrittiin saamaan eri mittareita vasten parhaiten suoriutuvia kirjastoja ja samalla myös viitekehyksiltään erilaisia kirjastoja.

Ensimmäinen selvä valinta oli Bootstrap, koska se on suosituin 5.1 web-käyttöliittymäkirjasto. Muiden kirjastojen valinnat on perusteltu lähinnä niiden poikkeavuudesta Bootstrappiin mitattavien suureiden: responsiivisuuden, suorituskyvyn ja käytettävyyden suhteen. Näitä poikkeavuuksia on tietenkin hankala tietää ennen vertailua, joten valinnat perustuivat kirjastojen omiin kuvauksiin [15] [17].

Alla olevasta taulukosta 5.1 voidaan nähdä, että Bootstrap on kolmesta kirjastosta selvästi suosituin. Semantic-UI on noin viisitoista kertaa ja Milligram noin tuhat kertaa vähemmän käytetty kuin Bootstrap.

Tiedot on haettu paketinhallintapalvelu yarnin sivuilta.

***Taulukko 5.1*** *Kirjastojen lataukset 30 päivän aikana yarn-palvelussa toukokuussa 2017.*

| **Kirjasto** | **Lataukset (kpl)** |
|---|---:|
| Bootstrap | 1537800 |
| Semantic-UI | 93100 |
| Milligram | 1500 |

### 5.1.1 Bootstrap

Bootstrap kannustaa kehittäjiä ”mobile first”-kehitysperiaatteen noudattamiseen. Se tarkoittaa, että sivustoa aletaan suunnittelemaan mobiilialustalle sen ominaisuudet ja rajoitteet huomioon ottaen. Kun sivusto on toteutettu ensiksi mobiilialustalle

se on helppo laajentaa vähemmän rajoittuneille tavallisille työasemille. Bootstrapin responsiivinen toiminnallisuus perustuu ruudukkojärjestelmään, jossa on maksimissaan 12 saraketta. Sivusto koostuu sarakkeista, joille määritellään leveys yhdestä kahteentoista. Tämä leveys voidaan muuntaa pikseleiksi, jolloin yksi yksikkö on $\frac{1}{12} \times vaakaresoluutio$. Nämä sarakkeet ovat DOMissa rivi-elementin sisällä muodostaen yhden rivin. Mikäli rivin sarakkeiden yhteenlaskettu leveys ylittää 12 yksikköä ylimääräiset sarakkeet siiretään seuraavalle riville. Bootstrap erottelee eri päätelaitteet neljään eri kokoluokkaan niiden vaakaresoluution perusteella. Sarakkeelle voidaan määritellä eri leveys jokaiselle eri luokalle. Tarkoituksena on määritellä sarakkeille pienempi leveys suuremmille kokoluokille, jolloin yhdelle riville mahtuisi enemmän sisältöä suuremmalla resoluutiolla. Bootstrapissa onnistuu myös elementtien piilottaminen tietyiltä kokoluokilta. Tämän avulla toisarvoinen sisältö voidaan karsia pois ahtaalta näytöltä.

### 5.1.2 Semantic-UI

Semantic-UI pyrkii helpottamaan käyttöliittymän kehittämistä ihmiskeskeisellä ja ymmärrettävällä syntaksillaan. Jo nimestä voi päätellä, että semantiikka on tärkeä osa kirjaston periaatteita. Semantic-UI:ssa on myös Bootstrappia muistuttava ruudukkojärjestelmä, joka oletuksena jakaa rivin 16 sarakkeeseen. Kokoluokkia on samat neljä ja niiden resoluutiovälit ovat lähes samat. Esimerkkinä kirjaston semanttisuudesta neljän yksikön sarake määritellään: `<div class=''four wide column''></div>` , kun taas Bootstrapissa: `<div class=''col-md-4''></div>`. Syntaksi vastaa siis enemmän ihmisen kieltä.

Responsiivisen suunnittelun kannalta ruudukkojärjestelmässä on kaksi helpottavaa ominaisuutta. Ruudukko joka kuuluu luokkaan `doubling` kaksinkertaistaa sarakkeiden leveyden siirtyessä pienempään kokoluokkaan. Toinen ominaisuus `stackable grid` toimii yksinkertaisemmin pinoamalla kaikki sarakkeet päällekkäin mikäli ne eivät mahdu ruudulle rinnakkain. Näiden lisäksi sarakkeiden leveyden eri vaakaresoluutioilla voi määritellä käsin `(x) wide device` kuten Bootstrapilla.

### 5.1.3 Milligram

Milligram valittiin, koska sen luvataan olevan suunniteltu nopeampaan kehittämiseen ja parempaan suorituskykyyn. Se on kolmesta kirjastosta pienin viedessään pakattuna css-tiedostona alle 10 kilotavua tilaa. Toinen erikoisuus on se, että se ei käytä ollenkaan JavaScriptiä vaan riippuu ainoastaan CSS:stä. Myös Milligramista

löytyy yksinkertainen ruudukkojärjestelmä, joka hyödyntää CSS3:n `flexbox` ominaisuutta. Kehittäjä ei voi itse vaikuttaa niin paljon lopputulokseen, vaan joutuu luottamaan, että kirjasto toimii oikein.

## 5.2 Vertailualusta, mittarit ja mittaustyökalut

Kirjastojen vertailua varten luotiin käyttöliittymäprototyyppi ja se toteutettiin jokaisella kirjastolla erikseen. Käyttöliittymäprototyyppi on kuvitteellinen yrityksen tilausten keräilyyn keskittyvä työkalu. Se koostuu neljästä osiosta: navigointipalkista, kiirreellisistä, kuljetusta odottavista, sekä muista tilauksista. Käyttöliittymän on tarkoitus korostaa kaikista tärkeimpiä tilauksia, sekä piilottaa turhaa tietoa. Prototyypistä pyrittiin tekemään tarpeeksi laaja ja monimutkainen, jotta vertailussa ilmenisi eroja kirjastojen välillä. Liian yksinkertainen prototyyppi ei tuo tarpeeksi hyvin kirjastojen välisiä eroja esiin, kun kaikkia ominaisuuksia ei käytetä. Vertailun prototyyppi sisältää navigointipalkin, pudotusvalikoita, hakukentän, taulukoita, haitari-elementtejä (accordion) ja painikkeita. Sivusto-toteutuksista tehtiin mahdollisimman responsiiviset kirjastojen ominaisuuksien rajoissa.

Ensimmäisenä käyttöliittymä toteutettiin Bootstrapilla, koska se oli kirjastoista tutuin. Tämä toteutus toimi hyvänä pohjana muiden kirjastojen vastaaville toteutuksille. Kaikki käyttöliittymätoteutukset saatiinkin tehtyä hyvässä aikataulussa tutkimuksia varten.

Vertailua suorittaessa käytettiin Google Chromen kehittäjätyökaluja 5.4. Kehittäjätyökaluilla sivustoa voi helposti emuloida erikokoisilla resoluutioilla. Valittavina on muutama yleinen mobiililaite, sekä erillinen ”responsive”-valikko, jossa on seitsemän eri vaakaresoluutiota. Vaakaresoluutiot vaihtelevat 2560 ja 320 pikselin välillä. Responsiivisuuden vertailuun näistä valittiin taulukon 5.2 arvot, jotka edustavat älypuhelinta, tablettia ja kannettavaa tietokonetta. Muita selaimia ei käytetty vertailussa, koska niiden väliset erot oletettiin mitättömiksi vertailun kannalta.

***Taulukko 5.2*** *Vertailussa käytettävät vaakaresoluutiot*

| **Nimi** | **Vaakaresoluutio (px)** |
|---|---:|
| Mobile M | 375 |
| Tablet | 768 |
| Laptop L | 1440 |

Sivustojen responsiivisuutta vertaillessa ei voida suoraan käyttää Nielsenin heuristiikkoja [14], sillä ne keskittyvät pitkälti käyttöliittymän toiminnallisuuteen. Tämän

takia responsiivisuuden mittaamiseen luodaan omat heuristiikat. Vertailussa keskitytään kirjastojen kykyyn esittää sivusto selkeästi kullakin laitekoolla. Tämä tarkoittaa sitä, että kaikki haluttu sisältö on näkyvillä ja oikeassa järjestyksessä. Sisältö voi esimerkiksi ”vuotaa” sivulle, eli sivustoa pitää liikutella vaakasuunnassa. Tälläinen virheellinen toiminnallisuus ei ole tavoiteltavaa. Tärkeä tekijä on myös tilankäyttö, eli hukkaako kirjasto tilaa tarpeettomuuksiin.

Suorituskykymittauksissa mitattavat suureet ovat latausaika ja siirretyn datan määrä. Näitä suureita pystytään mittaamaan Chromen verkko- ja aikajanatyökaluilla. Mittaus suoritetaan avaamalla ensin testattava sivusto selaimella tämän jälkeen avataan itse mittaustyökalut, jotka yhdistyvät automaattisesti sivustoon. Sivusto virkistetään ja mittaustyökalut keräävät tiedot automaattisesti. Jokaisesta verkkoasetusten ja kirjaston yhdistelmästä otetaan useampi mittaus ja näistä lasketaan keskiarvo. Siirrettyä dataa mitattaessa on syytä ottaa selaimen välimuisti pois käytöstä, jotta koko sivu latautuu kokonaan joka kerta.

Sivustot sijoitetaan paikalliselle Node.js-palvelimelle, josta ne Express-kirjastoa hyödyntäen tarjotaan selaimelle. Koska palvelin sijaitsee samassa verkossa vertailua tehtävän tietokoneen kanssa, simuloidaan erilaisia verkkoasetuksia kehitystyökalujen avulla. Vertailussa käytetään taulukon 5.3 verkkoasetuksia, jotka löytyvät Google Chromen kehitystyökaluista.

***Taulukko 5.3*** *Vertailussa käytettävät verkkoasetukset*

| **Nimi** | **Latenssi (ms)** | **Latausnopeus (Mb/s)** | **Lähetysnopeus (Mb/s)** |
|---|---:|---:|---:|
| Wifi | 2 | 30 | 15 |
| Good 3G | 40 | 1,5 | 0,75 |

Käyttöliittymäkirjastoista ja testaukseen vaadittavista ohjelmistoista käytettiin niiden uusimpia saatavilla olevia versioita, jotka ovat listattuna taulukossa 5.4.

***Taulukko 5.4*** *Vertailussa käytettävät ohjelmistot/kirjastot ja niiden versiot*

| **Nimi** | **Versio** |
|---|---:|
| Google Chrome | 57.0.2987.133 |
| Node.js | 7.9.0 |
| Express | 4.15.2 |
| Bootstrap | 3.3.7 |
| Semantic-UI | 2.2.10 |
| Milligram | 1.3.0 |

Responsiivisuuden ja suorituskyvyn lisäksi vertailtiin kehitettävyyttä. Koska kahta

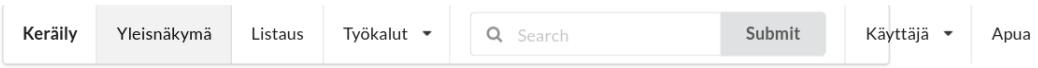
ensimmäistä asiaa käsitellään työssä kattavemmin, kehitettävyyttä vertaillaan vain omien havaintojen näkökulmasta.

# 6. TULOKSET JA NIIDEN TARKASTELU

## 6.1 Responsiivisuus

Responsiivisuutta vertaillessa kolmella eri vaakaresoluutiolla Bootstrap suoriutui kolmesta parhaiten. Vertailun aikana kirjastolla ei ollut ongelmia mukautua eri resoluutioille, vaan se suoriutui ongelmasta ilman huomattavia virheitä. Kuva Bootstrap-toteutuksesta Tablet-koossa löytyy liitteestä 3.

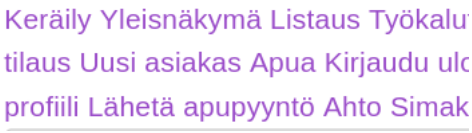
Semantic-UI:n suurin ongelma on navigaatiopalkki. Oikealle asetellut palkin elementit eivät pysy ruudun leveyden rajoissa vaan vuotavat sivulle (kuva 6.1). Tällöin sivustoa pitää liikuttaa vaakasuunnassa, että nämä elementit tulisivat näkyviin. Toinen pienempi ongelma navigaatiopalkissa on, että elementtejä ei pysty piilottamaan samalla tapaa kuin Bootstrapissa. Pienimmällä vaakaresoluutiolla palkki vie suuren osan näytön tilasta, jolloin heti sivun ladattua pitää selata alaspäin saadakseen tärkeimmän tiedon esille. Tällaisen toiminnallisuuden voisi toteuttaa itse, mutta tarkoituksena on vertailla kirjastojen tarjoamia toiminnallisuuksia. Semantic-UI:n käyttämissä kortti-elementeissä on myös joitakin ongelmia. Käyttöliittymän yhtenä tavoitteena on korostaa kaikkeista kiireisimpiä tilauksia. Bootstrapilla tämä korostus on selkeä, mutta Semantic-UI:lla korostus on vain pieni punainen viiva kortti-elementin alaosassa. Toinen ongelma näissä elementeissä on sisäkkäiset marginaalit, jotka rajoittivat tekstin määrää. Kun kortti-elementin sisälle sijoitetaan taulukko, sille tulee oma marginaali eikä se käytä täten koko kortin leveyttä. Ongelmana on myös erikorkuiset kortit, joissa sisältö keskitetään pystysuunnassa. Tämän ansiosta sivustoa lukiessa katse joutuu liikkumaan paljon pystysuunnassa. Vaikka ongelmia on useampia ne ovat kaikki pieniä, eikä Semantic-UI:n ja Bootstrapin välillä ole paljoa eroa. Kuva Semantic-UI -toteutksesta Tablet-koossa löytyy liitteestä 2.




***Kuva 6.1*** *Semantic-UI:n navigaatiopalkki Tablet-koossa*

Milligram suoriutui kolmesta kirjastosta huonoiten. Suurin ongelma oli tilausten tietoja käsittävien osioiden asettelu. Tablet-koossa taulukon tekstit menevät päällek-

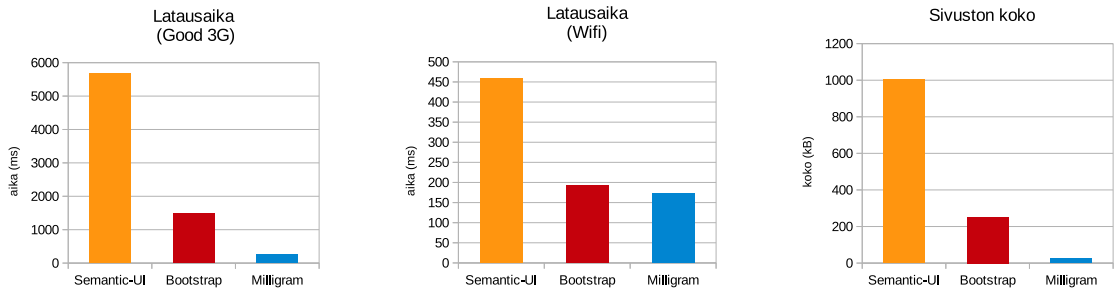
käin mikä vaikeuttaa lukemista huomattavasti (Liite 1). Sisältö ei myöskään mahdu vaakasuunnassa näytölle ja tämän takia sivustoa pitää liikutella myös vaakasuunnassa (kuva 6.2). Parhaiten Milligram toimi pienimmällä vaakaresoluutiolla, jolloin vain navigaatiopalkin linkit eivät mahtuneet näytölle. Näiden ongelmien vuoksi kirjastoa ei sellaisenaan voi käyttää hiemankaan monimutkaisemmissa sivustoissa.




***Kuva 6.2*** *Milligramin navigaatiopalkki Mobile M -koossa. Kaikki linkit eivät mahdu vaakasuunnassa näytölle, vaan käyttäjän pitää liikkua sivulla vaakasuunnassa.*

## 6.2 Suorituskyky

Suorituskykymittauksista ilmenee, että Semantic-UI on selvästi kolmesta kirjastosta raskain (Kuva 6.3c). Mittauksessa käytettiin taulukon 5.4 versioita ja taulukon 5.3 verkkoasetuksia. Bootstrap-sivusto on toiseksi raskain ollessa neljäsosan pienempi ja Milligram-sivusto on 40-kertaa pienempi. Verkon ollessa hitaampi nähdään, että latausaikojen suhteet alkavat muistuttaa enemmän sivustojen kokojen välisiä suhteita (Kuva 6.3a). Tämä johtuu siitä, että tiedonsiirtoon kuluva aika suhteessa muuhun toimintaan kasvaa. Matkapuhelinverkoissa toimivat mobiililaitteet kärsivät siis eniten raskaista kirjaistoista. Yli viiden sekunnin latausaika voi olla usealle jo liikaa. Toisaalta hyvillä tietoliikenneyhteyksillä kaikkien kirjastojen latausajat ovat alle puoli sekuntia, eli käytännössä mitättömät.




**(a)** Latausajat Good 3G -asetuksilla

**(b)** Latausajat Wifi-asetuksilla

**(c)** Sivustojen koot

***Kuva 6.3*** *Suorituskykymittausten kuvaajat*

Sivustot ovat suhteellisen yksinkertaisia eivätkä ne sisällä itsessään käytännössä ollenkaan JavaScript-koodia. Tästä johtuen tiedonsiirto on huomattavasti suurempi

tekijä sivuston latausajassa. Mikäli sivustoilla olisi käytössä enemmän JavaScript-koodia voisi päätelaitteen suoritin jo vaikuttaa tuloksiin. Tässäkin tapauksessa siis mobiilikäyttäjät kärsisivät pidemmistä latausajoista.

Erot tuloksissa korreloivat jossakin määrin kirjastojen ominaisuuksien määrässä. Milligram tarjoaa hyvin rajalliset ominaisuudet ja on täten huomattavasti kevyempi. Tästä johtuen Milligram-sivusto on toiminnallisuudeltaan heikompi Bootsrappiin ja Semantic-UI:hin verrattuna. Sivustosta puuttuu esimerkiksi navigaatiopalkki ja haitari-elementit. Jos nämä ominaisuudet olisi toteutettu itse Milligramin päälle, sivuston koko olisi kasvanut. Toisaalta myös Bootstrapin ja Semantic-UI:n kaikkia ominaisuuksia ei käytetty jolloin ne ladattiin turhaan. Tämä kärjistää eroa toiseen suuntaan. Ainakin Bootstrap tarjoaa mahdollisuuden käyttää vain osaa kirjaston komponenteista [3], mikä tekee mahdolliseksi pienemmän sivuston koon ja täten myös nopeammat latausajat.

Tutkimuksen tarkoituksena on kuitenkin verrata kirjastoja itsessään, eikä niiden optimointia tiettyyn toteutukseen. Ohjelmistokirjaston tehtävänä on tarjota toiminnallisuutta johonkin yleiseen ongelmaan, joka on tämän työn tapauksessa responsiivisen käyttöliittymän toteuttaminen. Koska kaikkien vertailtavien kirjastojen kaikkia ominaisuuksia ei käytetty, tulokset eivät ole täysin vertailukelpoisia ja vertailu suosii vähemmän ominaisuuksia sisältäviä kirjastoja.

## 6.3 Kehitettävyys

Sovelluskehittäjälle tärkeä työkalu dokumentaatio oli tärkeässä osassa vertailua varten tehdyissä toteutuksissa. Ilman hyvää dokumentaatiota kirjaston käyttäminen menee arvailun varaan, kun komponenttien toiminnallisuudesta ei ole tarkempaa tietoa. Dokumentaation yksi hyvä mittari on sen kattavuus. Bootstrappia lukuun ottamatta muut kirjastot olivat uusia tutkimusta suorittaneelle ja tästä syystä dokumentaatio oli tärkeässä roolissa kehitettävyyden kannalta. Kaikkien kolmen kirjaston dokumentaatiot ovat kattavat suhteessa ominaisuuksien määrään [2] [15] [17], eli kirjastojen toiminnallisuudet oli kuvattu siten, että niitä voi käyttää. Rakenteeltaan Semantic-UI:n dokumentaatio on selkein ja sieltä on helppo etsiä tietoa. Toisaalta Milligramin dokumentaatio on yksittäinen sivu ilman linkkejä eri osioihin. Koska dokumentaatio on melko lyhyt, tämä ei kuitenkaan haittaa suuresti käytettävyyttä.

Toinen mittari kehitettävyydelle on ongelmatilanteiden ratkaistavuus. Pelkkä dokumentaatio ei kata kaikkia käyttökohteita kirjastolle vaan se lähinnä kuvaa eri ominaisuudet. Kun on toteuttamassa jotain uutta ja varsinkin itselleen uudella kirjastolla,

on hyvä hakea vinkkejä muilta kehittäjiltä. On mahdollista, että joku muu on törmännyt samaan ongelmaan aiemmin. Stack Overflow on ohjelmoijien yhteisö, josta voi saada apua näihin asioihin. Palvelusta Bootstrapille merkittyjä kysymyksiä löytyi 83025 kappaletta 24.7.2017 [19], kun taas Semantic-UI:lle vain 1381 kappaletta [18]. Milligramiin liittyviä kysymyksiä ei löytynyt. Bootstrapillä kehitettäessä on siis paljon todennäköisempää löytää ratkaisu ongelmiinsa.

# 7. YHTEENVETO

Kirjastoja valittaessa vaikutti siltä, että Bootstrap ja Semantic-UI eivät eroasi suuresti toisistaan. Responsiivisuutta verratessa näiden kahden välillä ei ilmennyt merkittäviä eroja ja molemmat kirjastot ovatkin täysin kykeneviä toteuttamaan responsiivisen web-käyttöliittymän. Vaikkakin Semantic-UI oli noin neljä kertaa raskaampi sivuston koon näkökulmasta, eroa pystynee vähentämään karsimalla käyttämättömät ominaisuudet pois.

Mahdollisena jatkotutkimuksena vertailtavat sivustot voisivat olla raskaampia ja ominaisuuksiltaan vastaavanlaisia. Erityisesti JavaScriptiä käyttävät toiminnot kuten DOMin manipulointi voisivat tuoda ilmi eroja kirjastojen välillä. Puutteellisuudet kirjastoissa paikattaisiin omilla toteutuksilla, mikä taas vaikuttaisi kehitettävyyteen. Tällä tutkimuksella tulokset voisivat olla erilaiset ja vastaisivat enemmän todellista tilannetta.

Milligram on kolmesta kirjastosta selvästi erilaisin. Kirjaston sivuilla sanotaan: "*Milligram provides a minimal setup of styles for a fast and clean starting point.*" [15], mikä kuvaakin sitä hyvin. Se toimii hyvin prototyyppien, sekä pienten ja kevyiden sivustojen toteuttamiseen, mutta monimutkaisempaa kokonaisuutta sillä ei kannata tehdä.

Jos erot ovat kahden paremman kirjaston välillä näin pienet, onko tarvetta valita näistä kahdesta parempi? Jokainen hyvin toteutettu ja tarpeeksi kattava kirjasto pystyy tarjoamaan työkalut responsiivisen käyttöliittymän toteuttamiseksi. Näin ainakin tutkimuksen tulokset antavat uskoa.

# LÄHTEET

[1] Adobe, "Flash & the future of interactive content," https://blogs.adobe.com/conversations/2017/07/adobe-flash-update.html, 2017, haettu 7.8.2017.

[2] Bootstrap, "Bootstrap dokumentaatio," https://bootstrapdocs.com/v3.3.6/docs/, haettu: 5.5.2017.

[3] Bootstrap, "Customize bootstrap's components, less variables, and jquery plugins to get your very own version." https://getbootstrap.com/customize/, haettu: 16.4.2017.

[4] ECMA international, "Ecmascript® 2017 language specification (ecma-262, 8th edition, june 2017)," https://www.ecma-international.org/ecma-262/8.0/index.html, 2017, haettu: 31.7.2017.

[5] J. Gillies & R. Cailliau, *How the Web was Born: The Story of the World Wide Web*. Oxford University Press, 2000.

[6] httparchive.org, "Top 1000 sites with flash," http://httparchive.org/trends.php?s=Top1000&minlabel=Nov+15+2010&maxlabel=Jul+15+2017#perFlash, haettu 7.8.2017.

[7] Internet Engineering Task Force, "Internet engineering task force (ietf) hypertext transfer protocol version 2 (http/2)," https://tools.ietf.org/html/rfc7540, 2015, haettu: 20.2.2017.

[8] ITU, "Measuring the information society report 2016, chapter 5. measuring mobile uptake," http://www.itu.int/en/ITU-D/Statistics/Documents/publications/misr2016/MISR2016-w4.pdf, 2016, haettu: 6.8.2017.

[9] ——, "Ict facts and figures 2017," http://www.itu.int/en/ITU-D/Statistics/Documents/facts/ICTFactsFigures2017.pdf, 2017, haettu: 6.8.2017.

[10] B. Kim, "Responsive web design, discoverability, and mobile challenge," *Library Technology Reports; Chicago*, vol. 49, no. 6, pp. 39,2, Aug/Sep 2013. [Online]. Available: http://search.proquest.com.libproxy.tut.fi/docview/1441522432/abstract/8EB0F9EC41B042ACPQ/1

[11] E. Marcotte, "Responsive web design," https://alistapart.com/article/responsive-web-design, 2010, haettu: 13.3.2017.

[12] MITRE, "Cve details adobe flash player: Vulnerability statistics," http://www.cvedetails.com/product/6761/Adobe-Flash-Player.html?vendor_id=53, haettu: 20.3.2017.

[13] J. Nielsen, "Usability inspection methods," in *Conference Companion on Human Factors in Computing Systems*, ser. CHI '95. New York, NY, USA: ACM, 1995, pp. 377–378.

[14] J. Nielsen & R. Molich, "Heuristic evaluation of user interfaces," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, ser. CHI '90. New York, NY, USA: ACM, 1990, pp. 249–256.

[15] C. Patoilo, "Milligram," http://milligram.io/, haettu: 5.5.2017.

[16] E. Perakakis, G. Ghinea, & E. Thanou, "Are websites optimized for mobile devices and smart tvs?" June 2015, pp. 47–53.

[17] Semantic-UI, "Semantic-ui dokumentaatio," https://semantic-ui.com/introduction/getting-started.html, haettu: 5.5.2017.

[18] Stack Overflow, "Tagged questions semantic-ui," https://stackoverflow.com/questions/tagged/semantic-ui, haettu: 5.5.2017.

[19] ——, "Tagged questions twitter-bootstrap," https://stackoverflow.com/questions/tagged/twitter-bootstrap, haettu: 5.5.2017.

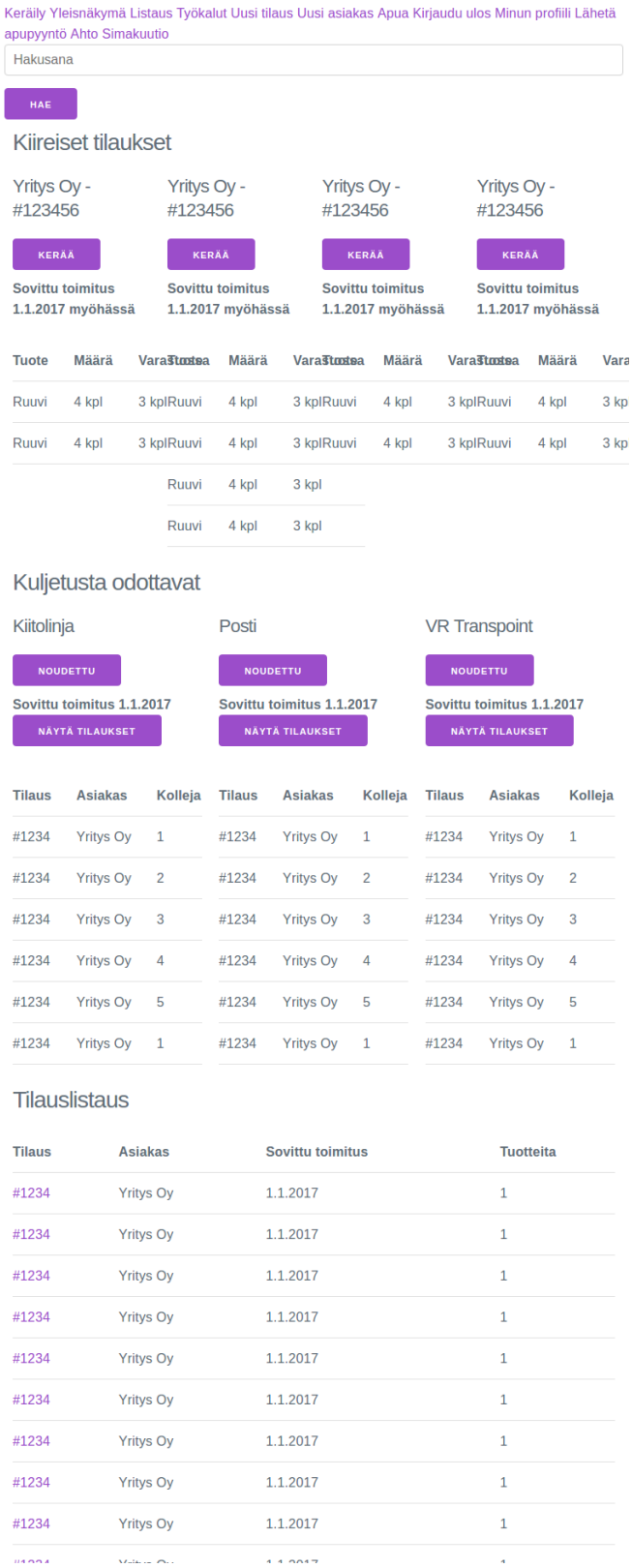
[20] W3C, "World wide web consortium (w3c) current members," https://www.w3.org/Consortium/Member/List, haettu: 20.2.2017.

[21] ——, "World wide web consortium (w3c) html and css," https://www.w3.org/standards/webdesign/htmlcss, haettu: 20.2.2017.

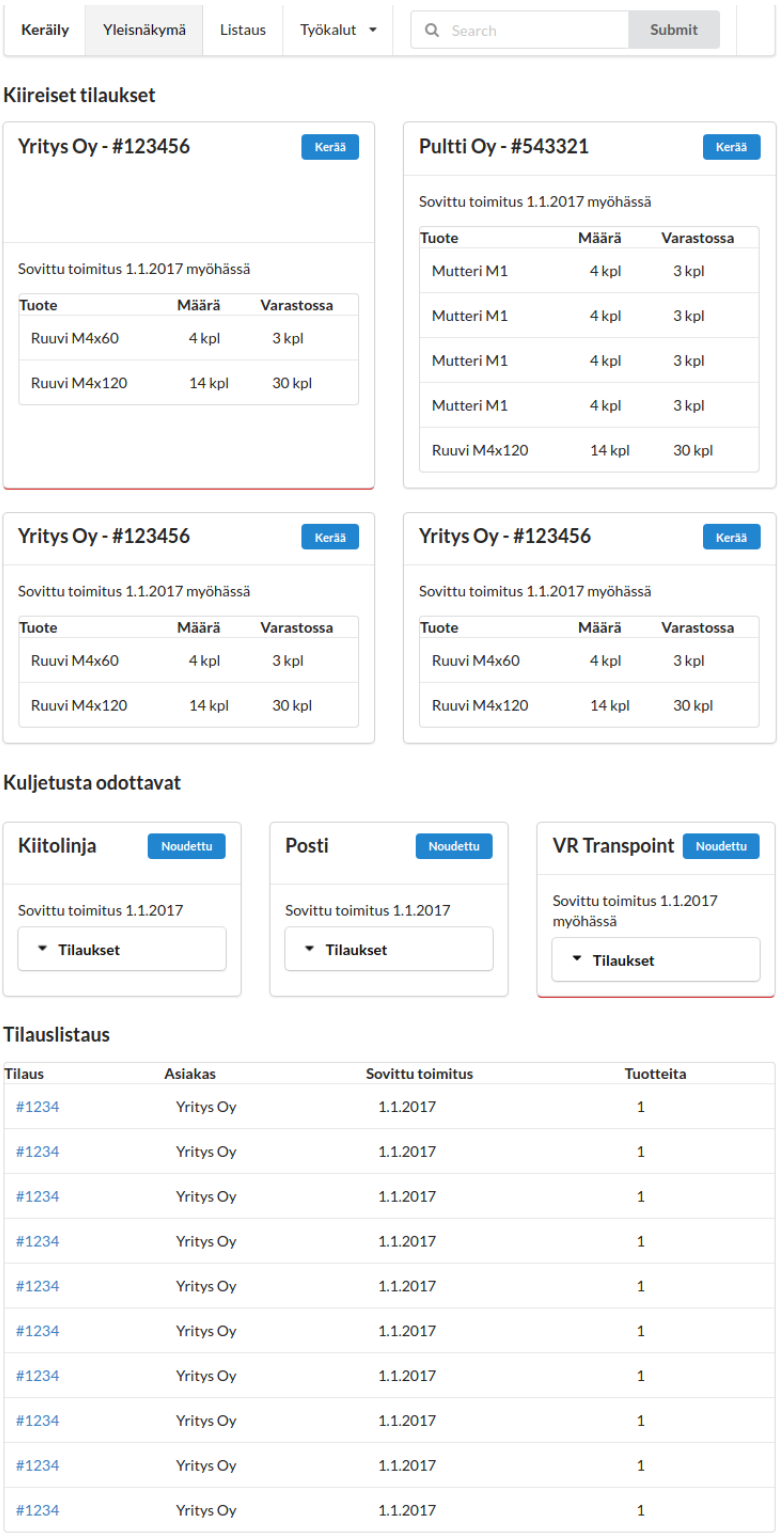
[22] ——, "Media queries," https://www.w3.org/TR/css3-mediaqueries/, 2012, haettu: 20.3.2017.

[23] ——, "Html5 a vocabulary and associated apis for html and xhtml," https://www.w3.org/TR/html5/, 2014, haettu: 20.3.2017.

# LIITE 1: MILLIGRAM TABLET-KOOSSA



# LIITE 2: SEMANTIC-UI TABLET-KOOSSA



# LIITE 3: BOOTSTRAP TABLET-KOOSSA