Opinnäytetyö (AMK)

Tietojenkäsittelyn koulutusohjelma

Yrittäjyys ja sähköinen liiketoiminta

2016

Jani Kalliomäki

RESPONSIIVISEN VERKKOSIVUN TOTEUTUS DRUPAL –SISÄLLÖNHALLINTA-JÄRJESTELMÄLLÄ

– Case: Nostokonepalvelu Oy

TURUN AMMATTIKORKEAKOULU
TURKU UNIVERSITY OF APPLIED SCIENCES

OPINNÄYTETYÖ (AMK) | TIIVISTELMÄ

TURUN AMMATTIKORKEAKOULU

Tietojenkäsittelyn koulutusohjelma | Yrittäjyys ja sähköinen liiketoiminta

Toukokuu 2016 | Sivumäärä 27

Ohjaaja: Päivi Killström

Jani Kalliomäki

RESPONSIIVISEN VERKKOSIVUN TOTEUTUS DRUPAL – SISÄLLÖNHALLINTAJÄRJESTELMÄLLÄ

Verkkosivuja selataan nykypäivänä yhä enenevässä määrin mobiililaitteilla. Tämän on faktan on myös huomannut hakukone Google, joka alkoi vuoden 2015 huhtikuussa suosimaan mobiiliystävällisiä sivustoja hakutuloksissaan. Siksi sivustoja täytyy muokata sopimaan paremmin pienemmille näytöille. Aikaisemmin tämä on tarkoittanut käytännössä sitä, että sivustolle luodaan erillinen mobiiliversio johon käyttäjä ohjataan mikäli hän saapuu sivustolle mobiililaitteella. Tämä on kuitenkin useimmiten kallis vaihtoehto, joka tuottaa toteuttajalle ylimääräistä työtä ja tekee sivustoista hitaita. Vaihtoehto mobiiliystävällisten sivustojen toteuttamiseen on luoda responsiivinen sivusto, joka skaalautuu käyttäjän laitteen näytön mukaan.

Tässä opinnäytetyössä esitellään yksi tapa, joilla responsiivisuuden voi toteuttaa. Työssä käsitellään responsiivisuuden toteuttamista CSS- tai JavaScript-tekniikoita käyttäen. Lisäksi työssä perehdytään siihen, mitä kannattaa ottaa huomioon responsiivista verkkosivua suunnitellessa. Sivuston luomisessa käytän työvälineenä Drupal –sisällönhallintajärjestelmää, johon rakennan responsiivisen teeman.

Opinnäytetyön tuloksena syntyy nosto- ja kuljetusalan yritykselle sivusto, joka toteutetaan responsiivisesti. Sivusto saatiin lähes valmiiksi, ja se julkistetaan kun sisältö saadaan syötettyä paikalleen. Asiakkaan toiveena oli muuttaa sivuston ulkoasu nykyaikaisemmaksi, mutta silti säilyttää hallinnan käyttöliittymä mahdollisimman muuttumattomana. Näissä tavoitteissa onnistuttiin pysymään, minkä takia opinnäytetyö onnistui tavoitteiden mukaisesti.

ASIASANAT:

Drupal, responsiivinen verkkosuunnittelu, CSS media query

BACHELOR´S THESIS | ABSTRACT

TURKU UNIVERSITY OF APPLIED SCIENCES

Degree Programme in Business Information Technology | Entrepreneurship and e-Business

May 2016 | Total number of pages 27

Instructor: Päivi Killström

Jani Kalliomäki

RESPONSIVE WEB DEVELOPMENT WITH DRUPAL CONTENT MANAGEMENT SYSTEM

Nowadays websites are used more often with mobile devices. This is a fact that also Google has noticed. In April 2015 Google started preferring mobile-friendly websites in their search results. Thus, the websites have to be modified to fit better for narrow screens. Earlier this was done by building a separate mobile site, in which the user was redirected, if s/he tried to access the site with a mobile device. However that is an expensive method, which causes a lot of work for developers and makes websites slow. Responsive web design is a very feasible alternative, where the website and its elements response to the width of the device and its screen.

This thesis presents a method to build a responsive website. Two alternative ways for responsive development are introduced: CSS media queries and JavaScript. Some important considerations when designing a responsive website are also discussed. Drupal content management system was used as the tool for building the website.

As an outcome, a new responsive website was designed for a logistics and lifting company. The website is almost ready, and will be published as soon as the text content is completed and placed. The client wished that the layout of the website would be modified in more modern look, but also the administration left as similar as possible as in the existing website. These wishes were fulfilled and as a whole the thesis met the set goals.

KEYWORDS:

Drupal, responsive web design, CSS media queries

SISÄLTÖ

KUVAT

KÄYTETTY SANASTO

Breakpoint	Tyylitiedostoihin määriteltävä leveysmääre, jota käytetään responsiivisilla verkkosivuilla.
CSS	Verkkosivuilla käytetty tyylitiedoston koodauskieli.
Drupal	Avoimen lähdekoodin sisällönhallintajärjestelmä. Drupal perustuu PHP –ohjelmointikieleen.
Drush	Drupalin komentorivityökalu, jolla voidaan suorittaa komentoja Drupalissa ilman tarvetta itse sivustolla käymiseen.
Javascript	Ohjelmointikieli jolla voidaan tehdä dynaamisia elementtejä verkkosivuille. JavaScript prosessoidaan käyttäjän selaimessa.
jQuery	Avoimen lähdekoodin JavaScript –kirjasto. jQuerya käytetään esimerkiksi animaatioiden luomiseen verkkosivun elementeille.
Media query	CSS –attribuutti johon määritellään breakpoint, ja mitä verkkosivulla tapahtuu kun näytön leveys ylittää tai alittaa sen.
Meta tag	Tarjoaa tietoa sivustosta selaimille ja hakukoneille. Meta tag ei näy verkkosivun selaajalle.
MySQL	Avoimen lähdekoodin relaatiotietokanta.
PHP	Avoimen lähdekoodin ohjelmointikieli, jota käytetään dynaamisten verkkosivujen teossa. PHP prosessoidaan verkkopalvelimella.
SASS	Laajennus CSS –kieleen, millä voidaan pitää tyylitiedostot järjestelmällisempinä ja nopeammin käsiteltävinä.
Viewport	Verkkosivun näkyvä alue selaimessa.

1 JOHDANTO

Internetin käyttömäärä mobiililaitteilla on kasvanut paljon viime vuosina. Yhdysvalloissa yli puolet internetmedian käytöstä tapahtuu älypuhelimilla tai tabletilla (Bosomworth 2015). Suomessakin internetin käyttö mobiilisti on selvässä kasvussa, sillä 90% prosenttia älypuhelimen omistajista käyttää sitä internetin selaamiseen (SVT 2015). Näiden lisäksi Google aloitti vuoden 2015 huhtikuussa suosimaan mobiiliystävällisiä sivustoja hakutuloksissaan (Kocher 2015). Näistä syistä voidaan päätellä, että mobiiliystävällisille verkkosivuille on nykypäivänä paljon tarvetta, minkä takia kaikkien verkkokehittäjien pitäisi niitä osata luoda.

Mobiiliystävällisten verkkosivujen luomiseen on olemassa kaksi tapaa. Tavallisen verkkosivun jatkeeksi voidaan luoda erillinen mobiilisivusto, johon käyttäjä ohjataan kun hän saapuu sinne mobiililaitteella. Tämä ratkaisu on toimiva, mutta sille on olemassa vähemmän työtä vaativa, ja kustannustehokkaampi vaihtoehto (Ghazarian 2014.) Ethan Marcotte esitteli termin ”responsive web design” vuonna 2010 A List Apart –nimisessä verkkolehdessä (Wikipedia). Responsiivinen verkkosivusuunnittelu (responsive webdesign) tarkoittaa sitä, että sivusto ja sen sisältö skaalautuvat käyttäjän laitteelle sopivaksi (Marcotte 2010).

Tässä opinnäytetyössä käsittelen verkkosivujen responsiivista suunnittelua ja toteutusta, sekä sivuston toteuttamista Drupal -sisällönhallintajärjestelmällä. Työn tarkoituksena on toteuttaa responsiivinen verkkosivu-uudistus nosto- ja kuljetusalan yritykselle.

2 RESPONSIIVISET VERKKOSIVUT

Ennen internetiä selattiin pääasiallisesti vain tietokoneilla. 2000-luvun puolivälissä älypuhelimet alkoivat yleistyä, ja niissä olevat internetselaimet kehittyivät sellaiseksi, että niillä saatettiin selata samoja sivustoja kuin tietokoneillakin. Haasteena tässä kuitenkin oli se että puhelinten näytöt olivat kapeita, ja verkkosivujen elementit liian pieniä sormella käytettäväksi (Soojian 2015.)

Responsiivinen verkkosivu tarkoittaa sitä, että sivusto skaalautuu lukijan näytön leveyden mukaan. Toisin kuin erikseen rakennetussa mobiilisivustossa, selain lataa laitteesta riippumatta aina saman sivuston (Knight 2011.) Sivuston metatietoihin (meta tag) lisättävä viewport-argumentti lukee käyttäjän laitteen näytön resoluution ja kertoo selaimelle minkä kokoiseksi verkkosivu pitää skaalata (Paulund 2013).

```
<meta name="viewport" content="width=device-width">
```

Kuva 1. Esimerkki viewport –meta-tiedosta, joka määrittää sivuston leveydeksi maksimissaan laitteen näytön leveyden.

2.1 Responsiivisuuden hyödyt ja haasteet

Responsiivisuuden hyödyt ovat sen tehokkuudessa. Verrattuna siihen, että tehtäisiin verkkosivun lisäksi erillinen mobiilisivusto, responsiivinen sivusto on huomattavasti halvempi vaihtoehto koska kehitettävänä on vain yksi sivusto. Samasta syystä myös sivuston päivittäminen on yksinkertaisempaa. Jos sivustoja on kaksi, siinä tapauksessa päivitettävä on kaksi sisältöä. Mobiililaitteilla verkkoyhteydet ovat usein hitaita. Erillinen mobiilisivusto vaatii URL -uudelleenohjauksen, joka hidastaa sivuston latautumista. Responsiivinen verkkosivu on tästä syystä nopeampi vaihtoehto, koska se ei tarvitse kuin yhden URL –osoitteen (Ghazarian 2014.) Hakukoneiden kannalta responsiivinen verk-

kosivu on paras vaihtoehto. Useimmat erilliset mobiilisivut eivät pärjää hakutuloksissa, sillä ne jäävät normaalin työpöytä-versionsa varjoon. Responsiivisessa verkkosivussa kaikki sisältö on yhdessä paikassa, joten sivusto löytyy varmemmin hakukoneilla (Francis 2014.)

Yksi sivusto kaikille laitteille saattaa myös osoittautua haasteelliseksi, varsinkin kehittäjälle. Erilaisille näyttöleveyksille täytyy miettiä sisällön painotusta, esimerkiksi mobiilissa tärkeä informaatio pitää saada mahdollisimman ylös. Kehittäjän täytyy miettiä miten saa sisällön asettumaan toivotulla tavalla kun näyttö kapenee. Käyttäjäkokemus saattaa myös osoittautua haasteeksi. Erillisellä mobiilisivustolla voidaan keskittyä ainoastaan mobiilikäyttäjiin, jolloin saadaan paras käyttökokemus juuri niille käyttäjille. Responsiivisella sivustolla kehittäjän täytyy ottaa huomioon kaikki käyttäjät ja laitteet, jolloin hyvin todennäköisesti joudutaan tekemään kompromisseja elementtien asettelussa, jotta ne saataisiin asettumaan hyvin myös pienemmällä näyttökoolla. (Ghazarian 2014.)

2.2 Responsiivinen suunnittelu

Responsiivisessa suunnittelussa täytyy ottaa huomioon hyvä käyttäjäkokemus. Mobiililaitteilla sivustoa selataan sormella, minkä takia linkkien tulee olla suurempia. Mobiililaitteiden internetyhteydet ovat usein heikompia kuin kotitietokoneella, minkä takia täytyy miettiä sivuston latausaikaa. Jos kuvia on paljon ja ne ovat suurikokoisia, sivuston lataaminen saattaa kestää useita kymmeniä sekunteja.

Mobiililaitteilla yksi suurimmista haasteista responsiivisessa suunnittelussa on tilan käyttö. Tabletilla on vielä mahdollista käyttää kaksipalstaista sivusuunnittelua, mutta puhelimella selatessa näyttö on niin kapea että sivuston kaikki elementit täytyy asetella allekkain. Sivustoista tulee usein todella pitkiä, minkä takia kehittäjän täytyy miettiä että voisiko jotain elementtejä karsia pois. Varsinaista sisältöä ei kannata poistaa, mutta esimerkiksi mainokset jotka isoilla näyttöleveyksillä ovat luonteva osa sivua, saattavat viedä kapeilla näytöillä vain turhaa tilaa. Priorisoitavaa sisältöä joka kannattaa sijoittaa mahdollisimman ylös, on

esimerkiksi yhteystiedot. Koska tilaa leveyssuunnassa on rajoitetusti, myös valikko kannattaa piilottaa näkyvistä, esimerkiksi erillisen napin taakse (Boyd 2015.)

Usein suunnitteluvaiheessa on vaikeaa asetella sivuston elementtejä niin että mobiilissa ne asettuisivat hyvään prioriteettijärjestykseen. Tähän ongelmaan ratkaisu on niin kutsuttu ”mobile first”. Mobile first –tekniikassa suunnittelu aloitetaan nimensä mukaisesti mobiili edellä. Ensin suunnitellaan miltä sivusto näyttää puhelimella, ja sijoitetaan tärkein sisältö ylös. Pienimpään näyttökokoon laitetaan vain kaikki käyttäjälle olennainen tieto (Sexton 2015.) Mobile first –ajattelutapa on hyvä sillä, tällöin sisällön karsiminen vain kaikkein tärkeimpään on tehty jo projektin alussa, eikä sivustoa olla vielä täytetty kuvilla tai muilla graafisilla elementeillä.

2.3 Responsiivinen toteutus

Responsiivisten verkkosivujen keskiössä on sivuston koodiin määriteltävät breakpointit. Breakpoint on leveysmääre joka kertoo verkkoselaimelle, että missä leveydessä sivuston ulkoasussa tapahtuu muutoksia kun sitä skaalataan isommaksi tai pienemmäksi. Breakpointien määrittelyyn ei ole oikeaa eikä väärää tapaa, vaan niitä määritellään tarpeen mukaan, niin että sivusto näyttää hyvältä mahdollisimman usealla laitteella. Yleisenä ohjeena voidaan kuitenkin pitää, että määritellään breakpointit ainakin puhelinta ja tablettia varten. Se mitä breakpointin sisällä tapahtuu, määritellään sivuston CSS-tiedostoissa (Hay 2013.) CSS-tiedostot (cascading style sheet) ovat tyylitiedodtoja, joissa määritellään verkkosivun elementtien asettelu ja ulkonäkö (Rouse 2005).

2.3.1 Responsiivisuuden toteutus CSS media queryilla

Media queryt ovat sivuston CSS –koodiin lisättäviä kyselyjä jotka ilmoittavat selaimelle miltä sivusto näyttää eri näyttöleveyksillä. Media queryyn on mahdollista määrittää useita erilaisia ominaisuuksia. Näitä ominaisuuksia ovat leveys-

määreet, minimileveys (min-width) tai maksimileveys (max-width) (Frost 2013.) Siinä missä ”max-width” ja ”min-width” vastaavat selaimen leveyttä, on myös mahdollista viitata koko laitteen (device) näytön leveyteen ”max-device-width” tai ”min-device-width” ominaisuuksia käyttäen. Leveyksien haluttu arvo ilmoitetaan pikseleissä, prosenteissa tai em –arvona. Näiden lisäksi media queryyn voidaan määrittää myös lukulaitteen, kuten esimerkiksi tabletin asento (orientation), jonka arvo voi olla pysty (portrait) tai vaaka (landscape). Koodissa media query koostuu ”@media” –attribuutista, jonka perään kirjoitetaan ominaisuus ja arvo joihin selaimen halutaan reagoivan (Reese 2015.)

SASS (Syntactically Awesome StyleSheets) on erikseen tietokoneelle asennettava laajennus perinteiseen CSS-koodiin, ja sillä voidaan jäsennellä tyylitiedostojen koodia helpommin ja nopeammin käsiteltäväksi. SASS kirjoitetaan omaan ”.scss” –tiedostoonsa ja se näyttää CSS -koodilta mutta selain ei pysty sitä sellaisenaan käsittelemään, joten se käännetään CSS –koodiksi erilliseen tiedostoon esimerkiksi komentorivillä. Tavallisen CSS-koodin heikkoutena on, että siinä joudutaan usein toistamaan samaa koodia useaan kertaan. Jos HTML-elementin sisällä on useampi muu elementti, ja niiden kaikkien tyyliä täytyy muokata, niin perinteisessä CSS –koodissa isäntä-elementin tunnus täytyy kirjoittaa useaan kertaan jokaisen lapsi-elementin kohdalla, joka tuottaa paljon saman koodin toistoa. SASS –koodilla voidaan jäsennellä elementit ja niiden sisällä olevat muut elementit, kuin myös media queryt sisäkkäin, jolloin säästyy paljolta ylimääräiseltä kirjoittamiselta. SASS mahdollistaa myös erilaisten funktioiden luomisen ja koodin tuomisen muista tiedostoista (Coron 2015.)

2.3.2 Responsiivisuuden toteutus javascriptilla

Toisena vaihtoehtona responsiivisuuden toteuttamiseen on käyttää siihen javascriptia. Javascript on web-ohjelmointikieli, jolla pystytään luomaan dynaamisia ja interaktiivisia elementtejä verkkosivuille (TechTerms 2014). Javascriptilla toteutettuja lisäosia, joilla voi tehdä sivustosta responsiivisen on useita. Dynaamisuutensa ansiosta, javascriptilla on mahdollista ladata sivustolle oma CSS -

tyylitiedosto jokaiseen breakpointiin erikseen. Tähän tarkoitukseen sopiva javascript -lisäosa on esimerkiksi Adapt.js –kirjasto. Adapt.js laskee selaimen viewportin leveyden, kun sitä skaalataan isommaksi tai pienemmäksi, ja kertoo sille mitä CSS –tiedostoa sen tulee lukea. Adapt.js sisältää javascript-tiedoston, jossa on määritelty valmiit breakpointit, ja valmiit CSS –tiedostot breakpointeja varten. Breakpointeja on mahdollista myös muokata omien tarpeiden mukaan (Webmonkey 2011.)

Näytön leveyden lisäksi, sivustoa on mahdollista skaalata myös lukulaitteen tyypin mukaan. Restive.js on javascript –lisäosa, joka tarkkailee selaimen viewportin leveyttä, mutta sen lisäksi myös käytettävää laitetta. Restive.js antaa sivuston <body> -osalle CSS –luokkia viewportin leveyden ja käyttäjän laitteen perusteella, joiden avulla muokataan sivuston ulkoasua eri kokoisille näytöille. Restive.js tarvitsee lisäksi javascriptin jQuery –kirjaston toimiakseen. jQuery voidaan ladata sivustolle esimerkiksi linkittämällä se HTML –tiedoston <head> -osassa (Hill 2013.)

2.4 Responsiivisuuden testaaminen

Verkkosivun testaaminen ennen sen julkaisua on aina tärkeää. Responsiivisen verkkosivun testaaminen pelkästään eri selaimilla ei riitä, vaan sitä täytyy myös testata erilaisilla laitteilla. Harvalla kehittäjällä kuitenkaan on käytettävänään kaikkia laitteita joilla verkkoa voi selata. Tätä varten on olemassa erilaisia emulaattoreita, joilla voidaan simuloida sivuston selaamista erilaisilla laitteilla. Tällaisia ovat esimerkiksi Googlen Chrome -selaimessa oleva emulaattori-näkymä, jolla voidaan simuloida eri sivustoa eri laitteilla ja myös erilaisilla yhteyksillä (Setter 2014). Tämän lisäksi internetissä on useita sivustoja joissa voi testata sivustonsa responsiivisuutta, tällainen sivusto on esimerkki Browserstack. Browserstackissa voi testata omaa sivustoa lukuisilla eri selaimilla ja laitteilla (Wikipedia). Emulaattori ei ole sama asia kuin varsinainen laite, mutta se on hyvin suuntaa antava keino testaamiseen.

3 DRUPAL -SISÄLLÖNHALLINTAJÄRJESTELMÄ

Drupal on avoimeen lähdekoodiin perustuva sisällönhallintajärjestelmä. Drupal on kirjoitettu PHP-ohjelmointikielellä, ja sen on alun perin luonut Dries Buytaert vuonna 2001. Nykyään sitä kehittää yli miljoona rekisteröitynyttä käyttäjää käsittävä Drupal-yhteisö (Drupal Community). Drupalin uusin ja tähän mennessä suurin versio, Drupal 8, julkaistiin 19. Marraskuuta 2015 (Wikipedia.)

Drupalin kolme tärkeintä osaa ovat sen ydin (core), moduulit (modules) ja teemat (themes). Drupal core pitää sisällään drupalin perusasennuksen, johon kuuluu sen lähdekoodi, muutamia moduuleita (core modules) ja teemoja (core themes) (Drupal 2003.) Drupal asennetaan lataamalla se projektin verkkosivuilta, ja purkamalla ladattu paketti drupalille sopivalle web-palvelimelle kuten esimerkiksi Apache. Drupalia varten tarvitaan myös tietokanta kuten esimerkiksi MySQL (Drupal 2003.)

Perusasennusta voidaan laajentaa asentamalla moduuleita, joilla lisätään sivustolle erilaisia ominaisuuksia kuten esimerkiksi kuva-gallerioita tai tekstieditoreita. Syyskuussa 2015 drupaliin oli asennettavissa yli 31000 drupal- yhteisön kehittämää moduulia (contributed modules). Mikäli omiin tarpeisiin ei löydy sopivaa moduulia drupalin verkkosivuilta, niitä on myös mahdollistaa kehittää itse juuri omiin tarpeisiin sopivaksi (custom modules).

Drupalin teemat ovat sivuston ulkoasun muokkaamista varten. Sivuston rakennetta muutetaan muokkaamalla teemasta löytyvää page.tpl.php –tiedostoa, käyttämällä HTML- ja PHP –ohjelmointikieliä. Ulkoasun muokkaukseen käytettään CSS -tyylitiedostoja. Drupalin verkkosivuilla on ladattavissa yli 2000 valmista teemaa, mutta niitä voi myös rakentaa itse (Drupal 2012.) On myös mahdollista käyttää valmista teemaa drupalin teemakirjastosta, ja rakentaa sen pohjalta aliteema (sub-theme). Aliteema käyttää pääteemansa (base-theme) koodia, ja aliteemaan lisätään vain ne tiedostot joita halutaan muokata. Jokaisessa teemassa, mukaan lukien aliteemat, on ”.info” –tiedosto. Tähän tiedostoon kerrotaan drupalille mistä käytettävät tyylitiedostot ja javascript-tiedostot löytyvät.

Siihen määritellään myös käytettävät lohko-alueet (regions), joihin voidaan osoittaa sivuston eri elementtejä drupalin hallinnassa. Info –tiedostossa määritetään myös mitä teemaa aliteema käyttää pohjana (Drupal 2008.) Aliteema voidaan luoda käsin, tekemällä itse kaikki tarvittavat tiedostot, mutta joihinkin teemoihin on mahdollista luoda aliteema myös automatisoidusti drushilla. Drush komentorivi-työkalu drupalia varten, ja sillä voidaan suorittaa erilaisia toimintoja drupalissa, kuten esimerkiksi moduulien tai teeman latauksen ja asennuksen, ilman tarvetta tehdä niitä varsinaisella sivustolla (Digital Ocean 2013).

Responsiivisuus toteutetaan drupal-sivustolla samalla tavalla kuin missä tahansa muussakin verkkosivustolla. Drupal tarjoaa kehittäjälle kuitenkin apuvälineitä responsiivisen sivuston rakentamiseen. Useissa teema-kirjaston tarjoamissa teemoissa, kuten ”Zen” tai ”Bootstrap”, on valmis responsiivinen rakenne. Responsiivisia sivuja varten on käytettävissä myös erilaisia moduuleita, joilla voi luoda esimerkiksi responsiivisia valikoita (Drupal 2011.)

4 CASE: NOSTOKONEPALVELU OY

Nosto- ja kuljetuspalvelu yritys Nostokonepalvelu OY halusi lähteä uudistamaan verkkosivujaan tätä päivää vastaavaksi ja responsiiviseksi. Edelliset sivut eivät ole responsiiviset, vaan mobiililla selattaessa esiin tulee erikseen rakennettu mobiilisivusto. Sivusto on jo rakennettu drupalin päälle joten, sivustouudistus toteutetaan vain tekemällä uusi teema drupaliin. Uusi teema toteutetaan graafikon tekemän mallin mukaan.

Kuva 2. Kuvankaappaus vanhasta mobiilisivustosta.

Kuva 3. Kuvankaappaus vanhasta sivustosta tietokoneella selattaessa.

4.1 Drupalin asennus

Drupal asennetaan lataamalla se palvelimelle www.drupal.org -sivustolta. Vaikka drupalin uusin versio (Drupal 8) on jo julkaistu, käytämme projekteissamme vielä vanhempaa Drupal 7 –versiota, tarkemmin ottaen 7.43. Kun drupal on ladattu palvelimelle, jatkan asennusta siirtymällä verkko-osoitteeseen jossa palvelin sijaitsee. Selaimessa tulee näkyviin drupalin asennutyökalu, jossa määritetään tietokanta jota drupal tulee käyttämään, ylläpitäjän tiedot ja salasana, sekä sivuston perustietoja kuten esimerkiksi kieli. Tämän jälkeen sivusto on valmis personoitavaksi asiakkaan tarpeiden mukaiseksi.

4.2 Uuden teeman luonti

Aloitan teeman luonnin tekemällä drupalin ”Zen”-teemalle aliteeman ”NKP_2015”. Päädyin Zeniin, koska olen käyttänyt sitä paljon ennenkin ja siinä on SASS -valmius. Aliteeman luomiseen Zenille on olemassa yksinkertainen

drush-komento, jolla automatisoidaan koko aliteeman luonti prosessi. Komentoon määritellään teeman nimi ja koneluettava nimi, joka on yleensä yksinkertaisempi joten sitä on mukavampi käyttää koodissa. Näiden lisäksi komentoon lisätään sijainti johon teema halutaan asentaa.

```
drush zen "Uusi teema" uusi_teema path=sites/all/themes
```

Kun teema on asennettu, se voidaan ottaa käyttöön kirjautumalla sisään drupalin hallintaan ja valitsemalla ylläpitovalikosta ”Näyttöasetukset”, tai jos käyttöliittymä on englanninkielinen ”Appearance”.

4.3 Teeman rakenteen muokkaaminen

Uuden aliteeman luomisen jälkeen, alan muokkaamaan sivun perusrakennetta graafisen mallin mukaiseksi. Muokkaamista varten kopioin pohjateemassa olevan page.tpl.php –tiedoston omaan aliteemaani, jota muokkaan oman tarpeeni mukaiseksi. Tässä projektissa tarvitsen seitsemän lohko-aluetta, jotka olen määrittänyt teeman info-tiedostossa.

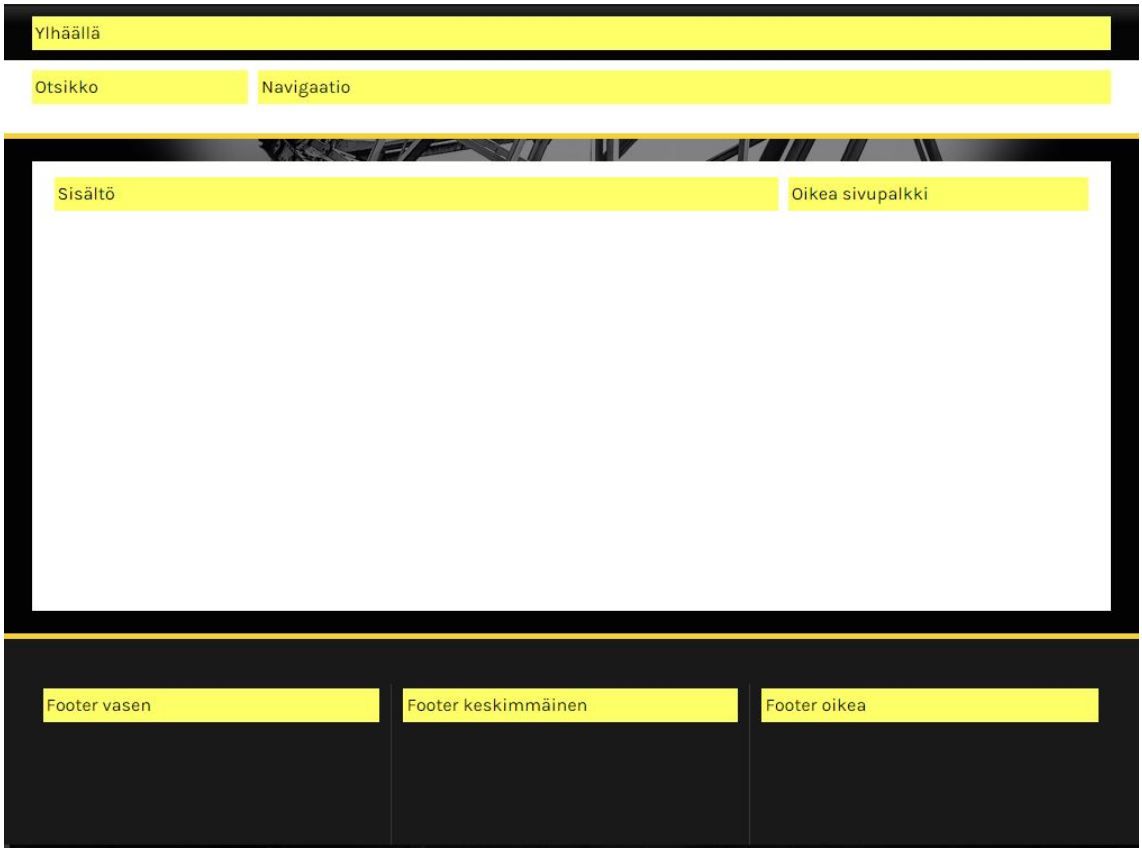
```
regions[top]            = Top
regions[header]         = Header
regions[navigation]     = Navigation bar
regions[help]           = Help
regions[content]        = Content
regions[sidebar_second] = Second sidebar
regions[footer_first]   = Footer First
regions[footer_second]  = Footer Second
regions[footer_third]   = Footer Third
```

Kuva 4. Kuvankaappaus .info -tiedoston lohko-alue määrittelyistä.

Luon sivun perusrakenteen kopioimaani teeman page.tpl.php –tiedostoon käyttämällä HTML –kieltä, jolla luon elementit ja tulostan lohko-alueet PHP:lla niiden sisään.

```
<div id="page-top">
  <div class="container clearfix">
    <?php print render($page['top']); ?>
  </div>
</div>
```

Kuva 5. Kuvankaappaus page.tpl.php -tiedostosta. Koodissa määriteltynä HTML -elementti "page-top", jonka sisään tulostetaan "top" lohko-alue.



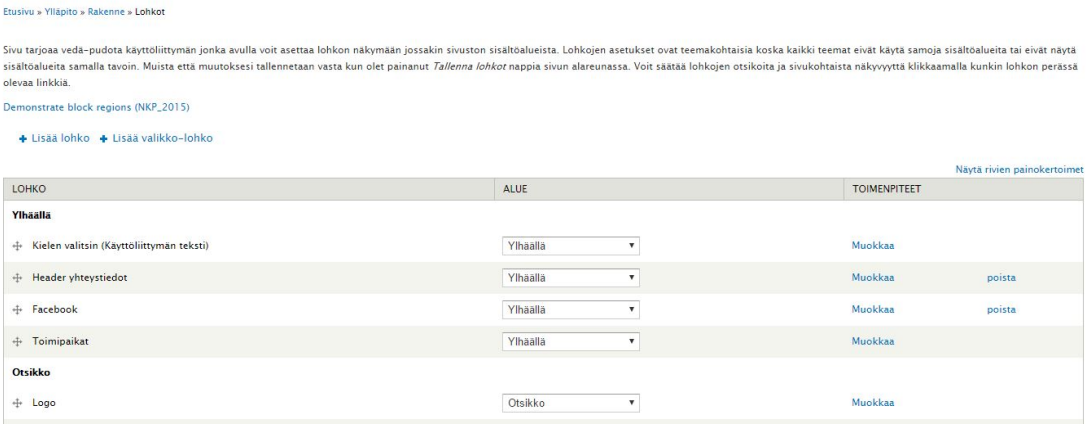
Kuva 6. Kuvankaappaus lohko-alueista aseteltuna oikeille paikoilleen.

4.4 Sivuston rakentaminen

Kun sivuston perusrakenne on kunnossa, alan rakentaa itse sivustoa ja sen sisältöä. Aloitan tekemällä sivustolle elementit jotka tulevat jokaiselle sivulle. Näitä ovat logo, ylätunnisteen kielivalinnat ja yhteystietolinkit sekä alatunnisteen kolme lohkoa. Navigaatioon käytän Superfish –moduulia, jolla on monia eri ominaisuuksia ja asetuksia esimerkiksi responsiivituutta ja kosketusnäyttöjä varten. Sisältöä sivustolta löytyy jo valmiiksi, sillä siirsin ennen työn aloittamista

nykyiseltä sivustolta olemassa olevan materiaalin, mikä helpottaa alkuun pääsyä kun niitä voi käyttää uusien elementtien luonnissa.

Graafisen mallin mukaan etusivulle tulee neljä sisältöelementtiä, jotka ovat slogan- sekä pikalinkki-elementit, uusimmat referenssit ja uusimmat uutiset. Sloganin ja pikalinkit luon tavallisena lohkona (custom block), joka on Drupalin ytimessä valmiina oleva ominaisuus. Uusi lohko luodaan valitsemalla ylläpitovalikosta: ”Lisää lohko”. Lohkon luonti-näkymässä on tekstieditori, johon voidaan lisätä tavallista tekstiä, tai HTML-koodia jota käytän tässä tapauksessa. Lohkot sijoitellaan paikalleen hallintapaneelin graafisessa käyttöliittymässä.



Kuva 7. Kuvankaappaus lohkojen hallintanäkymästä

Referenssejä ja uutisia varten tarvitsen Views -moduulin, jolla voidaan tehdä sivustolle näkymiä, joissa voidaan näyttää esimerkiksi uusimmat uutis-sisältötyypin tulokset. Käytännössä Views suorittaa SQL-kyselyitä, mutta siinä on helppokäyttöinen graafinen käyttöliittymä, jolloin koodia ei tarvitse kirjoittaa. Teen kaksi erillistä näkymää, molemmille sisältötyypeille omansa. Näkymästä valitaan suodattimeksi sisältötyyppi jonka sisältä tulokset haetaan. Sen jälkeen valitaan kentät jotka halutaan näyttää, ja lisäksi järjestelykriteeriksi uusin sisältö ensin. Tämän jälkeen rajoitetaan vielä tulosten määrä kolmeen. Uusi näkymä voidaan tehdä uutena sivuna tai lohkona, kuten tässä tapauksessa teen, ja silloin se sijoitellaan paikalleen samasta paikasta kuin muutkin lohkot.

Etusivun elementtien lisäksi muille sisältösivuille lisään sivupalkkiin valikkolohkon alalinkkejä varten. Käytän tämän luomiseen Menu Block –moduulia. Valikkolohkon lisäksi alasivuille ei tarvitse luoda muita elementtejä, sillä leipätekstialue on kaikissa samanlainen.

4.5 Teeman ulkoasun muokkaaminen

Aloitan teeman ulkoasun muokkaamisen tekemällä sivuston väreille ja fonteille valmiit SASS –funkiot. Funktioiden luomisen etuna on esimerkiksi se, jos värejä päätetäänkin jostain syystä muuttaa niin minun ei tarvitse vaihtaa sitä jokaiseen kohtaan koodissa, joita saattaa hyvinkin olla kymmeniä, vaan riittää että muutan sen paikkaan jossa funktion arvo määritellään.

4.5.1 Teeman perusrakenne

Seuraavaksi poistan riippuvuuden drush -komennon automaattisesti luomaan responsiiviseen pohjaan, koska aioin luoda sen itse. Tämän teen siten, että poistan teeman style.scss –tiedoston alusta rivin joka tuo responsiivisen pohjan teemaan. Tämä poistaa teemasta kaikki valmiit tyylit, jonka takia kaikki sivun elementit asettuvat allekkain selaimessa. Haluan saada elementit asettumaan siististi keskelle. Tätä varten olen luonut CSS –luokan, ”.container”, jonka olen antanut sivuston elementtejä ympäröiville isäntä-elementeille. Tähän luokkaan määrittelen sivun sisältöalueen leveyden eri näyttökokoja varten, ja asettelen sen keskelle näyttöä.

```
.container {
    width:960px;
    margin-left:auto;
    margin-right:auto;
    position: relative;
    @media (max-width:959px) {
        width: 768px;
    }
    @media (max-width:767px) {
        width: 480px;
    }
    @media (max-width:479px) {
        width: 100%;
    }
}
```

Kuva 8. Kuvankaappaus ".container" CSS -luokasta, ja sen tyylimäärittelyistä eri näyttöleveyksillä.

Tämän jälkeen alan asettelemaan sivun lohko-alueita, joihin olen jo sijoitellut niiden elementit, vastaamaan graafista mallia. Ylimpänä sivulla on alue nimeltä ”Ylhäällä”. Tämän sijoittelulle ei tarvitse tehdä mitään sillä sen kuuluukin olla ylimpänä ja koko sivun leveydellä. Seuraavana asettelen vierekkäin alueet ”Otsikko”, jossa on asiakkaan logo sekä ”Navigaatio”, joka nimensä mukaisesti navigaatiota varten. Varsinaiselle sisältö-alueelle määrittelen leveän leipäteksti-osion, ja kapeamman sivupalkin, johon olen sijoittanut alasivujen apunavigaation. Lisäksi sivupohjassa oikea sivupalkki on määritelty siten, että jos siihen ei ole osoitettu mitään, silloin sitä ei tulosteta sivulle ollenkaan ja leipäteksti-osio on tällöin koko sivun levyinen. Sisältö-alueen alle sivun alatunnisteen kolme lohko-aluetta asettelen vierekkäin, kaikki saman levyisiksi.

4.5.2 Rakenteen ja elementtien responsiivisuus

Tämän projektin responsiivisuuden luonnissa käytän tekniikkana CSS media queryja. Sivustolla elementti joka tarvitsee eniten asettelua mobiilia ajatellen on navigaatio Sisältöalueen pikalinkki-lohko, alatunnisteen kolme lohko-aluetta, ja etusivun uutis- sekä referenssinäkymät asetellaan siten, että kun niiden sisällä olevat elementit eivät enää mahdu olemaan vierekkäin, ne määritellään asettumaan allekkain. Lisäksi alasivujen mahdollinen sivupalkki piilotetaan mobiilikäyttäjiltä, koska kaikki linkit tulevat hyvin käytettäväksi uuteen mobiilivalikkoon.

```scss
.views-row {
    float: left;
    width: 33.33%;
    display: block;
    padding:0 10px;
    @media (max-width:959px) {
        width: 48%;
        &.views-row-3 {
            display: none;
        }
    }
    @media (max-width:767px) {
        width: 100%;
        margin: 10px 0;
    }
}
```

Kuva 9. Kuvankaappaus style.scss -tiedostosta, missä ilmenee näkymän yksittäisen tuloksen asettelu.

Yllä olevassa kuvassa näkyy kuinka näkymien yksittäisen elementin leveys muuttuu ensimmäisessä breakpointissa isommaksi, ja samalla myös kolmas elementti piilotetaan tilan säästämiseksi kapeilla näytöillä. Seuraavassa breakpointissa elementti määritellään koko ruudun leveydelle.

Navigaation responsiivisuus vaatii hyvän käytettävyyden takaamiseksi hieman enemmän työtä kuin muut elementit. Tavallisella tietokoneen näytöllä navigaation linkit ovat vierekkäin, ja vietäessä hiiren osoitin päälle, osassa linkeistä on alalinkkejä jotka tulevat esiin päälinkin alle.

Kuva 10. Kuvankaappaus etusivun navigaatio-elementistä.

Mobiililaitteiden kannalta tämä on ongelmallinen siksi, että linkit eivät mahdu olemaan vierekkäin kapealla näytöllä ja kosketusnäytöllä pudotusvalikko ei ole paras mahdollinen ratkaisu. Näistä syistä määrittelen tyylitiedostoon, että tablet-ti-koossa päävalikko piilotetaan, ja sen tilalle tulee Menu Block –moduulilla teht-y uusi valikko, mutta sekään ei tule automaattisesti näkyviin. Olen tehnyt pa-ge.tpl.php –tiedostoon HTML –elementin, joka näytetään käyttäjälle siinä vai-heessa kun käyttäjä tulee sivustolle tabletilla tai sitä pienemmän näytön omaa-valla laitteella. Elementtiin olen liittänyt jQuery –animaation, ja se toimii nappina jolla käyttäjä saa mobiilivalikon esiin. Napissa on valikon merkiksi kolme viivaa allekkain, ja sitä käytetään yleisesti navigaation symbolina monilla verkkosivuil-la.

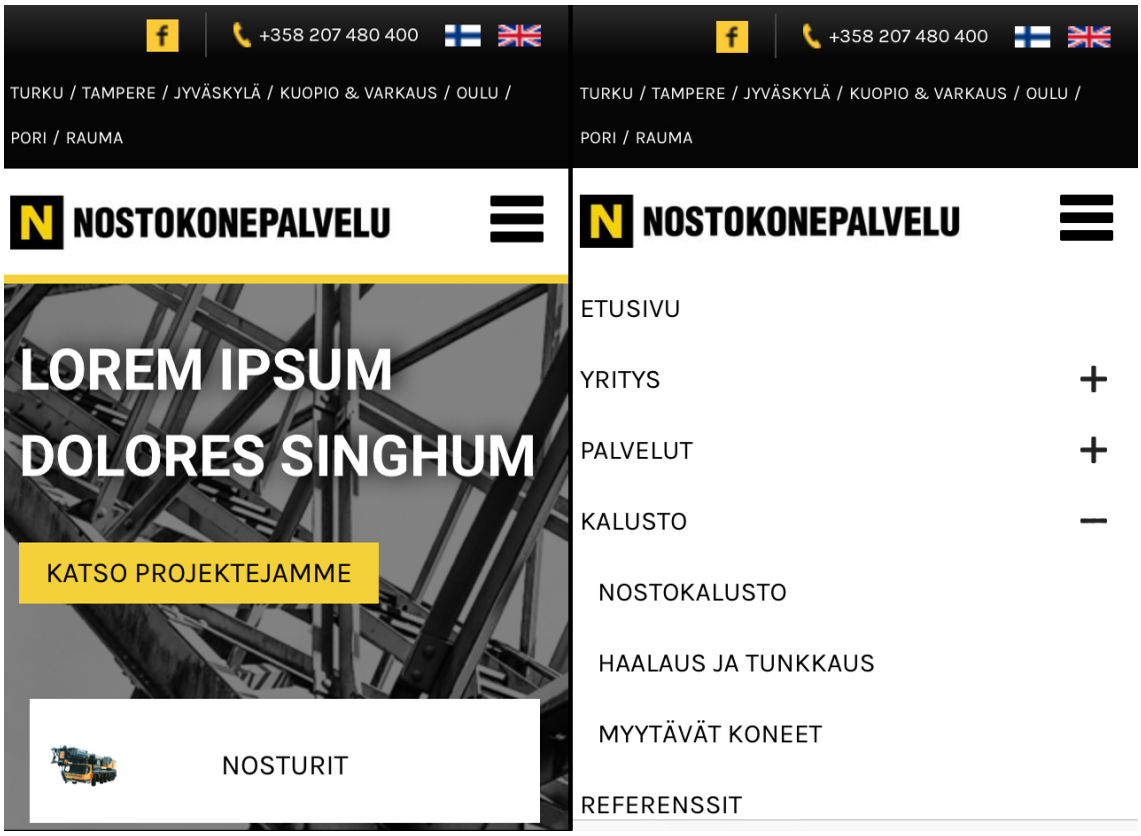
```
<div id="navigation">
  <?php print render($page['navigation']); ?>
  <div class="toggler">
    <div class="line"></div>
    <div class="line"></div>
    <div class="line"></div>
  </div>
</div>
```

```
$('.toggler').click(function(){
    $('#block-menu-block-5').slideToggle(500);
});
```

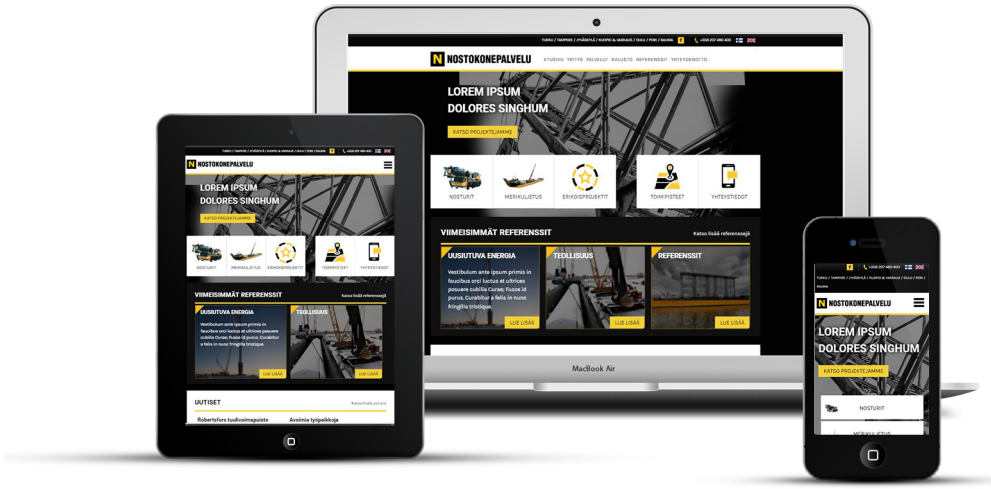
```
.toggler {
    background:#fff;
    display: block;
    padding:5px;
    margin-top: 13px;
    display: none;
    position: absolute;
    right: 0;
    top: 0;
    @media (max-width:767px) {
        display: block;
    }
    @media (max-width:479px) {
        margin-top: 6px
    }
    .line {
        background:#000;
        width: 30px;
        height: 6px;
        margin:4px;
        display: block;
        border-radius:1px;
    }
}
```

Kuva 11. Kuvankaappaus mobiilivalikon koodista. Vasemmalla ylhäällä napin HTML -rakenne. Oikealla tyylimääritykset, ja vasemmalla alhaalla jQuery -animaatio.

Uutta mobiilinavigaatiota tehdessä valitsin asetuksista että tässä valikossa näy-tetään kaikki linkit, myös alalinkit, allekkain. Tämä siksi että sivupalkki on piilo-tettu mobiilikäyttäjiltä, mutta heillä täytyy kuitenkin olla pääsy sivuston kaikkeen sisältöön. Valikosta tulee tästä syystä todella pitkä, joten piilotan tyylitiedostossa kaikki alalinkit, ja teen siihen jQuerya käyttäen napit jolla ne saa tarpeen tullen näkyviin.



Kuva 12. Kuvankaappaus navigaatiosta puhelimella selattaessa. Vasemmalla navigaatio kiinni, ja oikealla avattuna.



Kuva 13. Mockup -kuva sivustosta eri laitteilla.

5 YHTEENVETO

Verkon selaaminen tapahtuu tänä päivänä yhä useammin älypuhelimilla ja tableteilla. Verkkosivun käyttäminen mobiililaitteiden kapeilla kosketusnäytöillä ei onnistu, mikäli sitä ei optimoida laitteille sopivaksi. Työn tarkoituksena oli tutkia mitä tulee ottaa huomioon responsiivisen verkkosivun suunnittelussa, ja eri keinoja miten se voidaan teknisesti toteuttaa. Lisäksi tarkoituksenani oli kertoa miten responsiivinen verkkosivu voidaan toteuttaa Drupal –sisällönhallintajärjestelmällä.

Mielestäni tällä hetkellä CSS media queryt ovat paras vaihtoehto responsiivisen verkkosivun toteuttamiseen. Kaikki nykyaikaiset selaimet tukevat tänä päivänä media queryja, joten niiden käyttäminen on huomattavasti käytännöllisempää kuin usean eri CSS -tiedoston tekeminen eri näyttökokoja varten. Asiakasprojektin toteutus onnistui hyvin, ja se tullaan julkaisemaan kunhan sisällöt saadaan sivustolla paikalleen.

Ala kehittyy jatkuvasti, ja sitä täytyy seurata kokoajan pysyäkseen kehityksen perässä. Responsiivinen verkkosuunnittelu on tällä hetkellä web-alan päälinja, mutta on melko vaikea sanoa mitä sen jälkeen tulee. Luultavasti verkkosivujen teossa aletaan keskittymään yhä enemmän mobiililaitteisiin, sillä niiden määrä kasvaa kokoajan.

LÄHTEET

Bosomworth, D. 2015. Mobile Marketing Statistics 2015. Viitattu 13.12.2015 http://www.smartinsights.com/mobile-marketing/mobile-marketing-analytics/mobile-marketing-statistics/.

Boyd, N. 2015. Responsive Design Best Practices. Viitattu 13.12.2015 http://www.howdesign.com/resources-education/responsive-design-best-practices/.

Coron, T. 2015. What is Sass? Guide to CSS with superpowers. Viitattu 7.3.2016 http://www.creativebloq.com/web-design/what-is-sass-111517618.

Digital Ocean 2013. A Beginner's Guide To Drush: The Drupal Shell. Viitattu 14.12.2015 https://www.digitalocean.com/community/tutorials/a-beginner-s-guide-to-drush-the-drupal-shell.

Drupal 2003. Drupal Core. Viitattu 14.12.2015 https://www.drupal.org/project/drupal.

Drupal 2003. System requirements. Viitattu 14.12.2015 https://www.drupal.org/requirements.

Drupal 2008. Creating a sub-theme. Viitattu 11.12.2015 https://www.drupal.org/node/225125.

Drupal 2011. Responsive Design. Viitattu 23.3.2016 https://www.drupal.org/node/1322126.

Drupal 2012. Theming Guide. Viitattu 11.12.2015 https://www.drupal.org/documentation/theme.

Francis, H. 2014. 4 SEO Benefits of Responsive Web Design. Viitattu 13.12.2015 http://www.searchenginejournal.com/4-seo-benefits-responsive-web-design/92807/.

Frost, B. 2013. 7 Habits of Highly Effective Media Queries. Viitattu 12.12.2015 http://bradfrost.com/blog/post/7-habits-of-highly-effective-media-queries/.

Ghazarian, A. 2014. The Pros and Cons of Responsive Web Design vs. Mobile Website vs. Native App. Viitattu 22.11.2015 http://designmodo.com/responsive-design-vs-mobile-website-vs-app.

Hay, S. 2013. Designing for Breakpoints. Viitattu 12.12.2015 http://alistapart.com/article/designing-for-breakpoints.

Hill, O. 2013. CSS Media Queries vs Restive.JS. Viitattu 30.11.2015 http://blog.restive.io/posts/4887492/css-media-queries-vs-restive-js.

Knight, K. 2011. Responsive Web Design: What It Is and How To Use It. Viitattu 13.12.2015 http://www.smashingmagazine.com/2011/01/guidelines-for-responsive-web-design/.

Kocher, J. 2015. SEO: Google to Make 'Mobile-friendly' a Ranking Signal. Viitattu 13.12.2015 http://www.practicalecommerce.com/articles/83483-SEO-Google-to-Make-Mobile-friendly-a-Ranking-Signal.

Marcotte, E. 2010. Responsive Web Design. A List Apart. Viitattu 13.12.2015 http://alistapart.com/article/responsive-web-design.

Paulund 2013. Understanding The Viewport Meta Tag. Viitattu 5.10.2015 http://www.paulund.co.uk/understanding-the-viewport-meta-tag.

Reese, R. 2015. Media Queries: Width vs. Device Width. Viitattu 12.12.2015 http://www.sitepoint.com/media-queries-width-vs-device-width/.

Rouse, M. 2005. Cascading style sheet (CSS). Viitattu 15.2.2016 http://searchsoa.techtarget.com/definition/cascading-style-sheet-CSS.

Setter, M. 2014. Better Responsive Website Testing in Google Chrome. Viitattu 13.12.2015 http://www.sitepoint.com/better-responsive-website-testing-google-chrome/.

Sexton, P. 2015. Mobile first principles. Viitattu 12.12.2015 https://varvy.com/mobile/mobile-first.html.

Soojian, C. 2015. A Brief History of Responsive Web Design. Viitattu 19.10.2015 http://engage.synecoretech.com/marketing-technology-for-growth/bid/204297/A-Brief-History-of-Responsive-Web-Design.

Suomen virallinen tilasto (SVT): Väestön tieto- ja viestintätekniikan käyttö [verkkojulkaisu]. ISSN=2341-8699. 2015, 2. Internetin käyttö mobiililaitteilla . Helsinki: Tilastokeskus [viitattu: 13.12.2015]. Saantitapa: http://www.stat.fi/til/sutivi/2015/sutivi_2015_2015-11-26_kat_002_fi.html.

TechTerms 2014. JavaScript. Viitattu 13.12.2015 http://techterms.com/definition/javascript.

Webmonkey 2011. Adapt.js Offers JavaScript Alternative to CSS Media Queries. Viitattu 27.11.2015 http://www.webmonkey.com/2011/04/adapt-js-offers-javascript-alternative-to-css-media-queries/