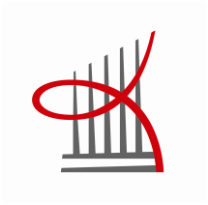


RESPONSIIVINEN VERKKOSUUNNITTELU

Sami Ylitalo

Opinnäytetyö
Kesäkuu 2014
Tietotekniikka
Ohjelmistotekniikka

TAMPEREEN AMMATTIKORKEAKOULU
Tampere University of Applied Sciences

TIIVISTELMÄ

Tampereen ammattikorkeakoulu
Tietotekniikka
Ohjelmistotekniikka

YLITALO, SAMI
Responsiivinen verkkosuunnittelu

Opinnäytetyö 31 sivua
Kesäkuu 2014

Tässä opinäytetyössä tarkoitukseni on perehtyä responsiiviseksi verkkosuunnitteluksi kutsuttuun verkkosivujen toteuttamistapaan.

Ensimmäiseksi käyn lyhyesti läpi yleisimmät verkkosivujen toteuttamiseen liittyvät tekniikat.

Seuraavaksi käyn läpi kuinka verkkosivuja tarkastellaan Internetissä ja kuinka verkkosivujen käyttäminen on muotoutunut sellaiseksi kuin se nykypäivänä. Käsittelen myös minkälaisia ongelmakohtia verkkosivujen perinteiset toteuttamistavat ovat aiheuttaneet yrittäessään vastata nykypäivän tarpeisiin.

Näihin ongelmakohtiin esittelen ratkaisuna responsiivisen verkkosuunnittelun ja sen keskeiset periaatteet.

Tämän jälkeen pureudun tarkemmin siihen, kuinka responsiiviset ominaisudet voidaan käytännössä toteuttaa verkkosivuille.

Osana opinnäytetyötäni toteutan yksinkertaisen verkkosivun, jossa esittelen responsiivisia toteuttamistapoja käytännössä.

Asiasanat: responsiivinen, verkkosuunnittelu, web-kehitys, ohjelmointi

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in ICT Engineering
Option of Software Engineering

YLITALO, SAMI
Responsive Web Design

Bachelor's thesis 31 pages
June 2014

In this thesis I am going to explore the web development paradigm known as responsive web design.

First I am briefly going through the most common web development techniques.

After that I am discussing how websites are used on the Internet and how the development of websites has formed into what it is today. I am also going to discuss what kind of problems have arisen when traditional ways of creating websites are used to meet the requirements of today.

As an answer to these problems I am going to introduce the responsive web design paradigm and it's main principles.

Next I will delve into how responsive features can be implemented into web sites in practice.

As a part of my thesis I am going to create a basic website to demostrate these responsive web development techniques in practice.

Key words: responsive, web design, programming

SISÄLLYS

1 JOHDANTO

Nykypäivänä erilaisten mobiililaitteiden käyttö on kasvanut valtavasti. Jokaiselle erilaiselle verkkosivujen tarkasteluun kykenevälle päätelaitteelle ei ole enää järkevää tehdä erillistä optimoitua sivustoa. Tämän takia nykyaikaisten verkkosivustojen tulisikin kyetä mukautumaan käytössä olevan päätelaitteen vaatimuksiin. Tätä tarvetta varten on kehitetty verkkosivujen toteuttamistapa jota kutsutaan responsiiviseksi verkkosuunnitteluksi.

Tässä opinnäytetyössä tavoitteeni on perehtyä responsiivisen verkkosuunnitelun ideaan ja sen tarkastella sen hyötyjä verrattuna perinteisiin verkkosivujen toteuttamistapoihin. Tulen tutustumaan responsiivisuuden keskeisiin tekniikoihin ja toteutan verkkosivun jossa havainnoillistan näitä tekniikoita käytännössä.

Opinnäytetyössäni luvussa 2 käyn läpi yleisimmät tekniikat, joilla nykypäivän verkkosivut toteutetaan. Luvussa 3 selostan verkkosivujen toimintatavan sekä verkkosivujen käyttämisen tilanteen nykypäivänä. Luvussa 4 perehdyn yleisesti responsiiviseen verkkosuunnitteluun ja mobiililähtöiseen ajattelutapaan. Luvussa 5 käyn läpi responsiivisten ominaisuuksien teknisen toteuttamistavat. Luvussa 6 toteutan yksinkertaisen verkkosivun jossa esittelen aiemmin käsiteltyjä tekniikoita käytännössä. Luvussa 7 käyn läpi loppupäätelmät ja työni tulokset.

Opinnäytetyöni aihe perustuu omaan ideaani tutkia tätä kyseistä aihepiiriä ja teen työn ensisijaisesti omaan käyttööni. Työni tarkoituksena on luoda havaintojeni pohjalta informatiivinen käytännön esimerkeillä varustettu dokumentti jota voidaan hyödyntää tiedonlähteenä nykyaikaisten verkkosivujen suunnittelussa ja toteuttamisessa.

2 VERKKOSIVUSTOJEN TOTEUTTAMISTEKNIIKAT

Tässä luvussa tulen käymään läpi yleisimmät verkkosivujen toteuttamistekniikat, jotka liittyvät nykyaikaisten verkkosivujen toteuttamiseen. Näitä ovat ohjelmointikielet ja kuvauskielet joilla toteutetaan verkkoselaimessa näkyvä verkkosivu ja palvelinpäässä sijaitseva logiikka. Perehdyn lyhyesti jokaisen kielen historiaan, käyttötarkoitukseen ja oleellisimpiin versioihin.

2.1 HTML

HTML (Hypertext Markup Language) on tietotekniikka-asiantuntija Tim Berners-Leen vuonna 1989 keksimä kuvauskieli, jota käytetään verkkosivujen rungon määrittelemiseen. Alkuperäinen käyttötarkoitus HTML-kielelle oli jakaa tieteellisiä dokumentteja Internet-verkkoa hyödyntäen ja mahdollistaa viittaus yhdestä dokumentista toiseen dokumenttiin hypertext-tekniikalla. Myöhemmin HTML-dokumenttien käyttö levisi yleiseen käyttöön yrityksille ja tavallisille kuluttajille. (Raggett 1998.)

HTML on kehittynyt huomattavasti uusien versioiden myötä ja laajentunut erilaisilla ominaisuuksilla. Oleellisimpina versioina voidaan pitää HTML 2.0 ja HTML 4.01. Tällä hetkellä uusin versio on HTML5. (Wikipedia 2014.)

2.2 CSS

CSS (Cascading Style Sheets) on erityisesti verkkosivudokumenttien kanssa käytetty tyylisääntökieli, jota käytetään sivujen ulkoasun muokkaamiseen ja elementien asemoimiseen. Se julkaistiin vuonna 1996 vastaamaan kehittäjien tarpeeseen muotoilla sivustojensa ulkonäköä yksityiskohtaisemmin kuin mitä HTML-kieli mahdollistaisi. (Wium Lie & Bos 1999.)

CSS:n oleelliset versiot ovat CSS 1 ja nykypäivänä käytössä oleva CSS 2.1. Uusin versio CSS 3 on otettu käyttöön mutta on kuitenkin vielä kehitysvaiheessa. (Wikipedia 2014.)

2.3 JavaScript

JavaScript on tulkattava ohjelmointikieli, jota käytetään dynaamisten ominaisuuksien luomiseksi verkkosivulle ja verkkosivun sisällön asynkroniseen muokkaamiseen. JavaScript-koodi liitetään osaksi HTML-dokumenttia ja suoritetaan verkkosivun käyttäjän selainohjelmassa. (Mozilla Developer Network 2014.)

JavaScript otettiin käyttöön ensimmäisen kerran vuonna 1995 osana Netscape-verkkoselainta. (Young 2010.)

2.4 PHP

PHP (PHP Hypertext Preprocessor) on dynaamisten verkkosivustojen luomiseen tarkoitettu ohjelmointikieli. PHP-koodi sijoitetaan verkkosivudokumenttiin HTML-koodin sekaan. Toisin kuin JavaScript-koodin kohdalla, PHP-koodi suoritetaan palvelintietokoneella verkkosivun lataushetkellä. (Rantala 2005, 9.)

PHP 1.0 julkaistiin vuonna 1995. Kirjoitushetkellä uusin versio on PHP 5.5. (Wikipedia 2014.)

3 VERKKOSIVUJEN TOIMINTA

Verkkosivujen toiminta Internetissä perustuu asiakas-palvelin-malliin. Tämä malli koostuu kolmesta erilaisesta osasta: asiakasohjelmasta, palvelinohjelmasta ja yhteysprotokollasta. Verkkosivun käyttäjä käyttää asiakasohjelmaa lähettääkseen pyyntöjä verkkopalvelimelle. Verkkopalvelimella sijaitseva palvelinohjelma reagoi näihin käyttäjän lähettämiin pyyntöihin ja vastaa niihin lähettämällä takaisin sopivan vastauksen, esimerkiksi pyydetyn verkkosivun tai muun tiedon. Käytössä oleva yhteysprotokolla määrittelee, miten asiakasohjelma ja palvelinohjelma viestivät keskenään. Verkkosivujen kohdalla käytetään HTTP-protokollaa. (Rantala 2005, 2.)

Verkkosivujen tarkasteluun käytetään asiakasohjelmana tyypillisesti verkkoselainohjelmaa. Käyttäjä lähettää verkkosivun kautta palvelimelle pyynnön esimerkiksi klikkaamalla verkkosivulla olevaa linkkiä ja palvelin vastaa pyyntöön lähettämällä käyttäjän verkkoselainohjelmalle pyydetyn sisällön tarkasteltavaksi. Tämä sisältö voi koostua tekstistä, kuvista, videosta tai muusta informaatiosta. Verkkoselainohjelman vastaanottama tieto muotoillaan käyttäjälle näytettäväksi verkkosivuksi esimerkiksi HTML- ja CSS-kielillä. (W3C 2014.) Tällä hetkellä suosituimpia verkkoselainohjelmia ovat Google Chrome, Microsoft Internet Explorer, Mozilla Firefox ja Apple Safari (Browser Statistics 2014).

Kun verkkosivujen käyttö aikanaan yleistyi, käytännössä kaikki niiden tarkasteluun sopivat päätelaitteet olivat työpöytäkoneita, jotka olivat ominaisuuksiensa kannalta hyvin toistensa kaltaisia. Niitä käytettiin samalla tavoin hiirellä ja näppäimistöllä ja niiden monitorien mittakoot olivat samankaltaiset. Tämän takia verkkosivut suunniteltiin käytettäväksi luonnollisesti juuri näiden päätelaitteiden ominaisuuksia silmällä pitäen. Myöhemmin erilaisten verkkoselaimella varustettujen mobiilipuhelimien kuten vuonna 2006 julkaistun Motorola Z3 -älypuhelimen tullessa markkinoille huomattiin nopeasti että työpöytäkonekäyttöön suunniteltujen verkkosivujen toimivuus mobiililaitteiden kohdalla oli hyvin heikkoa. (Wroblewski 2011, 9-11.) Älypuhelimien verkkoselainten tuki erilaisille tekniikoille kuten JavaScriptille ja CSS:lle, joilla pöytäkonekäyttöön suunnitellut verkkosivut olivat toteutettu, oli varsin puutteellista. Myöskin pöytätietokoneiden näytöille suunniteltujen

verkkosivujen toimivuus pienten mobiilipuhelimien monitoreilla oli heikkoa (Datta 2012). Applen vuonna 2007 julkaisemaa iPhonea voidaan pitää ensimmäisenä mobiililaitteena, jonka verkkoselain tarjosi käyttäjille hyvän käyttökokemuksen verkkosivujen selaamiselle. Tämän myötä verkkosivujen mobiilikäytön suosio lisääntyi ja muut laitevalmistajat alkoivat keskittyä mobiililaitteidensa verkkosivujen selaamisominaisuuksien parantamiseen. (Wroblewski 2011, 10.)

Mobiililaitteiden käyttäjien vaatimukset verkkosivujen toimivuudesta kasvoivat ja verkkosivujen kehittäjien ensimmäinen tapa reagoida tähän oli luoda työpöytäkonekäyttöön suunniteltujen verkkosivujensa rinnalla täysin erilliset mobiilisivut. Ongelmaksi erillisen mobiilisivuston kanssa muodostoi nopeasti se että sellainen sivusto tarjoaa toimivan käyttökokemuksen vain sellaisella päätelaitteella, jolle kyseinen mobiilisivusto oli suunniteltu. Erilaisten päätelaitteiden kirjo alkoi kasvaa nopeasti ja markkinoille tuli nopeasti erilaisen älypuhelimien lisäksi mitä erilaisemmilla ominaisuuksilla varustettuja verkon selaamiseen kykeneviä laitteita kuten normaalia pienikokoisemmat kannettavat tietokoneet, tabletit ja verkkoselaimilla varustetut pelikonsolit. Jokaiselle yksittäiselle laitteelle optimoidun verkkosivun tekeminen osoittautui nopeasti aikaavieväksi ja hankalaksi tehtäväksi. (Knight 2011.) Eräiden laskelmien mukaan erilaisia monitorikokoja on olemassa markkinoilla jo 230 kappaletta (Miksi toteuttaa sivusto responsiivisena? 2013).

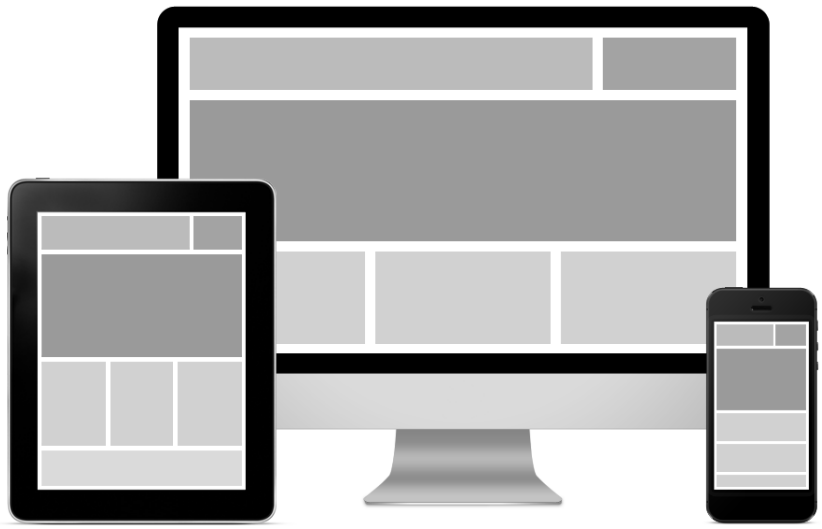
Vuonna 2013 suomalaisista käyttäjistä 32% käytti verkkosivuja usein mobiililaitteilla. Kaksi vuotta aikaisemmin vastaava luku oli 17%. (Suomalaiset verkossa -tutkimus 2013.) Yhdysvaltalaisista käyttäjistä 55% käytti verkkosivuja mobiililaitteella vuonna 2012 (Smith 2012). Vuonna 2011 tehdyn ennustuksen mukaan mobiililaitteiden verkkoselainkäyttö ylittää pöytäkoneiden verkkoselainkäytön vuoden 2014 aikana (Hepburn 2011). Näiden asioiden perusteella nähdään että on tärkeää mikäli yritys haluaa tavoittaa asiakkaansa, on verkkosivujen käyttökokemus oltava kunnossa muillakin päätelaitteilla kuin vain pöytäkoneilla ja mobiilikäyttäjien tarpeet on otettava huomioon.

4 RESPONSIIVINEN SUUNNITTELU

Vastauksena mobiilikäytön yleistyessä ilmenneisiin ongelmakohtiin on esitelty responsiivinen verkkosuunnittelu. Tässä luvussa käsittelen responsiivisuuden ideaa yleisellä tasolla ja pohdin kuinka se vastaa nykyajan verkkoselaamisen tarpeisiin. Tämän lisäksi perehdyn myös Mobile First –ajattelutapaan ja siihen kuinka se liittyy responsiiviseen suunnitteluun.

4.1 Yleistä

Responsiivinen verkkosuunnittelu on tapa totetuttaa verkkosivuja jotka kykenevät automaattisesti mukautumaan päätelaitteen ominaisuuksien mukaisiksi. Tämä voi tarkoittaa esimerkiksi sisällön elementtien näyttämistä pystysuorassa jonossa tai vaakasuorassa rivissä riippuen päätelaitteen näytön leveydestä. Verkkosivun toteuttaja ei aseta tarkkoja rajoja sivun ominaisuuksille, vaan rakentaa sivun siten että käytössä oleva laite määrää sen ominaisuudet. Täten tuloksena saadaan verkkosivu joka tarjoaa hyvän käyttökokemuksen missä tahansa ympäristössä. Kyse ei ole uudesta teknologiasta tai ohjelmointikielestä vaan jo olemassa olevien tekniikoiden oikeanlaisesta hyödyntämisestä. (Yksi sisältö kaikkiin laitteisiin sopivana: Responsiivinen suunnittelu 2012.) Responsiivisen verkkosuunnittelun idea esiteltiin ensimmäisen kerran Ethan Marcotten kirjoittamassa artikkelissa, joka ilmestyi vuonna 2010 A List Apart -verkkojulkaisussa. (Schenker 2013).



KUVA 1. Sivun sisältö mukautuu erilaisille näytöille

Tällä hetkellä erilaiset päätelaitteet voidaan karkeasti jakaa pöytätietokoneiseen, tabletteihin ja älypuhelimiin. Markkinoille tulee kuitenkin koko ajan uusia laitteita ja tällaiset rajat muuttuvat jatkuvasti. Tällä hetkellä eksoottisempia päätelaitteita ovat esimerkiksi Google Glass –silmälasit ja Android Wear –rannetietokoneet, jotka asettavat omat rajoitteensa verkkosivujen tarkastelulle. Tulevaisuudessa markkinoille saattaa myös tulla täysin uudenlaisia laitetyyppejä, joita on hankala ennustaa. Responsiivinen sivusto kykenee mukautumaan parhaassa tapauksessa nykyajan päätelaitteiden lisäksi kaikkiin tulevaisuuden laitteisiin, mikä tekee siitä kestävän ratkaisun, koska silloin ei tarvitse tehdä uutta optimoitua versiota jokaiselle uudelle suositulle päätelaitteelle. (Vainio 2013.)

Useampi erillinen verkkosivusto hankaloittaa verkko-osoitteiden hallintaa. Mikäli yrityksellä on erillinen mobiilisivusto, sille tulee olla oma erillinen verkko-osoite. Tämä voi haitata verkkosivuston sijoittumista hakukoneiden tuloksissa, sillä useamman eri osoitteen omaavan verkkopalvelun sijoittaminen on hankalempaa. Tämän takia hakukoneyritys Google suosittelee responsiivista toteutustapaa useamman optimoidun verkkosivun sijasta (Building Smartphone-Optimized Websites 2013). Responsiivinen sivusto tarjoaa myös paremman käyttökokemuksen tilanteessa jossa käyttäjä tallentaa verkko-osoitteen muistiin pöytäkoneella ja avaa sen myöhemmin esimerkiksi älypuhelimella. Näin käyttäjä voi jatkaa suoraan siitä mihin jäi eikä käyttökokemus kärsi vaikka käytössä onkin erityyppinen laite sillä käyttäjä ei ohjaudu sivuston pöytäkoneversiolle. Vastaavasti käyttäjä voi jakaa linkin älypuhelimella sosiaalisessa mediassa ja linkin pöytäkoneella avaava käyttäjä saa omaan tilanteeseensa sopivan käyttökokemuksen.

Responsiivinen verkkosuunnittelu ei ole jokaisessa tapauksessa kannattavin vaihtoehto. Yrityksellä saattaa olla jo olemassa laajat verkkosivut, joiden muuntaminen responsiiviseksi vaatisi koko sivuston uudelleenrakentamisen. Tämä voi joissakin tilanteissa olla liian kallis ja aikaavievä projekti siitä saatavaan hyötyyn nähden. Joskus yrityksen verkkopalvelun tarjoama ominaisuus toimii paremmin jos se toteutetaan erillisenä mobiilisivuna tai natiivina sovelluksena, kuten on esimerkiksi sosiaalisen median sovelluksen tai verkkokaupan toteuttamisen kohdalla. (Need a Mobile-Friendly Website? 2013.) Verkkopalvelun toteuttamistapa tuleekin valita sen perusteella minkälaista sisältöä aikoo tarjota kävijöille.

Responsiivinen verkkosuunnittelu on otettu hyvin vastaan verkkokehitysyhteisössä. Julkaisut Mashable, Creative Bloq ja Hongkiat ovat nimenneet responsiivisuuden vuoden 2013 merkittävimmäksi ilmiöksi (Cashmore 2012; Rocheleau 2012; Creative Bloq 2013). Forbes nimesi responsiivisuuden vuoden 2014 tärkeimmäksi verkkosuunnittelun ilmiöksi (Forbes 2013).

4.2 Mobile First

Eräs responsiiviseen suunnitteluun läheisesti liittyvä käsite on Mobile First -näkökulma, joka voidaan suomentaa mobiililähtöiseksi suunnitteluksi. Mobiililähtöinen suunnittelu on verkkosivujen kehityksen uranuurtajan Luke Wroblewskin luoma näkemys jonka mukaan verkkosivujen toteuttamisessa lähtökohdaksi tulisi laittaa kaikilta ominaisuuksiltaan vaatimattomimmat päätelaitteet, joka pääsääntöisesti tarkoittaa mobiililaitteita. (Frost 2011.)

Kun verkkosivujen suunnittelussa otettiin ensiaskeleet responsiivisen suunnittelun suuntaan, sivuja toteutettiin yhä ensisijaisesti työpöytäkonekäyttöön. Niissä oli siten hyödynnetty kaikista uusimpia ominaisuuksia ja tehosteita jotka tyypillisesti löytyivät juuri työpöytäkoneiden verkkoselaimilta ja sisältö oli suunniteltu oletusarvoisesti näytettäväksi isoilla monitoreilla. Verkkosivujen käytön yleistyttyä mobiililaitteilla huomattiin että työpöytäkonekäyttöön suunniteltu sisältö toimi yleensä hyvin heikosti mobiilaitteilla, jos lainkaan. Responsiivisuutta lähdettiin toteuttamaan hyödyntämällä niin sanottua siistin latistamisen paradigmaa (graceful degradation). Siinä verkkosivulta piilotetaan tai poistetaan elementtejä siten että se näytti lopulta hyvältä myös vaatimattomammilla ominaisuuksilla varustetun mobiililaitteden monitorilla. Kuva- ja videoelementtejä piilotettiin näkyvistä, isolle monitorille suunniteltu runsas sisältö pakotettiin mahdollisimman pieneen tilaan. Yleensä tästä oli tuloksena heikosti toimiva, puutteellinen ja sekava lopputulos. (Johnson 2013.)

Mobiililähtöisessä suunnittelussa otetaan kaikilla parhaimmilla ominaisuuksilla varustetun työpöytäkoneversion sijasta lähtökohdaksi kaikista vaatimattomin ja todennäköisimmin kaikilla mahdollisilla päätelaitteilla toimiva perussivusto. Tätä sivustoa sitten laajennetaan eteenpäin erilaisilla ominaisuuksilla sen mukaan kun käytössä oleva laite tukee näitä ominaisuuksia. Tätä kutsutaan asteittaiseksi

parantamiseksi (progressive enhancement) joka on siis aikaisemmin mainitun siistin latistamisen vastakohta. (Wikipedia 2013.)

Asteittaisen parantamisen etu siistiin latistamiseen verrattuna responsiivisten sivujen suunnittelussa on se että nämä erilaiset tekniset tavat lisätä ja karsia verkkosivun ominaisuuksia toimivat todennäköisimmin uusimpien ja kehittyneimpien päätelaitteiden verkkoselaimilla jotka tukevat uusimpia verkkosivujen teknologioita. Jos sivuston suunnittelussa lähtökohdaksi on laitettu runsansominaisuuksinen työpöytäkoneversio jonka käyttäjä avaa vanhemmalla mobiililaitteella, ei toimimattomia ominaisuuksia voida karsia pois jos tämä vanhemman mobiililaitteen verkkoselain ei osaa suorittaa ominaisuuksia karsivaa koodia. Sen sijaan jos lähtökohdaksi laitetaan vaatimaton perussivu joka näyttää vain oleellisen sisällön niin teoriassa mikä tahansa nykypäivän ja tulevaisuudenkin laite osaa näyttää käyttäjälle vähintäänkin tämän version, josta laajemmilla ominaisuuksilla varustetut päätelaitteet kykenevät näyttämään asteittain monipuolisemman version kyseisen päätelaitteen ominaisuuksien puitteissa. Tämä tekee mobiililähtöisestä suunnittelusta parhaan tavan toteuttaa mahdollisimman laajalla päätelaiteskaalalla toimivia verkkosivustoja. (Frost 2011.)

5 RESPONSIIVISUUDEN TOTETTAMISTAVAT

Tässä luvussa käyn läpi responsiivisen verkkosivun toteuttamisen keskeisimmät periaatteet. Käyn jokaisessa kohdassa läpi teoreettisen perustelun ja käytännön esimerkin, joilla responsiivisuus voidaan toteuttaa verkkosivulle.

5.1 Joustava asemointi

Perinteisesti verkkosivuja toteuttaessa koko sisällön ja sen sisältämien erilaisten elementtien leveyksien arvoiksi on asetettu pikselimuotoiset luvut, esimerkiksi koko sisältö on määritelty 960 pikseliä leveäksi. Yksi tapa valita nämä leveyden arvot on ottamalla selvää tietokonekäyttäjien yleisimmin käytössä olevien monitorien leveyksistä ja pyrkimällä siten rakentamaan verkkosivu jonka ulkoasun leveys tarjoaa mahdollisimman monelle eri käyttäjälle optimaalisen käyttökokemuksen. (Kyrnin 2012.) Nykypäivänä erilaisten päätelaitteiden kirjo on jo niin suuri että minkäänlaisia yleistyksiä monitorien mitoista ei voi enää tehdä, sen sijaan verkkosivun elementtien leveydet tulisi määritellä pikseliarvojen sijasta suhteellisilla arvoilla eli prosenttilukuina. Tätä tapaa noudattaen verkkosivun sisällön elementtien mittasuhteet ovat aina samat toisiinsa verrattuna oli käytössä olevan päätelaitteen ruudun tai verkkoselaimen koko mikä tahansa. (Pettit 2012.)

Verkkosivun ulkoasun asemoinnista voidaan ensin tehdä suunnitelma pikseliarvoja käyttäen kuvakäsittelyohjelmalla kuten Adobe Photoshopilla. Tässä suunnitelmassa sivuston verkkosivun sisällön leveyden arvo voisi olla esimerkiksi yleisesti käytössä oleva 960 pikseliä. Sen sisällä voisi olla vierekkäin kaksi pystysuoraa elementtiä, joiden leveydet ovat 550 pikseliä ja 350 pikseliä. Prosenttiarvot lasketaan näiden elementtien leveyksille jakamalla leveysarvo kyseisen elementin kontekstin leveydellä. Kontekstilla tarkoitetaan tässä tapauksessa isäntäelementtiä, joka sulkee kyseisen elementin sisäänsä. Näin prosenttiarvot kahdelle edellä mainitulle leveydelle saadaan siis jakamalla ensimmäisen elementin leveys 550 pikseliä ja toisen elementin leveys 350 pikseliä nämä sisäänsä sulkevan isäntäelementin leveydellä joka on tässä tapaksessa 960 pikseliä. Näin kahden vierekkäin olevan elementin leveyden arvoiksi saadaan noin 57,2917% ja 36,4583%. Koko sivuston leveydeksi voidaan antaa esimerkiksi 90%.

Koska sivun koko sisällöllä ei ole isäntäelementtiä, tämä arvo voidaan saavuttaa kokeellisesti katsomalla mikä sopii parhaiten kyseiseen tapaukseen. (Marcotte 2011, 28-32.)

Tuloksena saadaan asettelu, jossa jokaisen elementin koko riippuu sen kontekstin koosta. Kun verkkosivua tarkastellaan erikokoisilla monitoreilla, elementtien koko pienenee ja suurenee samassa suhteessa ja käyttökokemus on sama sekä isokokoisilla että pienikokoisilla päätelaitteilla.

5.2 Joustavat kuvat

Tyypillisesti verkkosivulla olevan kuvan koko määritellään tarkkoina pikseliarvoina. Tällainen tapa toimii jos verkkosivun asettelun mittasuhteet pysyvät aina samoina, mutta erilaisiin tilanteisiin mukautuvia verkkosivuja suunnitellessa kuvien tulisi kyetä suurenemaan ja pienenemään muun sisällön mukana. Tämä voidaan toteuttaa yksinkertaisesti antamalla kuvalle maksimikokoarvoksi sen isäntäelementin koko. (Bradley 2011.)

Tämä saadaan aikaan asettamalla kuvaelementeille voimaan seuraava CSS-tyylisääntö:

```
img {
    max-width: 100%;
}
```

Tällä tavoin kuva pysyy aina samanlaisena suhteessa muuhun sisältöön kun verkkosivun asemoinnin ja isäntäelementin koko muuttuu eikä se voi murtautua ulos omasta asettelukohdastaan ja rikkoa sivun rakennetta.

Samaa keinoa voidaan käyttää myös videoiden ja muiden mediaelementtien kohdalla:

```
embed, object, video {
    max-width: 100%;
}
```

5.3 CSS3 Media Query

CSS-tyylitiedostot sisältävät kolmannesta versiosta lähtien mahdollisuuden ladata tyylisääntöjä erilaisten ehtojen perusteella. Tällaisia ehtoja voi olla esimerkiksi päätelaitteen monitorin leveys tai se missä asennossa päätelaitetta pidetään. Mikäli monitorin leveys on vähemmän kuin jokin pikseliarvo, voidaan ladata käyttöön tyylisäännöt jotka asettavat verkkosivun elementit käytössä olevan monitorin mittasuhteille sopivalla tavalla. Mikäli monitorin leveys on sen sijaan suurempi kuin tämä määritelty pikseliarvo, voidaan käyttöön ladata siihen kyseiseen tilanteeseen tyylisääntökokoelma. Näitä arvoja joissa käyttöön otettava tyylisääntökokoelma vaihtuu, kutsutaan murtumispisteiksi (breakpoint). (Koch 2013.)

Seuraavassa esimerkissä verkkosivun taustaväri muutetaan punaiseksi, mikäli käytössä olevan verkkoselaimen leveys on sekä suurempi kuin 600 pikseliä että pienempi kuin 900 pikseliä.

```
@media all and (min-width: 600px) and (max-width: 900px) {
    body {
        background: #ff0000;
    }
}
```

Käyttöön voidaan ladata myös kokonaan erillisiä CSS-tyylitiedostoja alla olevan esimerkin mukaisesti.

```
<link rel="stylesheet" type="text/css" media="all and (max-width: 900px)"
href="pieni.css" />
```

Tässä esimerkissä ladataan käyttöön tyylisäännöt sisältävä tiedosto pieni.css kun verkkoselaimen leveys on alle 900 pikseliä.

Verkkosivujen kehittäjät ovat tavallisesti määritelleet tällaiset käyttöönotettavia tyylisääntöjä määrittelevät murtumispisteet yleisesti käytössä olevien päätelaitteiden tunnettujen ominaisuuksien perusteella. Näitä ovat esimerkiksi pystysuunnassa olevan iPhone-älypuhelimen leveys joka on 320 pikseliä tai pystysuunnassa olevan iPad-tabletin leveys joka on 768 pikseliä. Tällä tavoin ollaan voitu rakentaa

tyylisääntökokoelmia jotka sopivat täydellisesti kullekin tietylle päätelaitteelle. Responsiivisuuden periaatteiden mukaisesti verkkosivun suunnittelijan ei kuitenkaan tulisi ottaa kantaa siihen minkälaisella laitteella verkkosivua käytetään vaan verkkosivun tulisi mukautua mihin tahansa olemassa olevaan tai tulevaan päätelaitteeseen. Täten murtumispisteet tulisi määritellä sen sijaan sisällön toimivuuden perusteella ottamalla kokeellisesti selvää pisteistä joissa verkkosivun käytettävyys muuttuu epäkelvoksi ja määritellä murtumispiste siihen kohtaan. Tämän voi selvittää toteutettavan verkkosivun kohdalla esimerkiksi pienentämällä selaimen leveyttä hieman kerrallaan kunnes sisältö ei enää mahdu ruudulle ja määritellä siihen kohtaan murtumispiste joka ottaa käyttöön tyylisäännöt jotka muotoilevat sisällön sopimaan paremmin siihen kyseiseen leveyteen. (Frost 2013.)

5.4 Viewport-määrittely

Mobiililaitteiden verkkoselaimet pääsääntöisesti pyrkivät skaalaamaan verkkosivut sellaisiksi että ne mahtuvat kokonaisena päätelaitteen ruudulle. Näin käyttäjä näkee verkkosivustosta yhdellä silmäyksellä kokonaiskuvan ja voi sen jälkeen tarvittaessa skaalata sitä haluamansa kokoiseksi. Tämä on hyödyllinen ominaisuus epäresponsiivisten työpöytäkäyttöön tarkoitettujen verkkosivujen kohdalla kun niitä tarkastellaan pienimonitorisella päätelaitteella, mutta responsiivisuuden perusidea on kuitenkin se että verkkosivuston sisältö mukautuisi käytettävään päätelaitteeseen eikä minkäänlaiselle skaalaukselle tulisi olla tarvetta. Tämän takia kaikenlaisen skaalauksen mahdollisuutta halutaan monesti rajoittaa tai se halutaan estää kokonaan toteuttaessa responsiivisia sivustoja. (Yates 2013.)

Verkkosivun skaalaustapaa voidaan muuttaa ottamalla käyttöön HTML-kielen metatieto-ominaisuus:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Tässä esimerkissä asetetaan verkkoselaimen näkymän leveys samaksi kuin käytössä olevan päätelaitteen leveys. Sen lisäksi verkkosivun oletusskaalaus asetetaan näyttämään skaalaamaton versio verkkosivun sisällöstä. Näin käyttäjälle näytetään skaalaamaton sivusto, jonka sisältö sitten mukautuu aiemmin mainittua joustavuutta ja eri ehdoilla ladattavia tyylitiedostoja hyödyntäen päätelaitteen verkkoselaimeen.

Viewport-metatiedon käyttö ei ole vielä standardoitu tapa, vaikka valtaosa verkkoselainten valmistajista onkin ottanut sen käyttöönsä. Verkkosivutekniikoiden standardeja ajava W3C (World Wide Web Consortium) on aikeissa standardisoida viewport-ominaisuuden muokkaamisen lisäämällä sen osaksi CSS-tyylisääntöjä @viewport-määrittelynä. (Frain 2013.)

Aiemmin esitelty viewportin määrittely voidaan siis ilmaista CSS-tyylisääntönä seuraavanlaisesti:

```
@viewport {
    width: device-width; zoom: 1;
}
```

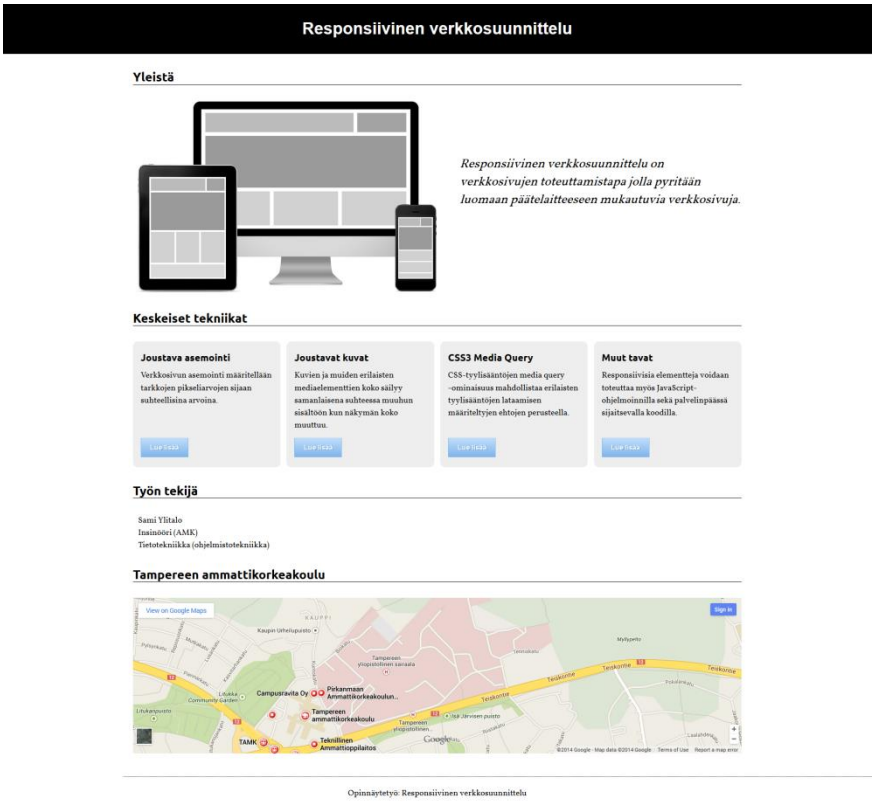
Tällä hetkellä CSS-tyylisääntöjen @viewport-määrittelyä tukevia verkkoselaimia ovat vain Internet Explorer 10 ja Opera. Mikäli haluaa toimivuuden kannalta pelata varman päälle, on tällä hetkellä suositeltavaa käyttää sekä HTML-metatietoa että CSS-tyylitiedoston määrittelyä. (Vieira 2013.)

6 RESPONSIIVISEN VERKKOSIVUN TOTEUTTAMINEN

Tässä luvussa toteutan yksinkertaisen responsiivisen verkkosivun nykyaikaisilla tekniikoilla ja käyn läpi tämän sivun eri osia. Sivu koostuu erilaisista elementeistä jonka sisältö mukautuu käyttäjän verkkoselaimen näkymään aiemmin käsiteltyjä responsiivisia tekniikoita hyödyntäen.

Verkkosivun sisältö koostuu otsikkopalkista ja varsinaisesta sisällöstä. Sekä otsikkopalkki että sivun sisältö on määritelty sopivan levyiseksi ja keskitetty näkymän keskelle. Sivun sisältö koostuu kuvaelementistä, eri järjestykseen aseteltavista tekstielementeistä ja yhteystietoelementistä.

Verkkosivun toimintaa on tarkastelu pöytäkonekäytössä käyttäen Mozilla Firefox –verkkoselaimen versiota 29.0.1. Mobiilikäytössä verkkosivua on testattu Samsung Galaxy S III –älypuhelimella.



KUVA 2. Verkkosivu työpöytäkäytössä

6.1 Viewport-metatiedon muokkaaminen

Ensimmäiseksi verkkosivutiedoston HTML-rakenteen head-osioon lisättiin viewport-metatieto. Tällä estetään verkkosivun skaalaus mobiililaitteiden verkkoselaimilla. Mobiililaitteiden selaimet oletusarvoisesti pienentävät verkkosivun näkymään kokonaisena laitteen ruudulla. Verkkosivun skaalaus vaatii käyttäjää suurentamaan ja pienentämään sivun sisältöä jotta sitä voidaan tarkastella kunnolla, mikä huonontaa käytettävyyttä. Sen sijaan käyttäjälle tulisi näyttää sivun sisältö normaalikokoisena joka sitten mukautetaan päätelaitteen ruudulle media query –säännöillä.

KUVA 3. Verkkosivu älypuhelimella ilman viewport-tiedon määrittelyä

Sivun skaalaus voidaan estää ottamalla käyttöön seuraava viewport-tieto:

```
<meta name="viewport" content="width=device-width, initial-scale=1">
```

Tällä asetetaan näkymän leveydeksi laitteen monitorin leveys ja sivun skaalaukseksi normaali pienentämätön koko. Mikäli sisällön elementit eivät mahdu päätelaitteen ruudulle, ne mukautetaan responsiivisia tekniikoita hyödyntäen eri asetelmaan riippuen käytössä olevan päätelaitteen ominaisuuksista. Tyypillisesti elementit asetellaan pystysuoraan jonoon, jotta niitä voidaan mobiililaitteilla tavalliseen tapaan tarkastella vierittämällä näkymää edestakaisin.

KUVA 4. Verkkosivu älypuhelimella kun viewport-tieto on määritelty

6.2 Sivun leveys

Kun verkkosivua tarkastellaan korkealla vaakatason resoluutiolla, sisällön leveydeksi asetetaan 70% koko näkymän leveydestä. Tämä arvo on saavutettu kokeilemalla eri arvoja ja katsomalla mikä sopii parhaiten ulkoasuun. Sisällön luettavuus on parempi kun sen leveys ei ole liian suuri, joten korkean leveyden näkymillä sisällön leveys halutaan pitää pienempänä. Kun näkymän leveys pienenee, esimerkiksi kun käytössä on

tabletti tai älypuhelin, sivun reunoilla olevaa tyhjä tila kasvaa suureksi ja sisältö ei mahdu enää kunnolla elementtien asetteluun. Tämän takia sisällön leveydeksi muutetaan 90% koko näkymän leveydestä joten sisällölle on enemmän tilaa pienemmällä resoluutiolla. Murtumispisteeksi on valittu tähän kohtaan 1000 pikseliä näkymän leveyttä, jonka alittuessa sisällön leveyden arvoa säädellään media query –säännön perusteella.

```
div.content
{
    width: 70%;
}

@media all and (max-width: 1000px)
{
    div.content
    {
        width: 90%;
    }
}
```

6.3 Kuvatiedoston käyttäminen

Erilaiset kuvatiedostot ja muut mediaelementit ovat oleellinen osa verkkosivujen sisältöä. Kuvatiedostoilla on jokin tietty kokoarvo ja jos elementti jossa kuvatiedosto sijaitsee käy liian pieneksi, kuvaelementti murtautuu ulos asettelustaan. Tämän takia kuvaelementtien halutaan mukautuvan muun sisällön mitta-arvojen mukaan.

KUVA 5. Kuvan koko suuremmalla leveydellä

Kuvaelementtien mukautuvuus voidaan toteuttaa määrittämällä sille maksimikokoarvoksi 100% isäntäelementin kokoarvosta. Näin kuvaelementti ei koskaan kykene murtautumaan sille varatusta tilasta ulos.

```
img
{
    max-width: 100%;
}
```

KUVA 6. Kuvan koko pienemmällä leveydellä

6.4 Elementtien asettelu

Verkkosivulla olevat elementit asetellaan erilaisiin järjestyksiin riippuen päätelaitteen verkkoselaimen leveydestä. Kun selaimen leveys tippuu tarpeeksi pieneksi, vierekkäiset elementit eivät mahdu enää näkymään. Tämä aiheuttaa sisällön ajautumisen kuvaruudun ulkopuolelle ja tällöin käyttäjän tulee vierittää näkymää eri suuntiin, mikä heikentää käytettävyyttä. Elementit tulee sen takia asetella kyseiselle näkymälle paremmin sopivaan järjestykseen hyödyntäen media query –sääntöjä.

Tässä tapauksessa verkkosivulla on muutamia elementtejä jotka voidaan sijoittaa vierekkäin tai pystysuoraan jonoon riippuen verkkoselaimen leveydestä. Tässä kohtaa tarkastellaan neljää tekstisisältöistä elementtiä. Kun näkymän leveys on suurempi kuin 1200 pikseliä, sivulla olevat nämä elementtiä voidaan asetella vaakasuoraan riviin ilman että sisällön luettavuus kärsii. Elementeille annetaan leveyksiksi 24.25% ja elementtien väliin tuleville tyhjille tiloille annetaan leveyksiksi 1%. Mikäli leveys muuttuu hieman eri suuntiin, prosenttiarvoilla määritellyt leveydet mukautuvat kyseiseen tilanteeseen.

KUVA 7. Elementtien asettelu kun leveys on yli 1200 pikseliä

```
div.block-4
{
    width: 24.25%;
    margin-right: 1%;
    float: left;
}
```

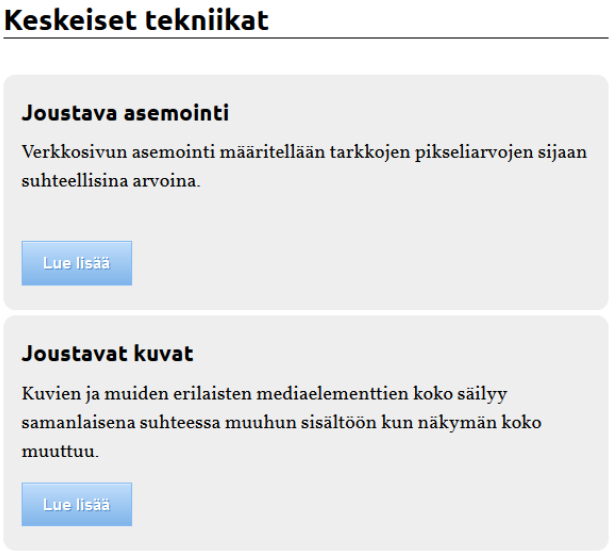
Kun leveys pienenee alle 1200 pikseliin, nämä elementit eivät enää mahdu verkkoselaimen näkymään siten että asettelu säilyisi eheänä. Tämän takia elementit asetellaan näkymään siten että ne menevät kahdelle riville. Näin yksittäisen elementin sisällölle jää enemmän tilaa. Yhden elementin leveyden arvoksi tulee tässä tapauksessa 49.5% isäntäelementin leveydestä. Elementtien leveys muutetaan murtumispisteen alittuessa media query –säännöllä.

```
@media all and ( max-width: 1200px )
{
    div.block-4
    {
        width: 49.5%;
        margin-right: 1%;
        margin-bottom: 1%;
    }
}
```

KUVA 8. Elementtien asettelu kun selaimen leveys on pienempi kuin 1200 pikseliä.

Kun verkkoselaimen leveys on pienempi kuin 600 pikseliä, elementit asetellaan pystysuoraan jonoon antamalla elementin leveydeksi 100%. Näin yksi elementti käyttää koko isäntäelementin leveyden verran tilaa.

```
@media all and ( max-width: 600px )
{
    div.block-4
    {
        width: 100%;
        margin-bottom: 1%;
    }
}
```



KUVA 9. Elementtien asettelu kun leveys on alle 600 pikseliä.

6.5 Kartta ja yhteystiedot

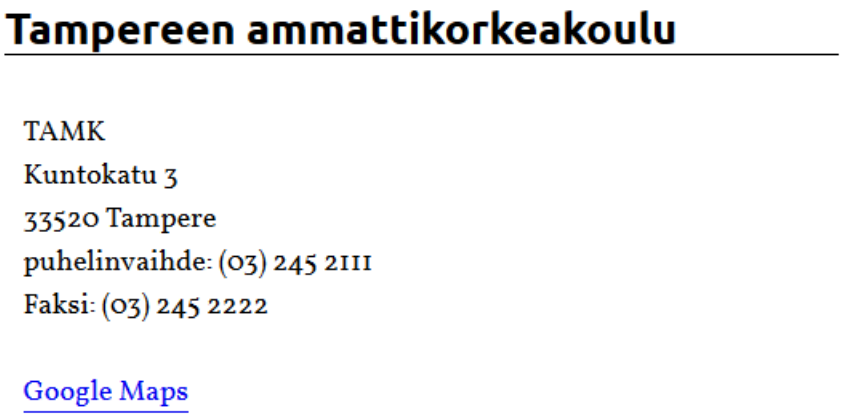
Alimpana sisältönä verkkosivulla on Google Maps –kartta, joka näyttää Tampereen ammattikorkeakoulun sijainnin. Tämä kartta on näkyvissä käyttäjälle, mikäli näkymän vaakatason resoluutio on riittävän suuri, kuten esimerkiksi pöytäkoneiden ja tablettien kohdalla. Karttaelementti mahdollistaa sijainnin tarkastelun ja kartan vierittämisen eri suuntiin.

KUVA 10. Google Maps –kartta

Mikäli näkymän resoluutio on pieni, piilotetaan karttaelementti näkyvistä ja näytetään sen tilalla Tampereen ammattikorkeakoulun yhteystiedot tekstimuodossa. Yhteystietolistauksen lisäksi käyttäjälle annetaan myös linkki Google Maps –karttaan, jonka mobiililaitteet pääsääntöisesti kykenevät avaamaan omassa karttaovelluksessaan.

```
@media all and ( max-width: 600px )
{
    div#googleMapsKartta
    {
        display: none;
    }

    div#yhteystiedotTeksti
    {
        display: inline;
    }
}
```



KUVA 11. Yhteystiedot tekstimuodossa

Kun käytössä on mobiililaite, kartan avaaminen mobiililaitteen omassa sovelluksessa on yleensä toimivampi ratkaisu kuin karttaelementin tarkasteleminen verkkosivuilla, esimerkiksi jos halutaan käyttää navigointilaitetta sijainnin etsimisessä. Karttaelementin lataaminen ja käyttäminen asettaa myös enemmän vaatimuksia mobiililaitteen verkkoyhteydelle, joten asettamalla se avattavaksi linkkiä klikkaamalla se voidaan ladata käyttöön vain tarvittaessa. Mobiililaitteen pieni monitori tekee myös karttaelementin käyttämisestä hankalaa, varsinkin karttaelementin vierttäminen voi helposti sekoittua varsinaisen verkkosivun vierittämiseen.

7 LOPPUPÄÄTELMÄT

Tarkoitukseni oli tässä opinnäytetyössä perehtyä responsiiviseksi suunnitteluksi kutsuttuun verkkosivujen toteuttamistapaan. Kävin opinnäytetyössäni läpi nykypäivän tilanteen asettamien haasteita verkkosivujen toteutukselle ja esittelin responsiivisen ajattelutavan ratkaisuksi näihin kohtiin. Tämän lisäksi perehdyin responsiivisien sivujen keskeisiin teknisiin toteuttamiskeinoihin ja rakensin yksinkertaisen verkkosivun jossa havainnoillistin käytännössä näitä keinoja.

Ottaen huomioon mobiililaitteiden käytön yleistymisen ja erilaisten markkinoilla olevien päätelaitteiden kirjon kasvamisen, responsiivista verkkosuunnittelua voidaan pitää parhaiten toimivana ratkaisuna kun halutaan vastata mahdollisimman suuren asiakasmäärän tarpeisiin. Jokaiselle olemassa olevalle päätelaitemallille optimoitua sivua ei ole enää järkevää tai edes mahdollista tehdä, varsinkin kun uudenlaisia päätelaitteita tulee jatkuvasti lisää myös tulevaisuudessa.

Responsiivinen suunnittelu ei ole välttämättä toimivin ratkaisu jokaisessa tapauksessa. Jos toteutetaan sisältökeskeisen verkkosivun sijasta toimintokeskeistä sivua, esimerkiksi sosiaalisen median sovellusta tai verkkokauppaa, voi mobiililaitteeseen asennettava natiivi sovellus olla paras ratkaisu. Tapaukseen sopivin toteutustapa riippuu siis siitä verkkopalvelun toteuttaja haluaa tarjota asiakkaalle.

Oman näkemykseni mukaan kykenin luomaan työssäni hyvän kokonaiskuvan responsiivisen suunnittelun periaatteista. Aihepiiriä voisi tutkia tulevaisuudessa tarkemmin myös verkkopalveluiden palvelinpäässä sijaitseviin tekniikoihin perehtyen.

LÄHTEET

Bradley, S. 2011. How To Create Flexible Images And Media In CSS Layouts. Julkaistu 23.6.2011. Luettu 9.6.2011.
http://www.vanseodesign.com/css/flexible-images/

Crasman. 2012. Yksi sisältö kaikkiin laitteisiin sopivana: Responsiivinen suunnittelu. Luettu 9.6.2014.
http://www.crasman.fi/fi/uutiskirjeet/asiakaskirje/1-2012/artikkeli/responsiivinen-suunnittelu/

Creative Bloq. 2013. The 10 hottest web design trends of 2013. Julkaistu 23.12.2013. Luettu 9.6.2014.
http://www.creativebloq.com/web-design/2013-trends-121310199

Datta, A. 2014. Be Responsive: A History of Responsive Design. Julkaistu 12.3.2012. Luettu 9.6.2014.
http://shout.setfive.com/2012/03/12/be-responsive-a-history-of-responsive-design/

Frain, B. 2013. Understanding the viewport meta tag, CSS @viewport and making an automatic link to your app. Julkaistu 23.1.2013. Luettu 9.6.2014.
http://benfrain.com/understanding-the-viewport-meta-tag-and-css-viewport/

Frost, B. 2011. Mobile-First Responsive Web Design. Julkaistu 19.6.2011. Luettu 9.6.2014.
http://bradfrostweb.com/blog/web/mobile-first-responsive-web-design/

Frost, B. 2013. 7 Habits of Highly Effective Media Queries. Julkaistu 18.9.2013. Luettu 9.6.2014.
http://bradfrostweb.com/blog/post/7-habits-of-highly-effective-media-queries/

Forbes. 2014. Top Web Design Trends In 2014. Julkaistu 10.2.2014. Luettu 9.6.2014.
http://www.forbes.com/sites/thesba/2014/02/10/top-web-design-trends-in-2014/

Google. 2013. Building Smartphone-Optimized Websites. Päivitetty 11.2.2014. Luettu 9.6.2014.
https://developers.google.com/webmasters/smartphone-sites/details

Hepburn, A. 2011. Infographic: Mobile Statistics, Stats & Facts 2011. Julkaistu 4.4.2011. Luettu 9.6.2014.
http://www.digitalbuzzblog.com/2011-mobile-statistics-stats-facts-marketing-infographic/

Johnson, J. 2013. Mobile First Design: Why It's Great and Why It Sucks. Julkaistu 5.4.2013. Luettu 9.6.2014.
http://designshack.net/articles/css/mobilefirst/

Knight, K. 2011. Responsive Web Design: What It Is and How To Use It. Julkaistu 12.1.2011. Luettu 9.6.2014.
http://www.smashingmagazine.com/2011/01/12/guidelines-for-responsive-web-design/

Koch, P. 2013. Media queries. Luettu 9.6.2014.
http://www.quirksmode.org/css/mediaqueries.html

Koodiviidakko. 2013. Miksi toteuttaa sivusto responsiivisena? Julkaistu 15.8.2013. Luettu 9.6.2014.
http://www.viidakko.fi/ajankohtaista/koodiviidakko-blogi/kirjoitus/miksi-toteuttaa-sivusto-responsiivisena.html

Kyrnin, J. 2012. Web Page Widths. Luettu 9.6.2014.
http://webdesign.about.com/od/webdesign/a/aa080904.htm

Marcotte, E. 2011. Responsive Web Design. 1. painos. New York: A Book Apart.

Mozilla Developer Network. 2014. JavaScript Overview. Päivitetty 7.3.2014. Luettu 9.6.2014.
https://developer.mozilla.org/en-US/docs/Web/JavaScript/Guide/JavaScript_Overview

Pete Cashmore. 2012. Why 2013 Is the Year of Responsive Web Design. Julkaistu 11.12.2012. Luettu 9.6.2014.
http://mashable.com/2012/12/11/responsive-web-design/

Pettit, N. 2012. Beginner's Guide to Responsive Web Design. Julkaistu 8.8.2012. Luettu 9.6.2014.
http://blog.teamtreehouse.com/beginners-guide-to-responsive-web-design

Raggett, D. 1998. Raggett on HTML 4. 1. painos. Boston: Addison Wesley. Luettu 9.6.2014.
http://www.w3.org/People/Raggett/book4/ch02.html

Rantala, A. 2005. Web-ohjelmointi. 1. painos. Jyväskylä: Docendo.

Rocheleau, J. 2012. Web Design: 20 Hottest Trends To Watch Out For In 2013. Luettu 9.6.2014. http://www.hongkiat.com/blog/web-design-trend-2013/

Schenker, M. 2013. Must-know facts about responsive design. Julkaistu 13.6.2013. Luettu 9.6.2014.
http://www.webdesignerdepot.com/2013/06/must-know-facts-about-responsive-design/

Smart Solutions. 2013. Need a Mobile-Friendly Website? Luettu 9.6.2014.
http://www.smartz.com/web-development/mobile/mobile-vs-responsive-design/

Smith, A. 2012. Cell Internet Use 2012. Julkaistu 26.6.2012. Luettu 9.6.2014.
http://www.pewinternet.org/2012/06/26/cell-internet-use-2012/

Vainio, V. Responsiivinen suunnittelu on tullut jäädäkseen. 2013. Julkaistu 22.11.2013. Luettu 9.6.2014.
http://brandstein.fi/?p=31

Vieira, S. 2013. How to build standards-compliant responsive design using @viewport. Julkaistu 13.8.2013. Luettu 9.6.2014.

http://www.webdesignerdepot.com/2013/08/how-to-build-standards-compliant-responsive-design-using-viewport/

W3C. 2014. How does the Internet work. Päivitetty 14.3.2014. Luettu 9.6.2014. http://www.w3.org/wiki/How_does_the_Internet_work

W3schools. 2014. Browser Statistics. Luettu 9.6.2014. http://www.w3schools.com/browsers/browsers_stats.asp

Wikipedia. 2013. Progressive enhancement. Päivitetty 28.10.2013. Luettu 9.6.2014. http://en.wikipedia.org/wiki/Progressive_enhancement

Wikipedia. 2014. Cascading Style Sheets. Päivitetty 1.6.2014. Luettu 9.6.2014. http://fi.wikipedia.org/wiki/CSS

Wikipedia. 2014. HTML. Päivitetty 15.4.2014. Luettu 9.6.2014. http://fi.wikipedia.org/wiki/HTML

Wikipedia. 2014. PHP. Päivitetty 9.6.2014. Luettu 9.6.2014. http://en.wikipedia.org/wiki/PHP

Wium Lie, H. & Bos, B. 1999. Cascading Style Sheets. Designing for the Web. 2. painos. Boston: Addison Wesley. Luettu 9.6.2014. http://www.w3.org/Style/LieBos2e/history/Overview.html

Wroblewski, L. 2011. Mobile First. 1. painos. New York: A Book Apart.

Yates, I. 2013. Quick Tip: Don't Forget the Viewport Meta Tag. Julkaistu 26.6.2013. Luettu 9.6.2014. http://webdesign.tutsplus.com/articles/quick-tip-dont-forget-the-viewport-meta-tag--webdesign-5972

Yle. 2013. Suomalaiset verkossa -tutkimus. Julkaistu 18.6.2013. Luettu 9.6.2014. http://www.slideshare.net/ylefi/suomalaiset-verkoss-2013-esitys-yleisradion-isossa-pajassa-1862013-klo-13

Young, A. 2010. History of JavaScript: Part 1. Luettu 9.6.2014. http://dailyjs.com/2010/05/24/history-of-javascript-1/