LAHDEN AMMATTIKORKEAKOULU
Muotoilu ja viestintä
Viestinnän Koulutusohjelma
Multimediatuotanto
Opinnäytetyö
31.5.2012
Mikki Kuukkanen

LAHDEN AMMATTIKORKEAKOULU
Lahti University of Applied Sciences

Uudet teknologiat ja animaatio webissä

Lahden ammattikorkeakoulu
Viestinnän koulutusohjelma
Kuukkanen, Mikki: Uudet teknologiat ja animaatio webissä

Multimediatuotannon opinnäytetyö, 41 sivua, 0 liitesivua
Kevät 2012

Lahti University of Applied Sciences
Degree Programme in Visual communication
Kuukkanen, Mikki: New technologies and animation in web

Bachelor's Thesis in Multimedia Production, 41 pages, 0 pages of appendices
Spring 2012

TIIVISTELMÄ

Tutkin uusien web-teknologioiden vaikutusta tämän päivän web-suunnitteluun. Tarkoituksena on ottaa selville, kuinka pystyn tämän hetkisessä työssäni käyttämään HTML5:ä, JavaScriptia ja etenkin CSS3:a juuri nyt ja tulevaisuudessa. Kokeilen uusia tekniikoita tekemällä alusta asti uuden portfolio-sivuston itselleni. Arvioin myös ratkaisuja, mitä muilla sivustoilla on tehty uusien tekniikoiden tiimoilta.

Trendit webissä noudattavat nykyään hyvin pitkälle mobiilisovellusten linjaa niin ulkoasultaan, kuin käyttäytymiseltäänkin. Yhtälailla webiin otetaan mallia myös printtimediasta. Uudet teknologiat antavat mahdollisuuden suunnitella sisältöä webiin printtimediasta ja liikegrafiikasta tutuin keinoin. Taitto, typografia ja animointi webissä ovat astuneet suuret harppaukset eteenpäin.

Juuri nyt elämme web tekniikoiden uusien versioiden siirtymävaihetta. Uusien tekniikoiden selaintuen vuoksi meidän pitää jatkuvasti olla tarkkana, mitä tekniikoita pystymme käyttämään ja mitä emme.

Key words: CSS3, web-design, web-technologies

ABSTRACT

I'll study the effect of new the web technologies on today's web-design. The purpose is to find out if I can take an advantage of HTML5, CSS3 and JavaScript on my job in the future. I'll test those new technologies by making a portfolio-site for myself. I'll also evaluate other web-pages that are made with new web-technologies.

These days web-design trends follow mobile-apps in their layout and behavior. Printed magazines have also influenced web design. New technologies give an opportunity to design web in ways, that have been familiar from print-media and motion graphics. Layout, typography and animation have been taking great leap forward.

Today we live a transition period of new versions of the web technologies. Because of a browser support we have to be very careful which technologies we can use and which we can't.

Key words: CSS3, web-design, web-technologies

SISÄLLYS

1 JOHDANTO

Opinnäytetyöni tarkoitus on tutkia uusimpia web-tekniikoita, kuten HTML5:ä, CSS3:a sekä hieman JavaScript-kirjastoja ja sitä, kuinka niitä pystytään hyödyntämään ja kuinka niitä on hyödynnetty jo olemassa olevien web-sivustojen suunnittelussa. Pyrin saamaan selville kuinka uudet tekniikat voivat korvata vanhoja ja helpottaa suunnittelutyötä.

Tavoitteena on rakentaa oma portfolio-sivusto webiin alusta saakka. Tutkin myös samalla muita sivustoja, kuinka niillä on käytetty edellä mainituita tekniikoita ja kuinka ne ovat niistä hyötyneet. Tarkoitus on antaa muista sattuman varaisesti valikoituneista sivustoista hyviä ja huonoja esimerkkejä tekniikoiden käytöstä ja muusta toteutuksesta.

Haasteet tulevat varmasti olemaan oman portfolio-sivuston teknisessä toteutuksessa, sillä en ole datanomi-koulutuksen lisäksi opiskellut juuri lainkaan web-kehitystä tai koodausta. Tällä hetkellä olen töissä IT-yrityksessä, jossa toimin visuaalisena suunnittelijana. Toimenkuvaani kuuluu kuitenkin joissain tapauksissa myös web-kehittäjän tehtäviä, jolloin joudun toimimaan useiden eri frontend web-tekniikoiden parissa. Toivon opinnäytetyön lisäävän omaa asiantuntemustani näihin tekniikoihin ja tuovan hyvän teknisen pohjan myös katsottaessa muutama vuosi eteenpäin.

Jaoin opinnäytetyöni itse portfolion suunnittelun lisäksi kolmeen suurempaan kokonaisuuteen: animaatioon, taittoon ja typografiaan, sillä katson näiden olevan asioita, joihin uusimmat web-tekniikat tuovat frontend puolella eniten apua. Tänään web-suunnittelijalla on vihdoin välineet suunnitella asioita webiin tavoilla, jotka ovat tuttuja printtimedian ja liikegrafiikan puolelta. Taitto ja typografia webissä ovat asioita, jotka ovat olleet aikaisemmin valovuosia jäljessä esimerkiksi aikakauslehtiä.

Juuri nyt uusien tekniikoiden avulla on tehty suuri harppaus eteenpäin ja pystymme myös webissä helposti vaikuttamaan esimerkiksi lehtien taitossa totuttuihin arkipäiväisiin asioihin, kuten kirjasinten valintaan ja palstoihin pohjautuvaan taittoon. Animointi webissä taas on aikaisemmin ollut todella työlästä ja vaatinut loppukäyttäjältä jopa lisäosien asentamista selaimeen, vajaasta käyttökokemuksesta puhumattakaan. Nykyään pääsemme jo pitkälle valitsemalla animointia varten oikeanlaisia JavaScript-plugineita, tai parhaimmillaan voimme käyttää pelkästään uusia CSS3:n mukanaan tuomia määrityksiä. Näin pys-

tymme luomaan käyttäjälle interaktiivisen kokemuksen, joka ei jätä myöskään käytettävyyttä puolitiehen. Kun tästä eteenpäin tutkin animaatiota webissä, en tarkoita tällä animoituja videoita, vaan käyttöliittymän animointia käyttökokemuksen tukena.

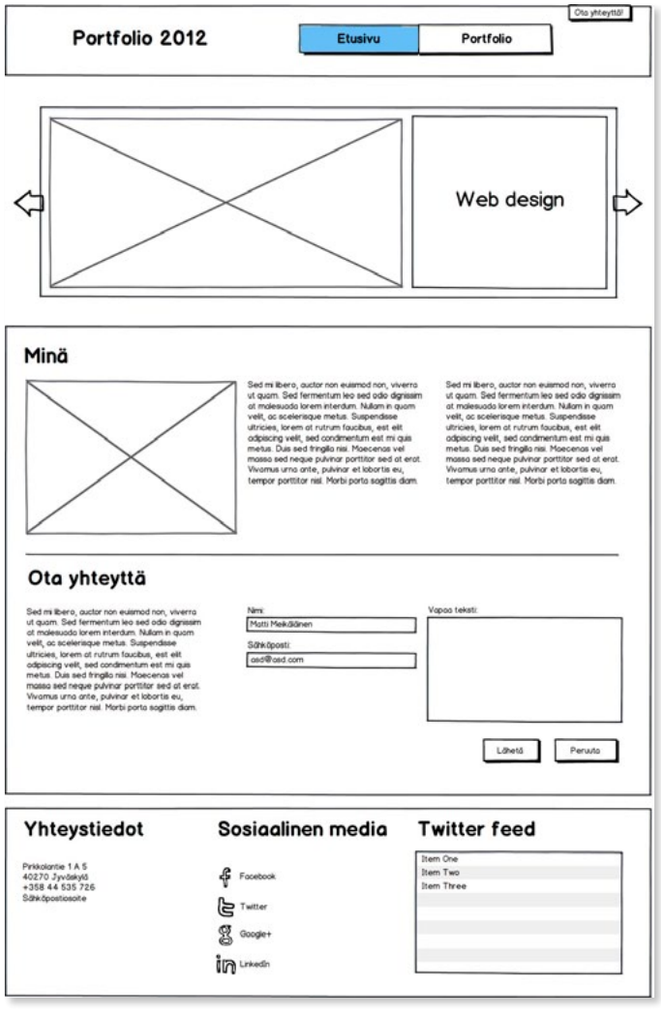
2 PORTFOLION SUUNNITTELU

2.1 Lähtökohdat portfoliolle

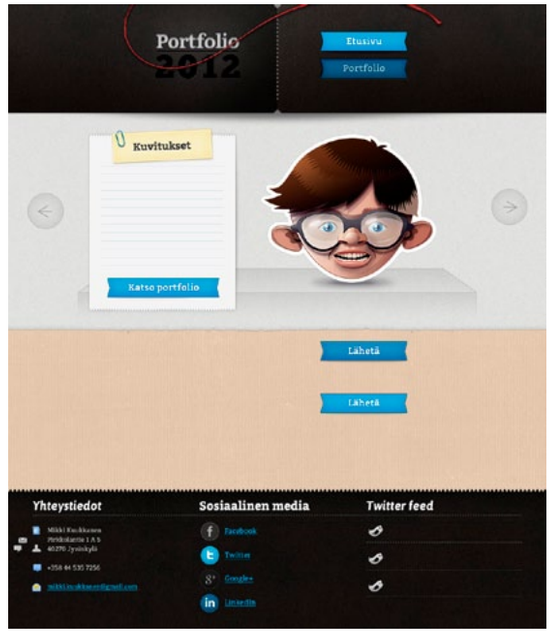
Teen välillä töitä myös omalla toiminimelläni, joten oma portfolio webissä on lähes pakollinen työväline. Ensimmäisen version tekemisestä on jo muutama vuosi aikaa, joten päätin päivittää koko portfolion nykypäivän standardien ja oman osaamiseni mukaiseksi. Webissä tekniikat ja trendit kehittyvät valtavaa vauhtia, muutama vuosi alkaakin jo olla pitkä aika sivuston päivittämiselle. Halusin myös kehittää itseäni, jolloin oma portfolio on hyvä paikka testata erilaisia tekniikoita.

2.2 Etusivun suunnittelu

Aloitin suunnittelun listaamalla portfolioon haluamiani asioita. Pääasia oli, että käyttäjän olisi helppo selata työnäytteitä ja ottaa yhteyttä. Halusin myös sosiaalisen median olevan selkeä osa sivustoa. Huomionherättäjäksi valitsin niin sanotun karusellin, joka pyörittää kuvia animoidusti. Tämän jälkeen aloitin rautalankamallien työstämisen siihen tarkoitetulla ohjelmalla nimeltä Balsamiq Mockups. Pidin koko ajan mielessä kuinka hyödynnän uusia tekniikoita toteutuksessa. Asemointi nykyiseen muotoonsa tapahtui vasta selaimen avulla, kun lopulliset elementtien ja kirjasinten koot alkoivat pikkuhiljaa hahmottua. Etusivulle ylimmäksi sijoitin navigaation, jonka alapuolelle suunnittelin huomionherättäjäksi ja suuremmaksi graafiseksi elementiksi käyttäjän selattavan karusellin, jossa on suoraan nähtävillä eri osaamisalueeni. Seuraavaksi suunnittelin osuuden, jossa kerron hieman itsestäni. Tämän vieressä on yhteydenottolomake, jotta asiakkaan olisi mahdollisimman helppoa lähestyä minua. Alimmaksi footeriin laitoin yhteystietoni, linkkejä sosiaaliseen mediaan ja Twitter-listauksen, joka näyttää neljä uusinta Twitter-päivitystäni. Myös nämä kaikki helpottavat helpottavat yhteydenottoa. (Kuvat 1 ja 2.)



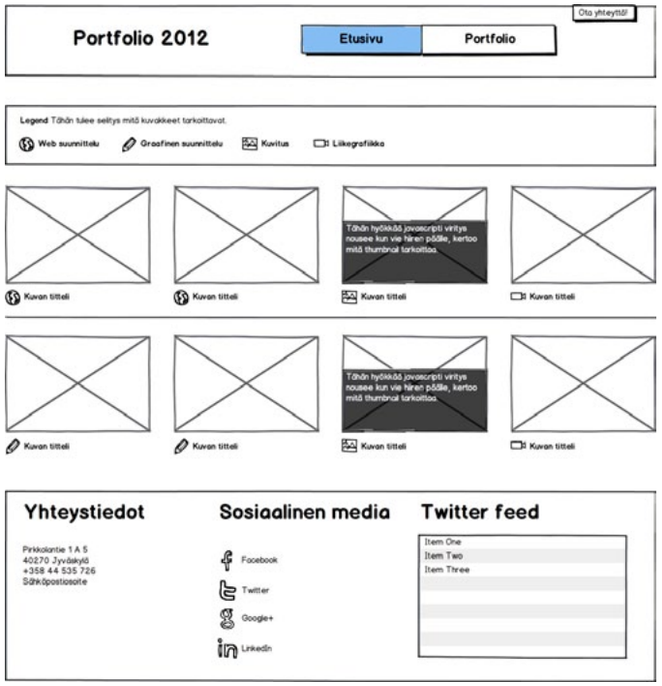
Kuva 1.
Portfolion etusivun rautalankamalli.



Kuva 2.
Etusivun layout.

2.3 Portfolio-sivun suunnittelu

Portfolio-sivulle suunnittelin hieman vaisumman ja mustavalkoisemman värimaailman, sillä halusin, että itse työnäytteet tulevat enemmän esiin. Itse näytteet asetin gridiin, joten eri resoluutioilla niitä mahtuisi eri määrä vierekkäin, tai ne olisivat hieman eri kokoisia. Jaoin myös työnäytteet osaamisalueitteni mukaan web suunnitteluun, graafiseen suunnitteluun, kuvitukseen ja liikegrafiikkaan. Työnäytteen alla on pelkkä nimi, mutta kun työnäytteen kuvan päälle viedään hiiri, sen päälle nousee infoteksti joka kertoo parilla sanalla tarkemmin projektista. (Kuvat 3 ja 4.)

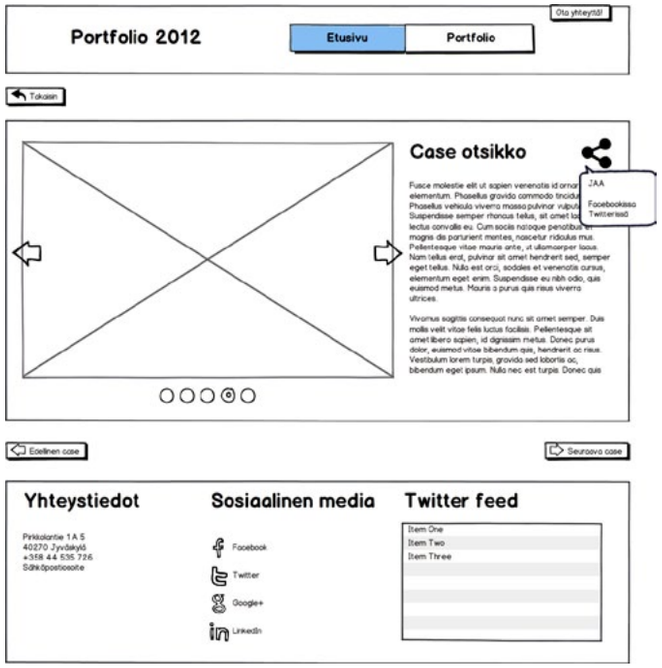


Kuva 3.
Portfolio-sivun rautalankamalli.

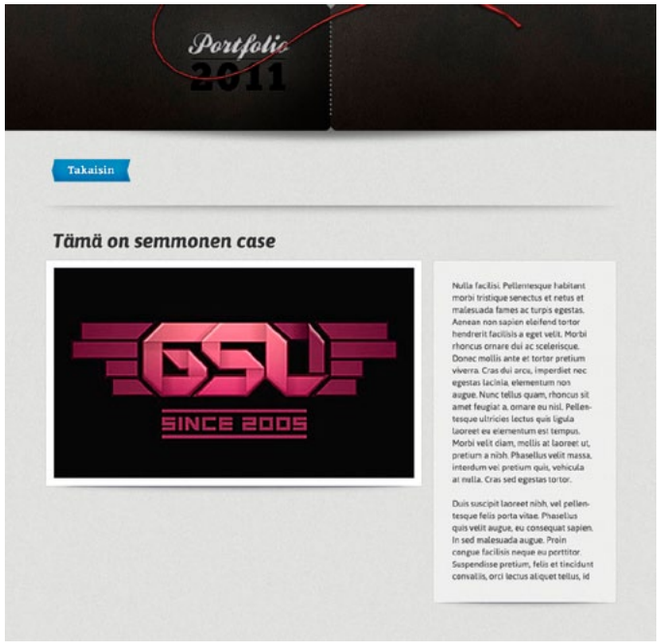
Kuva 4.
Portfolio-sivun layout.

2.4 Työnäyte-sivun suunnittelu

Työnäyte-sivulle suunnittelin myös karusellin, josta käyttäjä pystyy vaihtelemaan projektista otettuja kuvakaappauksia ja muita kuvia. Oikealle puolelle tulee pidempi infoteksti, jossa kerron tarkemmin projektin kulusta. Otsikon viereen tulee myös ikoni, jonka avulla käyttäjä pystyy jakamaan työnäytteen eri sosiaalisissa medioissa. Itse työnäytteen alapuolelle sijoitin navigointielementit, joilla käyttäjä pystyy selaamaan työnäytteitä ilman, että joutuu palaamaan takaisin työnäytelistaukseen. (Kuvat 5 ja 6.)



Kuva 5.
Yksittäisen näytteen rautalankamalli.



Kuva 6.
Yksittäisen näytteet layout.

3 ANIMAATIO WEBISSÄ

3.1 World Wide Webin synty

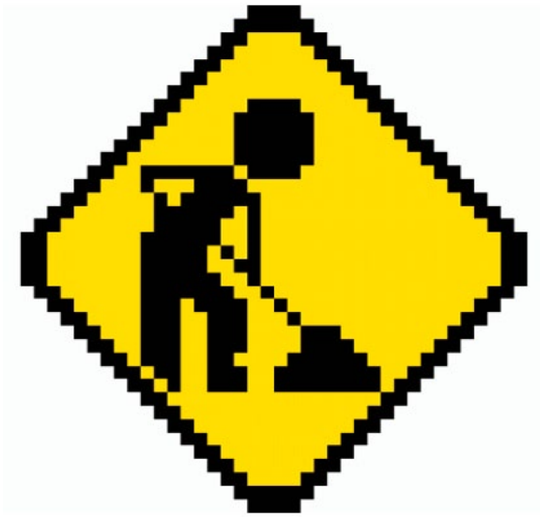
Idean World Wide Webistä voidaan sanoa syntyneen vuonna 1989 CERN-nimisessä fysiikkalaboratoriossa Sveitsin Genevessa. Alunperin oli tarkoitus, että webiä käytettäisiin tieteellisten dokumenttien jakamiseen laboratorion jäsenten kesken. Tällöin ei kuitenkaan ollut edes tarkoituksena luoda maailmanlaajuista verkkoa, vaan pelkästään tiedeyhteisön käyttöön tulevan hyvin paljon suppeamman verkon. Henkilökohtaisten tietokoneiden määrä oli kuitenkin näihin aikoihin alkanut kasvaa kotitalouksissa, joten käytännössä tekniikka WWW-sivujen tutkimiseen oli lähes jokaisen saatavilla. Kasvuun vaikutti suuresti myös Domain Name Service, eli DNS, joka mahdollisti yksinkertaisten sanallisten internetosoitteiden käytön monimutkaisten IP-osoitteiden sijaan. (Wesley, A. 1998)

World Wide Webin Merkkauskieleksi syntyi HyperText Mark-up Language, eli tuttavallisemmin HTML. Aluksi sen sisältö koostui ainoastaan tekstistä ja vasta myöhemmin, tarkemmin sanottuna vuonna 1992 alettiin pohtimaan sitä, kuinka web-sivuille pystyttäisiin lisäämään, kuvia, taulukoita ja muita piirroksia. Seuraavana vuonna IMG-tagi lisättiin Mosaic-nimiseen selaimeen, joka oli sen ajan suosituin. Tällöin IMG-tagin suosio räjähti käsiin. Vuonna 1995 HTML3:n julkaisun myötä HTML:ään alkoi tulla myös muita ulkoasuun vaikuttavia elementtejä, kuten esimerkiksi bgcolor ja class-attribuutit, sekä tuki tyylitiedostoille. (Wesley, A. 1998)

3.2 Animoinnin alkuajat webissä

Alkuaikoina kuvat olivat useimmiten JPG tai GIF-muotoisia. GIF-tiedostomuoto mahdollisti sen, että yksittäiseen kuvatiedostoon pystyttiin lisäämään erillisiä kuvaruutuja peräkkäin (tosin vain 256:lla eri värillä), jolloin syntyi liikkuvaa kuvaa. (Kuva 7.) Näitä animaatioita käytettiin yleensä vilkkuvina ja toistuvina graafisina elementteinä, tai pienimuotoisina videoina. GIF-animaatioita voidaankin varmasti pitää ensimmäisinä varsinaisina web-animaatioina, vaikkakin kokemus oli yleensä käyttäjän näkökulmasta katsottuna lähinnä epileptinen. Valitettavasti tapa jolla GIF-animaatioita käytettiin, ei yleensä tuonut käyttökokemukseen mitään lisää. Lisäksi se kiinnitti käyttäjän huomion täysin vääriin tai vähemmän tärkei-

siin asioihin. Tuohon aikaan suoranaisia videotiedostoja ei World Wide Webissä nähty, sillä ne olivat aivan liian raskaita ja hitaita ladattaviksi sen aikaisilla modeemiyhteyksillä.



Kuva 7.
Esimerkkikuva GIF-tiedostomuodosta.

3.3 Flashin synty ja tuho

Webin interaktiivinen sisältö koki murroksen, kun Macromedian kehittämä työkalu nimeltä Flash alkoi kehittyä ja yleistyä. Tällä kehitysympäristöllä pystyttiin luomaan elävää sisältöä, joka reagoi käyttäjän antamiin syötteisiin. Flashin oman ohjelmointikielen Actionscriptin avulla Flash-sovelluksista pystyttiin tekemään yhä monimutkaisempia; sillä pystyttiin luomaan erilaisia web-sovelluksia, jopa pelejä.

Flash pitää kuitenkin sisällään paljon erilaisia ongelmia jotka tekevät siitä lähes käyttökelvottoman: Ensinnäkin saatavuus mobiililaitteilla on todella kehno. Monet valmistajat, Apple etunenässä, ovat sulkeneet Flashin pois mobiiliselaimistaan. Myös Adobe itse on lopettanut mobiili-Flashin kehittämisen. Toisekseen hakukoneet eivät pysty kunnolla indeksoimaan Flash-sivustoja, jolloin sivustoa on vaikeaa löytää hakukoneiden avulla. Kolmanneksi, käytettävyys on todella heikkoa, sillä et voi käyttää selaimen edes takaisin-nappia. Käyttäjät ovat tottuneet painamaan selaimen takaisin-nappia web-sivuilla navigoidessaan. Kun kokonaan Flashilla tehdylä sivustolla painaa takaisin-nappia, selain menee siis takaisin sivustolle, jolta itse Flash-sivustolle on tultu. Tähän liittyy myös monia muita käytettävyysongelmia, kuten esimerkiksi se, että linkkien värit eivät muutu, sillä Flash ei tunnista milloin tietyillä linkeillä on jo vierailtu. Myöskään selaimeen rakennettu tekstin kasvatus ei toimi Flashin sisällä. Neljänneksi, et voi tehdä kirjanmerkkiä suoraan tietylle sivulle flash-sivustoa, koska kaikki tapahtuu saman URL:n sisällä. Viidenneksi, sivustolla, jolla on käytetty ainoastaan Flashia, koko Flash-tiedosto on ladattava kerralla. Tämä on käytettävyysongelma, koska silloin ei pystytä näkemään osaa sivustosta ennen kuin koko tiedosto on ladattu. Kuudenneksi, selaimeen on asennettava liitännäinen, muuten Flash-sisältöä ei pysty katselemaan. Tämä on lähtökohtaisesti väärin, sillä eihän käyttäjää pitäisi pakottaa asentamaan mitään ylimääräistä koneeseensa, jotta hän pystyisi vierailemaan sivustolla. Seitsemänneksi, Flash-sivustot ovat usein todella raskaita ladattaviksi, varsinkin ilman nopeaa internetyhteyttä. Tälläinen tilanne tulee usein vastaan esimerkiksi tien päällä langattomia internetyhteyksiä käytettäessä. (Nielsen, J. 2000) (Wilson, N. 2002) (Baker, L. 2006) (Clarke, A. 2007, 247)

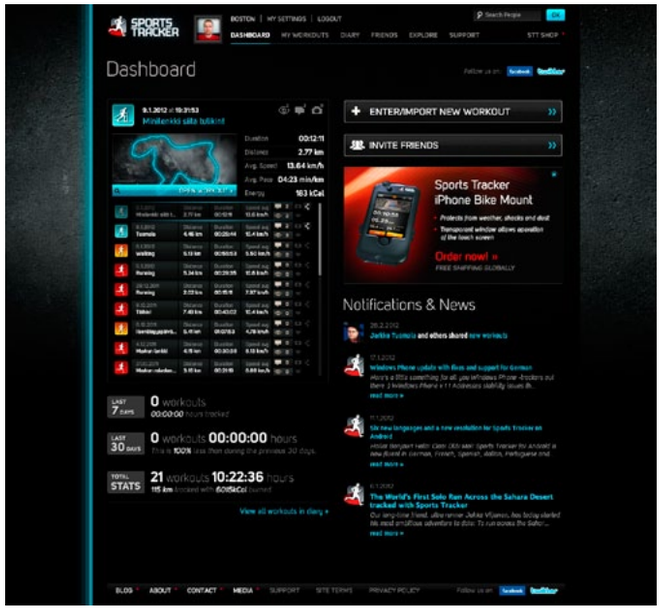
Kuva 8.
Flash intro.

Kuva 8.
Flash intro.

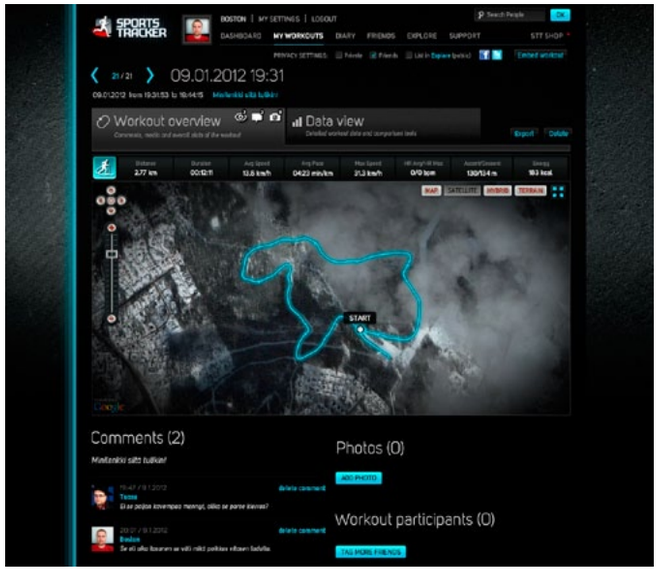
Kuten aikoinaan kävi GIF-animaatioiden, kävi myös sen seuraajan Flashin: käyttäjän huomio kiinnitettiin usein vääriin asioihin. Välillä käyttöliittymistä taas tehtiin lähes mahdottomia käyttää niiden raskauden tai monimutkaisuuden takia. Työkalunakaan Flash ei mielestäni ollut kovinkaan intuitiivinen, vaan pikemminkin melko hankalakäyttöinen ja vaikeasti opittava. Flashilla oli myös vaikeaa tehdä käyttökokemusta parantavia hienovaraisia animaatioita esimerkiksi käyttöliittymän nappeihin, koska tällöin jokainen nappi täytyi olla erillinen Flash-elementti. Tästä taas seuraa paljon turhaa työtä niin Flashin, kuin HTML:nkin puolella. Lisäksi jos selaimeen ei ole asennettu Flash-pluginiä, napit eivät näy lainkaan ja käyttöliittymää on tällöin mahdotonta käyttää. (Kuvat 8 ja 9.)

Nykyään Flashiä käytetään ja tuotetaan yhä vähemmän ja vähemmän. Adobe on jo lopettanut mobiililiitännäisensä kehittämisen ja ilmoittanut, että aikoo jatkossa panostaa HTML5:n hyödyntämiseen ja kehittämiseen. Applen edesmennyt toimitusjohtaja Steve Jobs on pitkään vastustanut Flashia vedoten sen raskauteen ja siihen, ettei Flashia ole suunniteltu tulevaisuuden kosketuskäyttöliittymiin. (Jobs, S. 2010) Apple hylkäsikin Flashin kokonaan iOS-käyttöjärjestelmästään. Luulenkin, että lähivuosina Flash tulee kuolemaan pois, kun Applen lisäksi myös monet muut ovat huomanneet, että HTML5 ja CSS3 voivat täysin korvata Flashin. Viimeisimmät sinnittelijät vetoavat siihen, etteivät pysty katsomaan videokuvaa, joka on upotettu sivustolle Flashin avulla. Teknisesti tämä on tottakai kiistaton tosiasia, mutta keskustellessani Flashin puolestapuhujien kanssa, olen lähes poikkeuksetta pystynyt osoittamaan, että Suomessa lähes kaikki suuremmat Flashia käyttävät palveluntarjoajat ovat erikseen tehneet omat sovelluksensa mobiilialustoille. Tämä taas kertoo siitä, että palveluntarjoajat ovat erittäin tietoisia siitä, mitä käyttäjät haluavat, kuinka he laitteitansa käyttävät ja millaisia ovat tämän päivän trendit. En myöskään usko, että palveluntarjoajien seuraavat sivustosukupolvet tulisivat käyttämään Flashia millään tasolla. Juuri edellä mainittujen asioiden takia en edes harkinnut käyttäväni sivustollani Flashia.

Huonona esimerkkinä Flashin käytöstä voidaan antaa sivusto http://www.serene-naturist.com/, jossa on käytetty sekä gif-animaatioita, että Flashiä kauheimmalla mahdollisella tavalla. Sivusto on niin rauhaton, että on suorastaan ihme, jos käyttäjä pysyy siellä muutamaa sekuntia pidempään. Toisaalta osaavissa käsissä Flashillä pystytään tekemään käytettäviä ja tyylikkäitäkin käyttöliittymiä, kuten esimerkiksi Sports Trackerin tapauksessa (http://www.sports-tracker.com/) on mielestäni onnistuttu tekemään. Animointi on varsin hillittyä ja sopii hyvin teemaan. Muutenkin koko sivusto on yksityiskohdiltaan varsin pitkälle hiottu. Mikään ei varsinaisesti töksähdä silmään ja kaikki animoinnit ovat hiottuja ja pehmeitä. (Kuvat 10 ja 11.)



Kuva 10.
Sports Tracker kuvakaappaus 1.



Kuva 11.
Sports Tracker kuvakaappaus 2.

Vaikkakin edellä mainitun sivuston asiat olisi olleet saavutettavissa helposti myös esimerkiksi JavaScriptin avulla, Flashille on mielestäni vielä toistaiseksi paikkansa webissä. Se toimii vielä kohtuullisen hyvin erilaisten webissä pelattavien pelien moottorina ja mainosbannereissa. Tosin jopa pelkästään HTML5:n ja CSS3:n avulla toteutettuja pelejä on alkanut ilmestyä webiin. Toki suurin osa vaatii vielä toiminnallisuuksiensa puolesta ainakin JavaScriptiä. Tämä yksinkertainen peli tosin on toteutettu pelkästään HTML:n ja CSS:n avulla (vaatii WebKit selaimen): http://cssdeck.com/item/preview/237/css-panic-game.

Tänä päivänä käyttöliittymissä tapahtuva animaatio toteutetaan suurilta osin JavaScriptillä, mutta myös uusin versio Cascading Style Sheeteistä (CSS-tyylitiedostot), eli CSS:n versionumero kolme on alkanut animoinnin osalta nostaa päätään. Web-sivustojen on yleensä voitu ajatella koostuvan kolmesta eri kerroksesta, rakenteesta (HTML), esitystavasta (CSS) ja toiminnallisuudesta (JavaScript). Asia ei enää ole aivan näin yksiselitteinen, sillä CSS3 on astumassa yhä syvemmälle JavaScriptin saappaisiin. CSS on aina pitänyt sisällään myös toiminnallisuuteen kuuluvia elementtejä, kuten esimerkiksi :hover-pseudoluokan (Elementin käyttäytyminen kun hiiren osoitin viedään sen päälle). Raja CSS:n ja JavaScriptin välillä alkoi häilyä, kun CSS3:een sisällytettiin toiminnallisuuteen vaikuttavia elementtejä, etunenässä transformaatiot ja transitiot. (Gasston, P. 2011, 163-164), (Hogan, P. 2010, 216). Portfolio-sivustollani käytin JavaScriptiä karusellin toiminnallisuuden toteuttamisessa. Pohjana on paljon käytetty JavaScript-kirjasto nimeltä jQuery (http://jquery.com/). Tämän lisäksi otin käyttöön vielä animoituja käyttöliittymäelementtejä tarjoavan jQuery Toolsin (http://jquerytools.org/).

CSS:stä näyttääkin tulevan Flashin seuraaja. Tämä on hyvä asia, sillä pääsemme eroon aikaisemmin mainituista Flashin käytettävyysongelmista. Takaisinnappi, linkkien värit ja tekstin kasvatus toimivat. Selaimeen ei tarvitse asentaa mitään ylimääräistä, koska CSS-tuki on niissä luonnollisesti sisäänrakennettuna. Mobiiliselaimet, URL:t tai hakukoneiden indeksointikaan ei ole ongelmana CSS:n käytössä.

3.4 Mitä CSS on?

Cascading Style Sheets, eli CSS, on webissä käytettävä tyylitiedostomuoto. Siinä missä HTML:n tarkoitus on merkata sisältöä, CSS pitää huolen siitä, miltä sisältö käyttäjälle näyttää. CSS ei oikeastaan ollut syntyessäänkään uusi keksintö. Rakenteen ja ulkoasun erottaminen toisistaan oli tarkoituksena jo kun HTML sai alkunsa. CSS esiteltiin ensimmäisen kerran web-konferenssissa Chigagossa vuonna

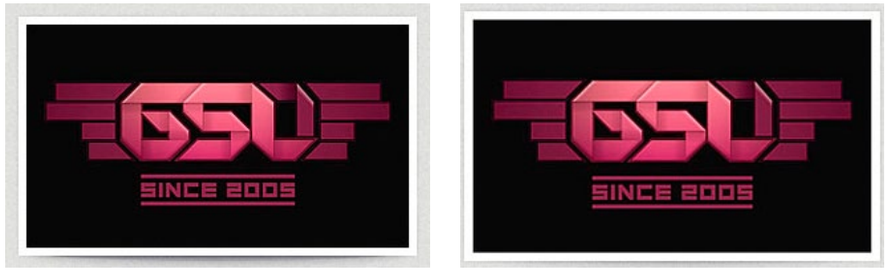
1994. Vuonna 1995 Microsoft päätti ottaa CSS:n osaksi omaa selaintaan. Hieman myöhemmin myös Netscape saatiin samaan kelkkaan, kun W3C (World Wide Web Consortium) antoi virallisen suosituksensa CSS:n käytöstä. Tästä alkoi CSS:n voittokulku HTML:n rinnalla. (W3C.org. 2012) Ajan saatossa CSS:ään on lisätty ominaisuuksia pikkuhiljaa. Uusimman CSS3-version ominaisuuksien avulla pystymme animoimaan HTML-elementtejä. CSS3:a voidaan käyttää hyvin monella eri tapaa käyttöliittymän animoinnissa. Hyvä esimerkki CSS3:n monimuotoisuudesta voidaan nähdä tässä OS X Lionia jäljittelevässä demossa: http://www.alessioatzeni.com/mac-osx-lion-css3/.

3.5 CSS3 animointi

3.5.1 CSS3 transformaatiot

CSS3:n transformaatioiden avulla HTML-elementtejä pystytään siirtämään, pyörittämään, skaalaamaan ja vääntämään. (Gasston, P. 2011, 147-160), (Goldstein, A. Lazaris, L. & Weyl, E. 2011, 175-196) Portfoliossani käytän transformaatiota suurentamaan hieman portfolion minikuvia, kun hiiren osoitin viedään niiden päälle. Tämä tapahtuu seuraavien tyylimääritysten avulla alla kuvatulla tavalla. (Kuva 12.)

```
.mosaic-block:hover {
    -moz-transform: scale(1.02);
    -webkit-transform: scale(1.02);
    -o-transform: scale(1.02);
    -ms-transform: scale(1.02);
    transform: scale(1.02);
}
```



Kuva 12.
Portfolio-näytteen hienovarainen suurennus.

3.5.2 *CSS3 transitiot*

CSS3 transitiot tarkoittavat CSS-tyylimääritysten vaihtumista tietyssä ajassa, saaden näin aikaan animaation. Transitiot on varsin helppo omaksua, tyylitiedostoon kirjoitetaan ensin transition kohde ja seuraavaksi aika, jonka halutaan transitioon kuluvan. Transition nopeudelle voidaan myös määrittää funktio oletuksena olevan lineaarisen arvon sijaan, tästä käytetään animointiohjelmista tuttua termiä ”easing”. (Gasston, P. 2011, 163-169), (Goldstein, A. Lazaris, L. & Weyl, E. 2011, 175-196) Portfoliossani käytän transitioita liu’uttamaan linkin värin toiseen väriin. (Kuva 13.) Linkin värit vaihdoin seuraavasti:

```
a {
    text-decoration: none;
    -webkit-transition:color 0.5s ease-in-out;
    -moz-transition:color 0.5s ease-in-out;
    -o-transition:color 0.5s ease-in-out;
    -ms-transition:color 0.5s ease-in-out;
    transition:color 0.5s ease-in-out;
}

footer a:hover {
    color: #6ae5ff;
}
```

Tällöin kun annamme linkin hover-tilalle eri värin, vaihdos tapahtuu liukumana puolen sekunnin aikana:

Kuva 13.
Linkin siirtymä normaalista hover-tilaan.

Saadakseni myös nappeihin samantyyppisen yksinkertaisen siirtymän, jouduin hieman kiertämään ongelmaa, sillä vielä tällä hetkellä testieni perusteella ainakaan WebKit-selaimet Safari ja Chrome eivät tue pseudo-elementtien transitioita. (Kuva 14.) (Fade Image Into Another (within a Sprite), 2011) HTML-merkkauksseltaan napit ovat seuraavanlaiset:

```
<nav>
    <ul>
        <li class="btn_home">
            <a href="index.html">
                koti
                <span></span>
            </a>
        </li>
    </ul>
</nav>
```

Yllä mainittuun home-nappiin vaikuttaa seuraavat CSS-tyylimääritykset:

```
nav ul li a {
    display: block;
    position: relative;
    height: 43px;
    width: 170px;
    text-indent: -9999px;
}

.btn_home {
    background: url("../img/btn_home.png")
    no-repeat scroll 0 0 transparent;
}

.btn_home span {
    position: absolute;
    top: 0; left: 0; bottom: 0; right: 0;
    opacity: 0;
    -webkit-transition: opacity 0.5s;
    -moz-transition: opacity 0.5s;
    -o-transition: opacity 0.5s;
    transition: opacity 0.5s;
}

.btn_home span:hover {
    opacity: 1;
}
```

Siirtymä siis näyttää kokonaisuudessaan seuraavalta:

Kuva 14.
Napin hover-efekti.

3.5.3 *CSS3 animoinnin tulevaisuus*

Pelkästään transformaatioilla ja transitioilla ei päästä animoinnissa vielä kovinkaan syvälle. WebKit selaimet tukevat kuitenkin jo nyt hyvin animaatio ja 3D transformaatio -moduuleita, joilla pystytään toteuttamaan todella monimutkaisia animointeja CSS:n avulla. Animaatio moduuli mahdollistaa key frame-animoimisen, joka on tavallaan kaiken animaation ydin. Keyframe animointi tarkoittaa sitä, että animaation aikajanalle asetetaan pisteitä, joiden välillä tietty muutos tapahtuu. Eli esimerkiksi jos haluamme luoda animaation, jossa objekti muuttuu läpinäkyvästä näkyväksi ja sen jälkeen puolittain näkyväksi, voimme kirjoittaa sen CSS-määrityksiin seuraavasti:

```
@keyframes opacity_animation {
    0%  {opacity:0;}
    50%     {opacity:1;}
    100%    {opacity:0.5;}
}
```

Tämän jälkeen meidän pitää antaa animaatio jollekin elementille, eli esimerkiksi ID:n perusteella näin:

```
#box {
    animation: opacity_animation 5s infinite;
}
```

3D transitiot taas antavat mahdollisuuden muokata objekteja kolmiulotteisesti. Ne vaativat koneelta paljon tehoja ja 3D transitioiden käyttäminen vaatiikin selaimelta tuen näytönohjaimelle. Nämä ovat kuitenkin vielä melko kokeellisella pohjalla, joten 3D transitioiden käyttäminen tuotantokäytössä ei vielä toistaiseksi ole järkevää. (Gasston, P. 2011, 172-180)

Tällä hetkellä kaikki uusimmat selaimet tukevat jo CSS3:a melko pitkälle. Selaimissa on kuitenkin eroja, esimerkiksi WebKit-selain Chrome on juuri nyt hieman edellä Firefoxia, Safaria, Operaa ja Internet Exploreria. (CSS3.info/Chris, 2012) Selaimensa CSS3-tuen voi käydä testaamassa osoitteessa: http://css3test.com/.

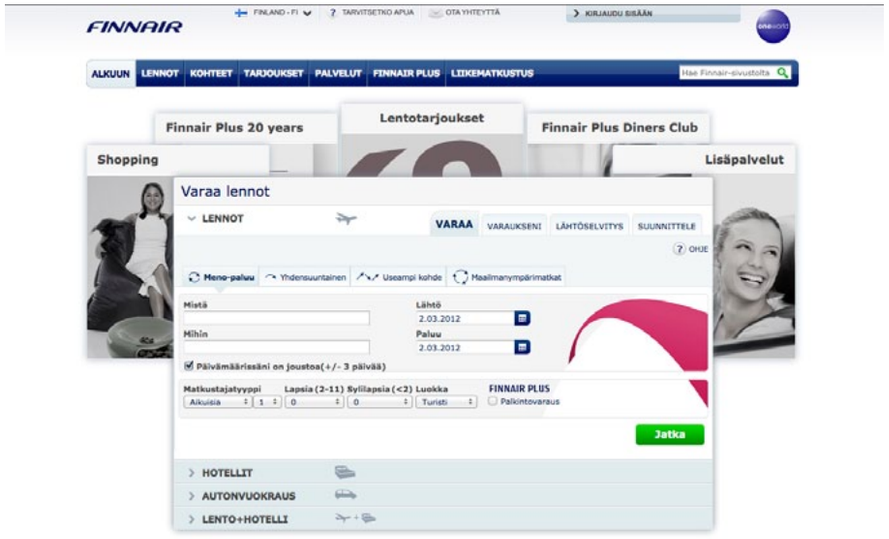
3.6 WYSIWYG-editorit

WYSIWYG-editorien (tulee sanoista What You See Is What You Get) tie on pitkä. Ensimmäinen oikea WYSIWYG-tekstieditori Bravo kehitettiin jo vuonna 1974 Charles Simonyin ja Butler Lampsonin toimesta. Tämä johti myöhemmin muun muassa Microsoft Wordin kehittämiseen. Ensimmäiset editorit olivatkin tekstinkäsittelyohjelmia. Käytännössä nämä editorit siis auttavat sinua tuottamaan koodia ilman, että sinun täytyy nähdä sitä riviäkään. (Markoff, J. 2007)

Markkinoille on jo tullut CSS3:a ja HTML5:ä hyödyntäviä WYSIWYG-työkaluja kuten Adobe Edge (http://labs.adobe.com/technologies/edge/), Sencha Animator (http://www.sencha.com/products/animator) ja Tumult Hype (http://tumultco.com/hype/). Täytyykin kysyä, että nyt kun nämä uusia web tekniikoita hyödyntävät WYSIWYG-editorit alkavat yleistyä, tullaanko jälleen näkemään Flashille ominaisia vaikeakäyttöisiä ja surkuhupaisan sirkusmaisia käyttöliittymiä (http://iccmworldwide.org/, http://www.evangelcathedral.net/)? Tämä ”vallankumoushan” on jo tavallaan kertaalleen koettu. Tuovatko uudet tekniikat sisältöön ja käyttökokemukseen mitään uutta, vai mennäänkö tässä tavallaan takapakkia jälleen kerran? Loppukäyttäjällehän on täysin yhdentekevää mitä ”konepellin alta löytyy”, kunhan käyttöliittymässä navigointi on nopeaa ja sulavaa, sillä ihmiset eivät enää halua lisää ominaisuuksia, vaan toimivan konseptin. Itse uskon, että olemme jo oppineet virheistämme ja tulevaisuudessa osaamme käyttää uusia tekniikoita järkevämmin ja selkeämmin. Ottamalla oppia mobiilikäyttöliittymistä ja sovelluksista ja tuomalla niitä sopivalla tavalla webbiin emme voi mennä kovin väärään suuntaan.

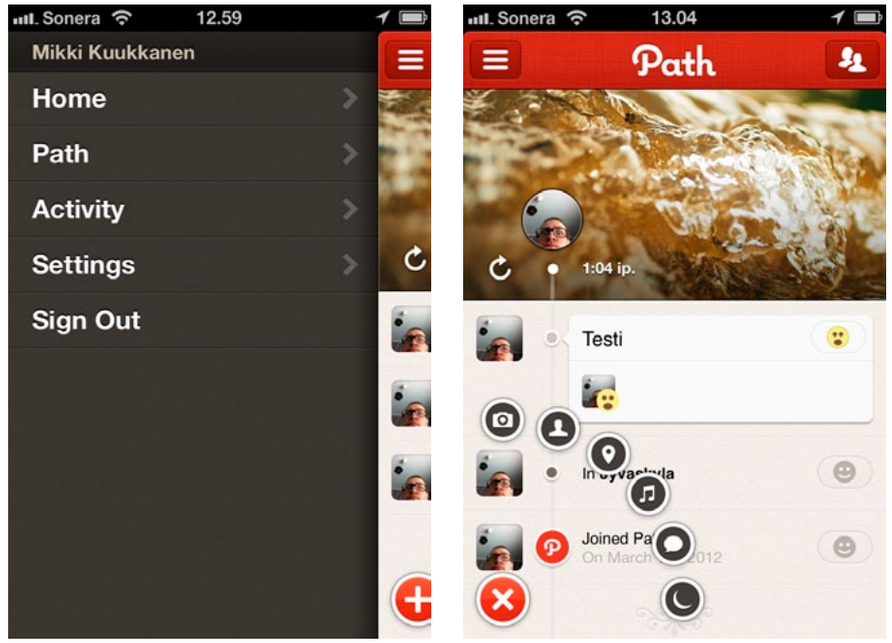
3.7 Animaatio web-sivustoilla, mobiili- ja työpöytäkäyttöliittymissä

Tällä hetkellä web-sivustojen animointi tapahtuu pääasiassa JavaScriptillä. JavaScriptille on olemassa erilaisia kirjastoja, joita voi käyttää hyödykseen animointia tehdessä. Myös näillä tekniikoilla voidaan astua harhaan, kuten mielestäni Finnair teki julkaistessaan uuden sivustonsa. Vaikeakäyttöinen ja sekava karuselli, jossa sisältö on ahdettu pieneen tilaan, ei tuo käyttökokemukseen mitään lisäarvoa. (Kuva 15.)



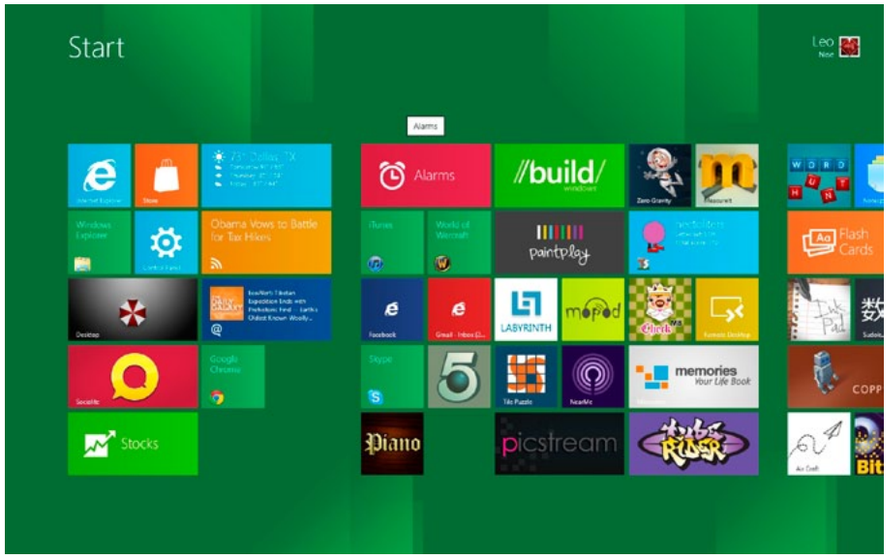
Kuva 15.
Kuvakaappaus Finnairin käyttöliittymästä.

Käyttöliittymän animointi on tärkeää, koska sen avulla pystytään säilyttämään jatkuvuus ja havainnoimaan käyttäjälle missä kohdassa käyttöliittymää hän kullakin hetkellä on. Hyvä käyttöliittymäanimointi on sellaista, mihin ei juurikaan kiinnitä huomiota, mutta se samalla tukee käyttökokemustasi. (Cossey, M. 2012) Animaationa web-sivuilla kannattaa esittää sellaisia asioita, jotka vaikuttavat käytettävyyteen. Tarkoitan tällä erilaisia hienovaraisia siirtymiä käyttäjän navigoidessa sivuilla. Mielestäni monet mobiilisovellukset ovat edelläkävijöitä käyttöliittymän animoinnissa, kun taas tuntuu siltä, että web sivustot ja työpöytäkäyttöliittymät ovat jääneet hieman polkemaan paikalleen animoinnin suhteen. Flash toi aikanaan sivustoille interaktiivisuutta, mutta tekniikkana se oli huono ja sitä käytettiin väärin tarkoitusperin. Web-sivustoilla navigoiminen on usein töksähtelevää, eikä lainkaan sulavaa jatkuvaa liikettä kuten sen pitäisi olla. Linkkiä painaessa uusi sivu hyppää yhtäkkiä silmille ilman minkäänlaista transitiota. Näin ei synny minkäänlaista kuvaa siitä, edettiinkö käyttöliittymässä eteenpäin vai taaksepäin. Toisin on esimerkiksi iOS-mobiilikäyttöliittymässä. Jo ohjelmaa avatessa nähdään kuinka ohjelma kirjaimellisesti aukeaa ruudulle. Path-ohjelmassa taas voidaan hyvin nähdä monin eri tavoin kuinka näkymävalinnassa ikään kuin vaihdetaan sivua. Eli sisältö ei ainoastaan vaihdu, vaan käyttäjä näkee myös siihen selvän viittauksen. Tällöin käyttäjä pysyy hyvin perässä siitä missä kohdassa käyttöliittymää hän kullakin hetkellä on. Myös eteen ja taaksepäin mentäessä animaatio tukee käyttökokemusta. Path on muutenkin panostanut käyttöliittymänsä animoinnin yksityiskohtiin todella paljon. (Kuva 16.)



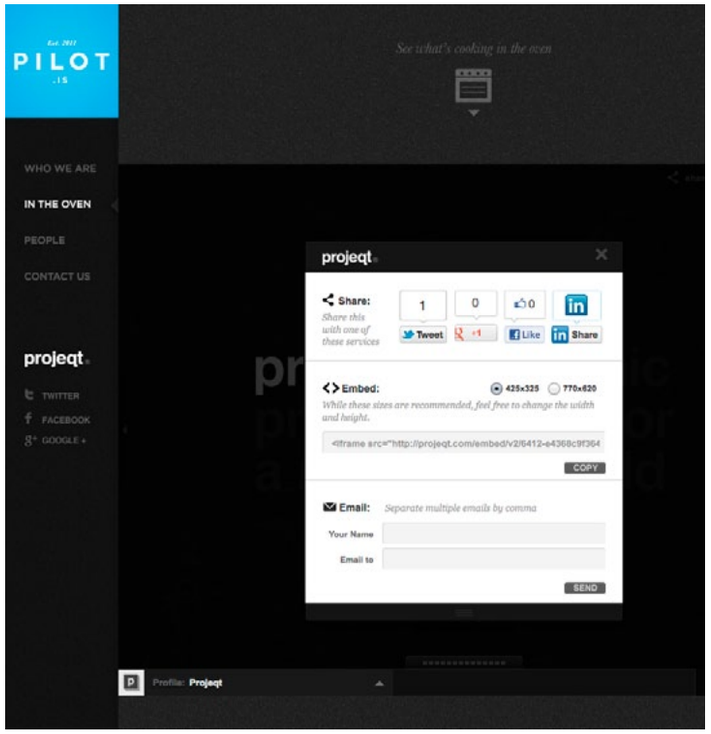
Kuva 16.
Kuvakaappauksia Path-sovelluksen käyttöliittymästä.

Myös työpöytäkäyttöliittymät ovat lähteneet mobiilisovellusten suuntaan, luultavasti juuri siitä syystä, että kehittäjät uskovat nyt kosketusnäyttöihin myös työpöytäkäytössä. Mielenkiintoisin uusista ratkaisuista on mielestäni Windows 8. (Kuva 17.) Se on suunniteltu käytettäväksi etenkin kosketusnäytöillä. Toki myös hiiri ja näppäimistö on otettu huomioon. Sama versio Windows 8:sta toimiikin sekä pöytäkoneissa, että mobiililaitteissa. Tämä uusi versio sekä tuntuu, että näyttää mobiilikäyttöliittymältä. Myös animointi on hyvin tuttua mobiilisovelluksista ja käyttöliittymistä.

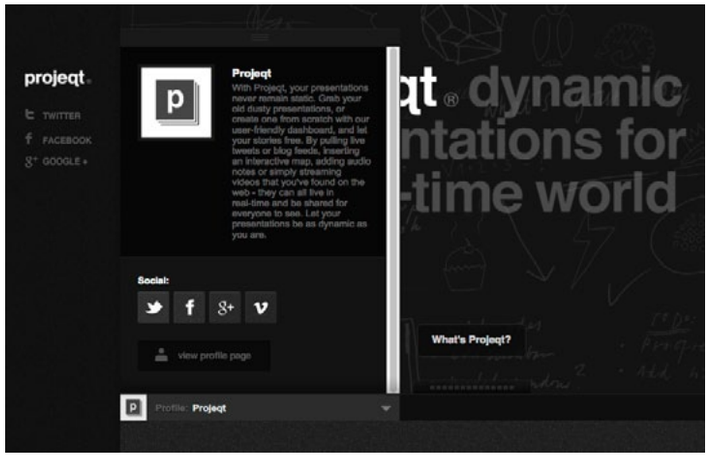


Kuva 17.
Kuvakaappaus Windows 8:n työpöytänäkymästä.

Websivustotkin tulevat hitaasti perässä. Siirtymiin ja dynaamisuuteen on taas alettu kiinnittää yhä enemmän huomiota. Hyviä esimerkkejä tästä ovat sivustot http://pilot.is ja http://nclud.com, joissa käyttöliittymän animointi on kohtuullisen luontevaa ja harkittua. (Kuvat 18 ja 19.)



Kuva 18.
Pilot.is käyttöliittymän kuvakaappaus 1.



Kuva 19.
Pilot.is käyttöliittymän kuvakaappaus 2.

Haasteena käyttöliittymän animoinnille webissä on eri selainten suuri lukumäärä. Jotkin tukevat paremmin uusia ominaisuuksia kuin toiset. Mielestäni ajattelumallin olisikin muututtava juuri tässä asiassa. Vaatimuksena ei pidäkään olla se, että sivustot näyttävät tismalleen samalta joka selaimella vaan se, että niiden täytyy toimia samalla tavalla.

3.8 Muutos ajattelumallissa

Selaimet ovat kehittyneet viime vuosina paitsi nopeammaksi, ne myös tukevat tänä päivänä suurta määrää erilaisia ulkonäköön vaikuttavia ominaisuuksia. Näihin CSS3 ominaisuuksiin kuuluvat esimerkiksi elementtien pyöristetyt kulmat, tekstin ja elementtien varjot, liukuvärit, animaatiot, useat taustakuvat yhdellä elementillä, useat palstat yhdellä elementillä, RGBA-värit (Red, Green, Blue, Alpha) ja erilaiset laatikkomallit. Mikäli käytämme näitä määrityksiä vanhemmilla selaimilla, ne eivät toimi, mutta eivät myöskään estä käyttöä millään tavalla.

HTML5:n suhteen meidän pitää olla hieman enemmän varpaillamme, sillä se pitää sisällään elementtejä, kuten esimerkiksi header, footer, article, section ja nav -elementit, joita vanhat selaimet eivät tunnista. Meidän täytyy tällöin esitellä nämä elementit erikseen esimerkiksi itse portfoliossa käyttämäni HTML5 JavaScript shivin (http://code.google.com/p/html5shiv/, https://github.com/aFarkas/html5shiv) avulla. Se on pätkä koodia, joka auttaa vanhoja selaimia tunnistamaan nämä uudet elementit.

Nykyään yhä useammat tahot ovat alkaneet mieltyä ns. progressive enhancement ajattelumalliin, joka tarkoittaa yksinkertaistettuna sitä, että vanhat selaimet saavatkin näyttää sisällön hieman ”rumemmin”, kuin miten uudet selaimet sisällön näyttävät. Tällöin pitää kuitenkin pitää huolta siitä, että sivusto on edelleen yhtä käytettävä mitä moderneillakin selaimilla. (Castro, E. Hyslop, B. 2012, 376) Suuremmatkin yritykset ovat jo alkaneet myydä projekteja asiakkaille niin, että vanhoja selaimia ei tueta ulkonäköseikoissa aivan loppuun saakka. Tämä säästää kehittäjiltä valtavan määrän työtunteja ja tätä kautta asiakkaan rahoja.

Internetistä löytyy kuitenkin myös paljon erilaisia koodikirjastoja, joiden avulla vanhojen selainten puutteita pystytään paikkailemaan. Esimerkiksi CSS3Pie (http://css3pie.com/) tuo monet CSS3:n avut myös vanhemmille Internet Explorer selaimille. Toinen paljon suosiota saavuttanut JavaScript-kirjasto on Modernizr (http://www.modernizr.com/), joka lähtee hieman erilaisesta näkökulmasta, eikä edes pyri auttamaan vanhoja selaimia tuottamaan CSS3:n efektejä. Sen sijaan se tunnistaa selaimen, ja antaa luokan sen mukaan, tukeeko se tiettyjä ominaisuuksia. Tällöin pystymme helposti paikkaamaan vanhojen selainten puutteita antamalla niille esimerkiksi CSS2:n määrityksiä tähän tapaan:

```
.no-textshadow {
    color: black;
}
```

4 TAITTO WEBISSÄ

4.1 Palstat ja gridi

Taittaminen webissä on mielestäni yhä tänäkin päivänä lapsenkengissään. Lehtimäinen taitto on lisääntynyt huomattavasti lähivuosina, mutta silti ylipäätään käsite taittamisesta www-sivustoille on yhä valitettavan monille web-kehittäjille vieras. Hieman yleistäen voidaan ajatella, että aikaisemmin palstat käsitettiin parina rinnakkaisena osiona, jolloin www-sivustolla se tarkoitti usein sisältöosiota ja sub-navigaatiota. Sisältöosio on yleensä sisältänyt yhtenäisen tekstimassan, jonka kappaleiden väliin on yritetty mahduttaa kuvia. Palstoja ei siis aikaisemmin käytetty juurikaan muuhun, kun erottamaan sivustojen eri osiot toisistaan.

Meni varsin pitkään, ennen kuin aloimme käyttämään perinteisen median graafisesta suunnittelusta tuttua ruudukkoa, eli gridiä auttamaan web sivujen taitossa. Palstojen ja gridin käyttö www-sivuilla tuo mukanaan selviä etuja: Sisältö on selkeämpää, luettavampaa, jatkuvampaa ja tila voidaan käyttää tehokkaammin hyödyksi. Myös sisällön organisointi on gridin avulla huomattavasti helpompaa. (Samara, T. 2002) (Clarke, A. 2007, 198).

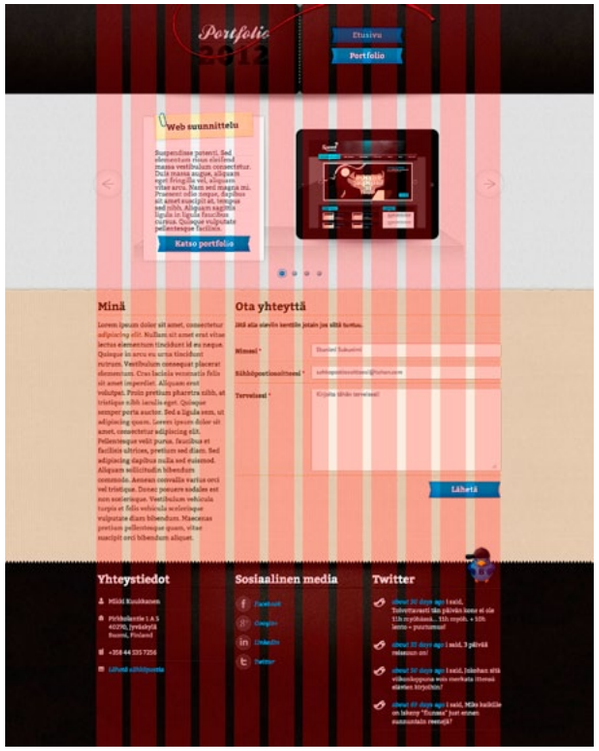
Webistä löytyy tänä päivänä monia palveluita jotka tarjoavat apuvälineitä gridin tekemiseen. Web-gridien suomalaista osaamista edustaa Jani Korpi (http://jonikorpi.com/), joka on kehittänyt monia erilaisiin tarkoituksiin sopivia gridejä.

Suurin yksittäinen web-sivun taittoon ja gridin asetteluun vaikuttava asia on päätelaitteen selainikkunan resoluutio. Aikaisemmin asiassa on yritetty löytää kultainen keskitie, eli absoluuttista sivun leveyttä, joka toimisi resoluutiolla kuin resoluutiolla. Mobiililaitteet on otettu yleensä huomioon todella huonosti. Mikäli tätä on yritetty, se on toteutettu jättämällä tyylitiedostot kokonaan näyttämättä, jolloin sisältö koostuu pelkästään html-tiedoston sisällöstä.

Nykyään www-sivuja selataan mobiililaitteilla yhä enemmän ja enemmän. Tämä tarkoittaa sitä, että sivustojen on pakko muovautua selainikkunoiden vaihteleville resoluutioille sopiviksi. Toisaalta taas näyttöjen fyysiset koot ja resoluutiot kasvavat jatkuvasti, myös mobiililaitteissa kasvu on tämän hetken kulkusuunta. Tämä antaa myös mahdollisuuden käyttää white spacea aivan uudella tavalla. Se tarkoittaa tyhjää tilaa elementtien välillä. Micro white space taas tarkoittaa tyhjää tilaa esimerkiksi tekstirivien välillä. White space parantaa oikein käytettynä luettavuutta, luo elementeille merkityksen, lisää hienostuneisuutta ja tasapainoa. (Ward, M., ym. 2011, 28) (Jones, H. 2010) Mobiililaitteilla voimme tilan säästämiseksi esimerkiksi vähentää marginaaleja tekstimassan ympärillä, mutta luettavuuden parantamiseksi voimme taas harventaa rivivälejä.

4.2 Palstat portfolio-sivustollani

Käytin omalla portfolio-sivustollani 12-palstaista gridiä, joka on 16-palstaisen gridin ohella todella paljon käytetty muun muassa aikakauslehtien taitossa. (Kuva 20.) Valitsin 12 palstaa 16:sta sijaan sen takia, ettei 16 ole jaollinen kolmella ja olin suunnitellut käyttäväni kolmea palstaa rinnakkain monissa tapauksissa. Kolme palstaa tuntuu olevan varsinkin tekstikappaleisiin sopivan levyinen varsinkin 960- ja 720-pikselisellä resoluutiolla. Tekstirivin ihanne merkkimäärä on 55-60 merkkiä ja jos otamme fonttikooksi 14-pikseliä, pääsemme kolmella yhtä leveällä vierekkäisellä palstalla aika lähelle tätä. Näin lukija säästyy turhilta rivinvaihdoilta ja katkonaiselta lukemiselta. (Itkonen, M. 2007, 84)



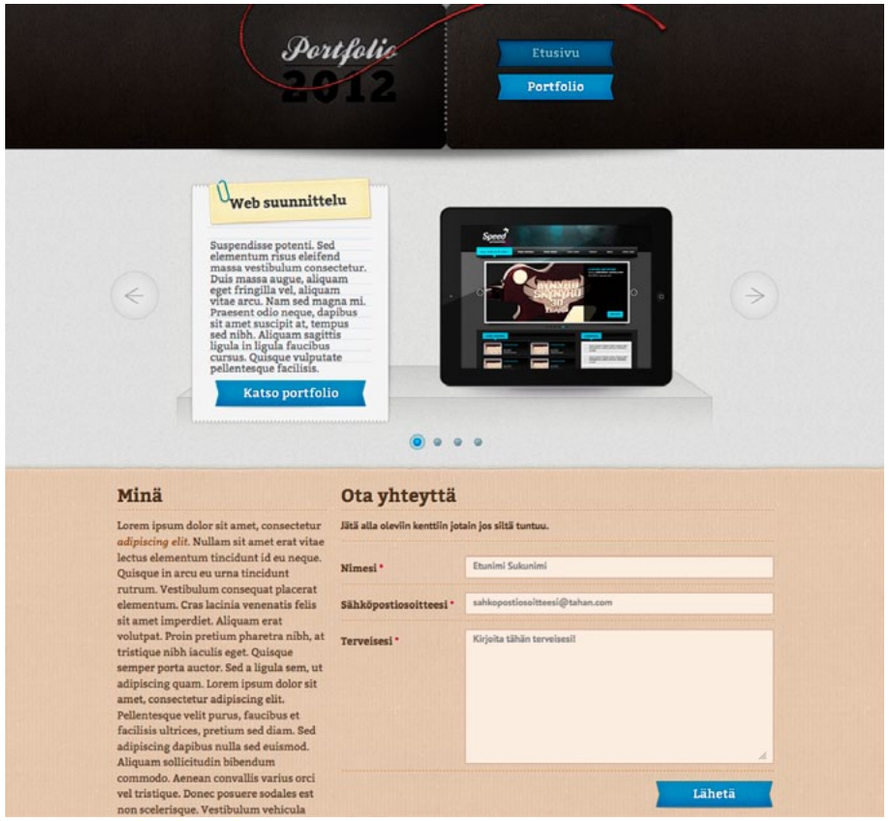
Kuva 20.
12-palstainen gridi portfolio-sivustolla.

Omalla sivustollani käytin kuitenkin 960 Grid Systemiä, joka on varmasti tunnetuin ja tätä myötä myös käytetyin gridi webissä. (Kotisivut löytyvät osoitteesta http://960.gs ja demo osoitteesta http://960.gs/demo.html.) Se toimii siten, että jos haluamme esimerkiksi kolme vierekkäistä palstaa, annamme kolmelle peräkkäiselle elementille luokan ”grid_4”. Tällöin kukin palsta vie 12:sta palstasta tilaa neljän palstan verran. Tällöin yhdelle ”riville” mahtuu luonnollisesti kolme palstaa rinnakkain. Jos taas haluamme yhden kolmasosan kokoisen palstan ja yhden kahden kolmasosan tilaa vievän palstan. Annamme ensimmäiselle elementille luokan ”grid_4” ja toiselle ”grid_8” ja niin edelleen. (Kuva 21.)

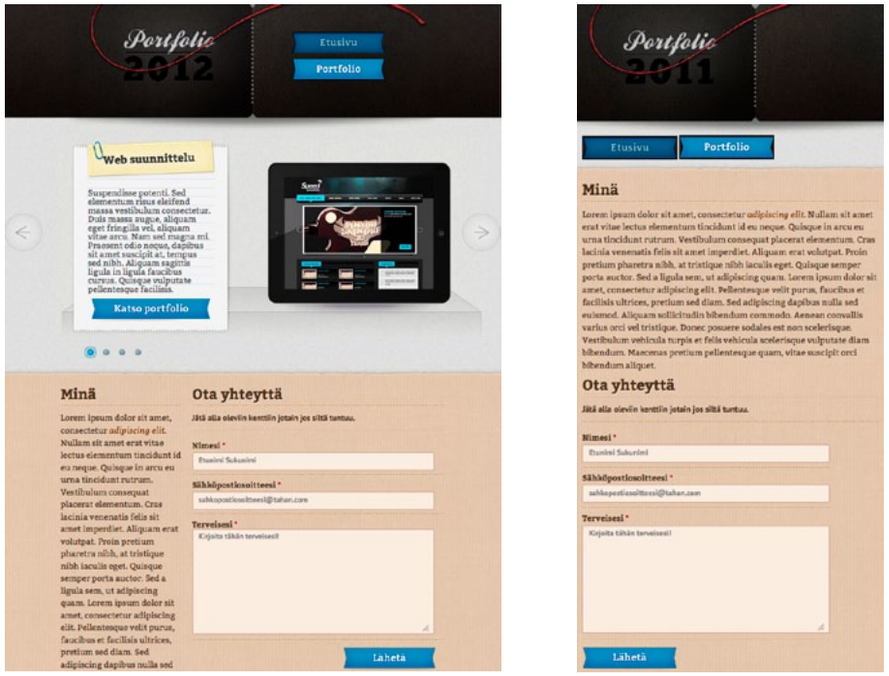
Kuva 21.
Gridit portfolio-sivustolla.

Valitsin tämän gridin, koska sen yhteyteen on mahdollista asentaa helposti avustava JavaScript-tiedosto Adapt.js (http://adapt.960.gs), joka sovittaa gridin sisältöosan ja palstojen määrän sopivaksi aina alle 760:n pikselin mobiiliresoluutioista yli 2540:n pikselin näyttöihin. Teknisesti tämä tapahtuu niin, että adapt.js-tiedosto ottaa selvää ikkunan resoluutiosta ja antaa selaimen käyttöön selainikkunan resoluutioon sopivan CSS-tiedoston, joka taas kertoo selaimelle kuinka leveitä palstojen täytyy olla ja kuinka monta niitä tulee olla vierekkäin. Tällöin esimerkiksi kännykällä selattuna palstat siirtyvät allekkain, jolloin sivun osia tarvitse enää erikseen suurentaa, jotta tekstiä pystyttäisiin helposti lukemaan. Näin

luettavuus ja käytettävyys paranevat huomattavasti. Kuvat voidaan pienentää css:n avulla mitä pieniresoluutioisemmalla päätelaitteella niitä luetaan ja vastaavasti suurentaa kun resoluutiokin suurenee. Olen myös piilottanut joitain sivuston osia alle 720-pikselin mobiiliresoluutiolta. Esimerkiksi etusivun raskas kuvakaruselli ei näy pieniresoluutioisilla mobiililaitteilla luettaessa lainkaan, koska suurien kuvien ansiosta se latautuu hitaasti ja tien päältä portfoliota selaileva potentiaalinen asiakas saattaisi turhautua jo heti alkumetreillä sivuston hitauteen. Käytännössä sivut voivat siis olla aivan täysin eri näköiset erilaisilla resoluutioilla katsottuna, ainoastaan CSS luo rajat sille kuinka voimme sivustoa muokata eri resoluutioihin sopiviksi. (Kuvat 22 ja 23.)



Kuva 22.
Sivusto yli 960-pikseliä leveässä selainikkunassa.



Kuva 23.
Vasemmalla sivusto yli 720 pikseliä, ja oikealla alle 720 pikseliä leveissä selainikkunoissa.

5 TYPOGRAFIA WEBISSÄ

5.1 Typografian alkuajat webissä

Web typografian alkoi pikkuhiljaa kehittymään vuonna 1995, kun silloinen Netscape lisäsi font-tagin HTML:ään. Tällöin itse tekstin ladontaan ja muihin ulkonäöllisiin seikkoihin pystyttiin vaikuttamaan todella vähän. Vasta CSS:n tultua mukaan pystyttiin vaikuttamaan muun muassa kirjainten ja sanojen välistyksiin, asemointiin, suuntaan ja tyyleihin. Suurin ongelma oli kuitenkin käytettävissä olevien kirjasinten pieni määrä. Valinta oli tehtävä muutaman niin sanotun web safe-fontin joukosta. Yleisesti ottaen se tarkoitti valintaa Arialin, Verdanan, Times New Romanin, Georgian ja muutaman muun kirjasimen väliltä. (W3schools.com, 2012) Tämä johtui siitä, että fontti oli oltava asennettu koneelle, jonka selaimella käyttäjä sivustoa katseli. Käyttöjärjestelmienkin fonttivalikoimissa on eroja, joka vaikutti siihen, että valikoima oli todella suppea. (Ward, M., ym. 2011, 173)

Joskus sivustoilla kuitenkin näkyi näiden niin sanottujen turvallisten web-fonttien lisäksi muita fontteja. Hyvin usein ne oli toteutettu niin sanotulla ”image replacement” -tekniikalla. Tällöin HTML-elementille määritetään tyylitiedostossa taustakuva jossa teksti on. Itse HTML-teksti taas piilotetaan antamalla sille miinus-merkkinen sisennys jolloin teksti katoaa selainikkunan ”ulkopuolelle”. Tekniikka on todella työläs, varsinkin jos kyse on useista erillistä otsikoista tai tekstipätkistä, puhumattakaan siitä, että joudumme usein muokkaamaan tekstiä, jolloin meidän täytyy tehdä joka kerta uusi kuva. (Gasston, P. 2011, 197-198). Leipätekstin korvaajaksi tästä tekniikasta ei siis ole, vaan korkeintaan yksittäisiä tekstejä ja otsikkoelementtejä korvaamaan. Joskus vastaan tulee tapauksia, joissa teksti on pelkkä kuva, jolloin käytettävyys kärsii. Tällöin jos sivustoa katsellaan ilman tyylitiedostoa teksti ei näy ollenkaan, eivätkä myöskään hakukoneet osaa indeksoida sivustoa oikein.

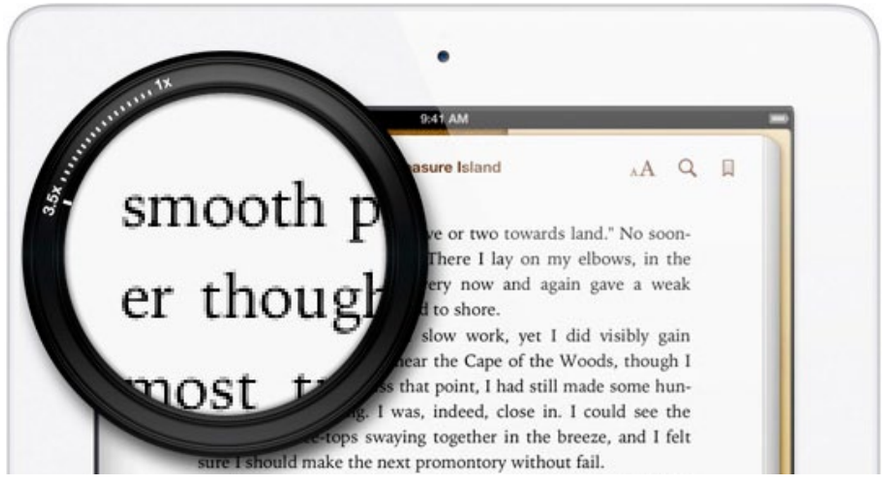
Muitakin tapoja käytettiin, myös Flashin ja JavaScriptin avulla on mahdollista näyttää käyttäjälle web safe-fonttivalikoiman ulkopuolisia kirjasimia. Käytetyimmät tekniikat ovat luultavasti sIFR ja Cufón. SIFR käyttää sekä Flashia että JavaScriptia fontin näyttämiseen. Sen huono puoli on luonnollisesti se, että käyttäjän selaimeen on oltava asennettuna Flash-liitännäinen ja selaimessa on oltava JavaScript päällä. Lisäksi tämä tekniikka on hyvin raskas. Cufon käyttää apunaan JavaScriptia näyttämään fontin. Tämän tekniikan haittapuoliin taas kuuluu muun muassa se, ettei tekstiä pysty valitsemaan. (ThinkClay.com, 2009).

5.2 Web-typografian kehitys nykypäivään

CSS2 määrittely julkaistiin vuonna 1998, se sisälsi määrityksen nimeltään @font-face, joka antoi vihdoin suoran vaihtoehdon linkittää webiin fontteja, jotka eivät kuuluneet web safe fontteihin. Kaikki ei kuitenkaan mennyt aivan kuten piti, Microsoft ja Netscape valitsivat omat fonttiformaattinsa sen sijaan, että olisivat tukeneet siihen aikaan kaikkein yleisintä formaattia, TrueTypea. Koko asia tavallaan kuivui kasaan ja lopulta päädyttiin siihen, että oli kehitettävä uusi vapaa fonttiformaatti. Näin syntyi WOFF, eli Web Open Font Format, joka on pakattu versio tutuista TTF (TrueType Font) ja OTF (OpenType Font) fonttiformaateista. Kaikki tunnetut selainvalmistajat tukevat tänä päivänä WOFF-formaattia, joten @font-facea pystytään jo melko huoletta käyttämään, unohtamatta tietenkään fallbackeja vanhemmille selaimille. (Ward, M., ym. 2011, 174-175), (Castro, E., Hyslop, B. 2012, 354).

@font-face siis mahdollistaa siis yhä laajemman kirjasinvalikoiman, ja CSS:n avulla pystymme paneutumaan yhä syvemmälle typografiaan. Enää typografia webissä ei tarkoita enää pelkästään kirjasimen valintaa, vaan meidän on mahdollista ottaa huomioon rivivälit, rivien pituudet, värimaailma, marginaalit, oikeat glyyfit ja niin edelleen. (Ward, M., ym. 2011, 176).

Yksi web-typografian ja muun sähköisen typografian ongelmista on ollut näyttöjen pienet resoluutiot. Kun hyvän printtimedian resoluutio on noin 300 pistettä tuumalla, on vastaava pistetarkkuus ollut näytöillä yleensä alle 100. Tämä on näkynyt käyttäjälle epäselvänä tekstinä, koska kirjainten muodot eivät ole kovinkaan tarkat. Nyt markkinoille alkaa tulla resoluutioltaan yhä tarkempia näyttöjä, jolloin myös luettavuus paranee, eikä eroa printtimaailman tarkkuuteen enää juuri ole. Tämä tulee varmasti vauhdittamaan web-typografian suosiota. (Kuvat 24 ja 25.)



Kuva 24.
Kirjaimia pieniresoluutioisella näytöllä.

Kuva 25.
Kirjaimia suuriresoluutioisella näytöllä.

5.3 Kirjasinpalvelut netissä

Yksi @font-facen ”ongelmista” on se, että fontti on tällä tekniikalla toteutettuna aina käyttäjän ladattavissa omalle koneelleen. Sen takia meidän täytyy ottaa huomioon aina lainopillinen puoli @font-facea käytettäessä. Vaikka olisit ostanut fontin omaan käyttöösi, et voi käyttää sitä @font-facen avulla juuri siitä syystä, että kuka vaan pystyy sen tallentamaan sivustolta omalle koneelleen. Freeware-lisensoidut fontit ovat sellaisia, joita voidaan huoletta käyttää @font-facen avulla, sillä ne ovat vapaita käytettäviksi niin omaan, kuin kaupalliseenkin käyttöön. (Goldstein, A. Lazaris, L. & Weyl, E. 2011, 205). Nykyään internetistä löytyy monia palveluja, jotka tarjoavat joko lisensoituja fontteja maksua vastaan, tai avoimen lähdekoodin fontteja ilmaiseksi. Esimerkiksi Google tarjoaa palvelun (http://www.google.com/webfonts), jonka avulla pystytään lisäämään web sivuille avoimen lähdekoodin fontteja suoraan Googlen omalta palvelimelta. HTML:n <head>-osioon tarvitsee ainoastaan lisätä linkki, joka osoittaa Googlen palvelimella sijaitsevan fontin omaan CSS-tiedostoon. Tämän jälkeen fonttia pystytään käyttämään aivan normaalisti CSS-tiedostossa lisäämällä font-familyyn fontin nimi. Suuri etu on myös se, ettei fontteja tarvitse lisätä omalle palvelimelle, vaan ne tulevat Googlen palvelimilta. Osa fonttipalveluista tarjoaa fontteja ladattaviksi omalle palvelimelle. Tällöin sinun pitää itse huolehtia fonttien oikeasta linkityksestä CSS-tiedostoihin. Font Squirrel (http://www.fontsquirrel.com/) on toinen tunnettu ilmaisfonttisivusto ja se tarjoaa myös palvelun, jonka avulla omia fonttejaan voi muuntaa moniin eri muotoihin. Se tekee puolestasi myös @font-face määritykset CSS-tiedostoa varten. Myös maksullisia palveluita löytyy, kuten esimerkiksi Typekit (https://typekit.com/) ja Fontdeck (http://fontdeck.com/). Hinnoittelu on kuukausi tai vuosipohjainen, ja hinta määräytyy kuukausittaisen vierailukertojen ja fonttien lukumäärän mukaan. Maksulliset palvelut tarjoavat paljon tunnettujen kirjasintoimistojen suunnittelemia fontteja, joiden laatu on varmasti parempaa kuin osassa avoimen lähdekoodin fonteissa. (Ward, M., ym. 2011, 188-189)

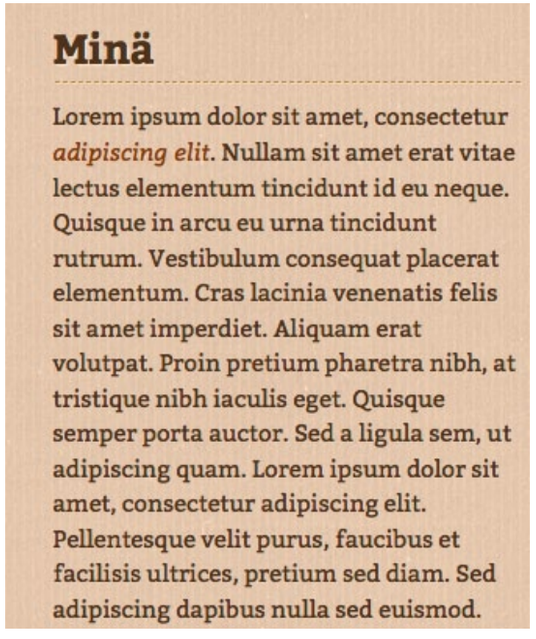
5.4 Typografia portfolio-sivustollani

Omassa portfoliossani päädyin käyttämään Googlen Web Fonts palvelua lähinnä sen helppouden ja verrattain hyvän kirjasinvalikoiman takia. Valintaan vaikutti myös se, että pidän Googlen palvelimia luotettavina ja nopeina.

Leipätekstifonttina käytin kirjasinta nimeltä ”Bitter”. Se on suunniteltu varta vasten kirjasimeksi näytöltä luettavaan tekstiin. (Fontsquirrel.com, 2012.) Sillä on suuri x-korkeus, eli, joka vaikuttaa siihen, että itse teksti vaikuttaa verrattain suurelta. (Itkonen, M. 2007, 83-84) Käytin leipätekstissä viidentoista pikselin kokoa, joka näyttää esimerkiksi Times New Romaniin verrattuna todella isolta. Kuten D Bnonn Tennant kertoo Smashing Magazinen artikkelissaan 16 Pixels For Body Copy, useimpien selaimien oletusfonttikoon olevan 16 pikseliä. Näin ei ole sattumalta, sillä karkeasti sanoen 16 pikselin koko vastaa 10 pisteen fonttikokoa kirjassa. Tämä johtuu siitä, että katselemme kirjaa huomattavasti lähempää, kuin mitä katsomme näyttöjämme. Tällöin luettavuus on huipussaan, eivätkä silmämme rasitu samalla tavalla kuin pientä tekstiä luettaessa. Muutenkin meidän täytyy ottaa huomioon heikkonäköiset ihmiset ja iän mukanaan tuoman normaalin näön heikentymisen. (D Bnonn Tennant, 2011).

Rivivälin nostin 22,5 pikseliin, sillä halusin tekstin olevan hieman väljempää ja näin ollen helpommin luettavaa. Liian väljästä tekstistä taas on pelkästään haittaa, sillä se rikkoo kappaleen yhtenevyyden ja hankaloittaa lukijan siirtymistä riviltä toiselle. Myös fontti, jossa on suuri x-korkeus kaipaa hieman tavallista suuremman rivivälin, jotta teksti ei näyttäisi ahtaalta. (Itkonen, M. 2007, 84-85)

Linkkien korostamiseksi käytin värikontrastin lisäksi kursivointia, joka tuo linkit hyvin esiin diagonaalisten linjojensa avulla. Valitsin myös käytettävän fontin niin, että sillä täytyi olla normaalin leikkauksen lisäksi ainakin yksi kursiivi ja lihava leikkaus. (Kuvat 26 ja 27.)



Kuva 26.
Typografiaa portfolio-sivustolla 1.

Kuva 26.
Typografiaa portfolio-sivustolla 2.

6 YHTEENVETO

6.1 Uudet tekniikat webissä

Tulevaisuudessa animointi tulee varmasti keskittymään kokonaan Flashin ulkopuolelle CSS:n ja JavaScriptin avulla tuotettavaan animaatioon. Juuri tällä hetkellä CSS-animaatiota on käytettävä todella harkiten ja otettava huomioon myös selaimet, jotka eivät tue uusimpia ominaisuuksia. Mitään kovin bisneskriittistä ei pelkän CSS:n avulla kannata toteuttaa. CSS:n osuus kasvaa kuitenkin jatkuvasti, varsinkin yksinkertaisempia animaatioita toteutettaessa. CSS tulee varmasti tulevaisuudessa ajan saatossa muotoutumaan yhä monimutkaisemmaksi animointialustaksi paremman selaintuen ja uusien ominaisuuksien myötä.

Taitto on mennyt paljon lehtimaailman suuntaan palstoihin perustuvan taiton tultua mukaan kuvioihin. Nykyään sivustoa suunniteltaessa täytyy ottaa huomioon useat eri resoluutiot, aina suurista työpöytänäytöistä mobiililaitteisiin. Ulkoasun täytyy siis venyä moneen eri kokoiseen näyttöön. Myös sisältöä täytyy pystyä karsimaan järkevästi, jotta jokaisella päätelaitteella käyttökokemus olisi mahdollisimman miellyttävä. Samalla sisällöllä voi nykypäivän järkevästi suunnitellulla sivustolla olla lähes lukemattomia eri ulkomuotoja, kun mukaan luetaan useat eri päätelaitteet ja vanhat selaimet.

Typografia tulee kehittymään webissä printtimediaa vastaavaksi. Monet kirjasintoimistot kehittävät fontteja myös varta vasten näytöltä luettaviksi. Kirjasinkaupoista onkin nykyään melkein poikkeuksetta saatavilla myös webversiot kirjasimista. Tällä hetkellä kirjasinten käyttö on varsinkin ulkoisten palveluiden avulla todella helppoa. Kirjasinten tarjonta on kasvussa ja laatu paranemassa. En näe mitään estettä muidenkin kuin niin sanottujen ”web-safe” fonttien käytölle, varsinkin kun font-stackit suunnitellaan huolellisesti tarjoamalla vanhemmillekin selaimille tutut ja turvalliset fontit vaihtoehdoiksi.

6.2 Portfolio-sivuston toteutus

Portfolion toteutus onnistui mielestäni loppujen lopuksi varsin hyvin. Pystyin hyödyntämään CSS3 ja JavaScript tekniikoita saadakseni aikaan käytettävän ja resoluutioihin sopeutuvan sivuston. HTML5:n vaikutus tutkimiini asioihin oli yllättävän vähäinen.

Ongelmia toki ilmeni, tekninen puoli vei yllättävän paljon aikaa, vaikka olin toki siihen varautunut, ettei kaikki mene heti kerralla oikein. Aivan kaikkia rautalankamalleihin suunniteltuja ominaisuuksia en ehtinyt toteuttaa lähinnä teknisen puolen ongelmien takia. Olisin myös halunnut animoida käyttöliittymää huomattavasti enemmän. Esimerkiksi etusivun ja portfolio-sivujen välille olisin halunnut saada jonkin näköisen siirtymän. Tekninen osaaminen kuitenkin asetti jonkin verran rajoituksia. En saanut portfoliota täysin valmiiksi, koska ajattelin, että sisällöntuotanto ei välttämättä ole kovin tärkeää tämän opinnäytetyön osalta. Myös ajankäytöllisistä syistä näin järkeväksi keskittyä enemmän itse opinnäytetyön sisältöön vaikuttavien asioiden suunnitteluun. Jätän siis sisällöntuotannon poikkeuksellisesti vasta opinnäytetyön jälkeiseen aikaan.

6.3 Mitä tulevaisuudessa?

Elämme tällä hetkellä uusien tekniikoiden murrosvaihetta, joten kestää aina aikansa ennen kuin pystymme ottamaan aivan uusimmat CSS-määritykset käyttöömme. World Wide Web Consortium (W3C) kehittää jo kovaa vauhtia CSS4:sta, ennen kuin CSS3 on edes ehtinyt näyttää täyttä potentiaaliaan.

Käytän tällä hetkellä työssäni HTML5:ä ja CSS3:a ja aion kehittää itseäni näiden uusien tekniikoiden parissa koska uskon, että niissä on web-suunnittelun tulevaisuus. Asiakasprojekteja tehdessä on tärkeää ottaa huomioon se, mitä tekniikoita voidaan käyttää jo nyt ja mitä kannattaa vielä hetken antaa odottaa. Se riippuu myös siitä, kuinka hyvin voimme rakentaa uusille tekniikoille niin sanottuja fall-backejä vanhemmille selaimille ja kuinka työlästä se on. Eli esimerkiksi tällä hetkellä on olemassa CSS3-määrityksiä jotka toimivat pelkästään WebKit selaimilla, eli Chromella ja Safarilla. Näitä ei ole vielä mitään järkeä käyttää muuta kuin demotarkoituksessa tai ”lisäherkkuna”, sillä käyttäjien saavutettavuus on aivan liian pieni.

Tällä hetkellä mobiilisovellukset näyttävät suuntaa. Niihin panostetaan tänä päivänä todella paljon, joten niitä kannattaa seurata ja ottaa niistä mallia. Itse uskon, että myös web kehittyy jatkuvasti mobiililaitteiden ja sovellusten tämän hetkiseen suuntaan. Tämä tarkoittaa käytännössä sitä, että meidän täytyy alkaa ottamaan huomioon kosketusnäytöt myös web-suunnittelussa. Jos suunta jatkuu samanlaisena, jonain päivänä kaikki näytöt tulevat olemaan kosketusnäyttöjä.

LÄHTEET

Baker, L. 2006. 5 Reasons Not to Use Flash [viitattu 26.4.2012] Saatavissa: http://www.searchenginejournal.com/5-reasons-not-to-use-flash/3949/

Castro, E., Hyslop, B. 2012. Visual Quickstart Guide - HTML5 and CSS3. Seitsemäs painos. Berkeley, USA: Peachpit Press.

Clarke, A. 2007. Transcending CSS, The Fine Art of Web Design. Berkeley, USA: New Riders.

Cossey, M. 2012. Mission Transition. Smashing Magazine [viitattu 13.2.2012]. Saatavissa: http://uxdesign.smashingmagazine.com/2012/02/28/mission-transition/.

Coyier, C. 2011. Fade Image Into Another (within a Sprite). CSS-Tricks [viitattu 26.2.2012] Saatavissa: http://css-tricks.com/fade-image-within-sprite/.

CSS3.info/Chris. 2012. Introducing the CSS3 Test. CSS3.info [viitattu 4.3.2012] Saatavissa: http://www.css3.info/introducing-the-css3-test/.

D Bnonn T. 2011. 16 Pixels For Body Copy. Anything Less Is A Costly Mistake. Smashing Magazine [viitattu 26.3.2012] Saatavissa: http://www.smashingmagazine.com/2011/10/07/16-pixels-body-copy-anything-less-costly-mistake/

Fontsquirrel.com, 2012. Bitter. [viitattu 26.3.2012] Saatavissa: http://www.fontsquirrel.com/fonts/bitter

Gasston, P. 2011. The Book Of CSS3 - A Developer’s Guide to the Future of Web Design. San Francisco: William Pollock.

Goldstein, A. Lazaris, L. & Weyl, E. 2011. HTML5 & CSS3 for the Real World. Collingwood, Australia: Sitepoint Pty. Ltd.

Itkonen, M. 2007. Typografian käsikirja. Kolmas, laajennettu painos. Helsinki: RPS-yhtiöt.

Jobs, S. 2010. Thoughts on Flash. [viitattu: 2.5.2012] Saatavissa: http://www.apple.com/hotnews/thoughts-on-flash/

Jones, H. 2010. Whitespace: The Underutilized Design Element. [Viitattu 29.4.2012] Saatavissa: http://webdesignledger.com/tips/whitespace-the-underutilized-design-element

Markoff, J. 2007. The Real History of WYSIWYG. [viitattu 28.4.2012] Saatavissa: http://bits.blogs.nytimes.com/2007/10/18/the-real-history-of-wysiwyg/

Nielsen, J. 2000. Flash 99% Bad. [Viitattu 26.4.2012] Saatavissa: http://www.useit.com/alertbox/20001029.html

Samara, T. 2002. Making and Breaking the Grid. Beverly, USA: Rockport Publishers, Inc.

ThinkClay.com, 2009. Cufón vs sIFR vs FLIR [viitattu 14.3.2012] Saatavissa: http://thinkclay.com/technology/cufon-sifr-flir.

Ward, M., Charchar, A., Inchauste, F., Rundle, M., Jovanovic, J., Heilmann C., Anayian, V., Kolb, C., Weinschenk, S., Bradley, S. & the Smashing Magazine Team. 2011. The Smashing Book #2. Freiburg, Germany: Smashing Media

Wesley, A. 1998. Raggett HTML 4 [viitattu 24.4.2012] Saatavissa: http://www.w3.org/People/Raggett/book4/ch02.html

Wilson, N. 2002. All-Flash A Fast Track to Failure. [viitattu 26.4.2012] Saatavissa: http://www.sitepoint.com/flash-fast-track-failure/

W3schools.com. 2012. CSS Web Safe Combinations [viitattu: 4.3.2012] Saatavissa: http://www.w3schools.com/cssref/css_websafe_fonts.asp.

W3.org. 2012. The CSS Saga [Viitattu 26.4.2012] Saatavissa: http://www.w3.org/Style/LieBos2e/history/