

Kustomoitu responsiivinen WordPress-sivusto suomalaiselle Toisenlainen-elokuvalle

Kati Räsänen



Tekijä Kati Räsänen	
Koulutusohjelma Tietojenkäsittelyn koulutusohjelma	
Opinnäytetyön otsikko Kustomoitu responsiivinen WordPress-sivusto suomalaiselle Toisenlainen-elokuvalle	Sivu- ja liitesivumäärä 44 + 4
Opinnäytetyön otsikko englanniksi Customized responsive WordPress-website for a Finnish movie called Toisenlainen	
<p>Opinnäytetyö kertoo Toisenlainen-nimiselle suomalaiselle elokuvalle toteutetusta verkkosivuprojektista. Sivuston pohjana käytettiin WordPress-sisällönhallintajärjestelmää ja sivuston teossa huomioitiin sen skaalautuvuus sekä käytettävyys eri päätelaitteilla.</p> <p>Tutkimusraportissa käsitellään myös tarkemmin responsiivista verkkosivukehitystä sekä sitä, millaisia uusia ajatus- ja toimintamalleja se verkkosivujen tekijöiltä vaatii. Lisäksi raportissa käydään läpi responsiiviseen kehitykseen liittyviä teknisiä vaatimuksia sekä hyväksi havaittuja käytäntöjä. Teorian taustana on käytetty pääasiassa Matthew Carverin The Responsive Web, Phil Dutsonin Responsive Mobile Design sekä Steve Krugin Don't Make Me Think -kirjoja. Teknisistä vaatimuksista käydään läpi mm. sivuston mittasuhteiden määrittämistä, kuvien asettelua ja optimointia sekä testausta.</p> <p>Raportin loppupuoliskolla perehdytään varsinaiseen elokuvasivuston toteutukseen. Siinä kerrotaan tarkemmin WordPressistä sekä miksi se valittiin tämän kyseisen projektin julkaisualustaksi. Lisäksi käydään läpi projektin kulkua, sen tekoon käytettyjä teknisiä ratkaisuja sekä tietysti arvioidaan valmista lopputulosta. Sivuston teossa käytettiin mm. responsiivista Bootstrap-viitekehystä, Less-tyylikieltä, javascriptia sekä WordPressin kustomoituja sisältötyyppejä. Sivusto julkaistiin maaliskuun lopulla 2016 osoitteessa toisenlainenelokuva.com, jonka jälkeen sivustoa on kehitetty ja ylläpidetty.</p>	
Asiasanat Responsiivinen kehitys, mobiilioptimointi, WordPress-sivusto, asiakasprojekti, verkkosivukehitys, moderni verkkosivukehitys	

Author Kati Räsänen	
Degree programme Business Information Technology	
Report/thesis title Customized responsive WordPress-website for a Finnish movie called Toisenlainen	Number of pages and appendix pages 44 + 4
<p>The bachelor's thesis report describes a website building project, which was made for a Finnish movie called Toisenlainen. The final website uses WordPress -content management system. The site has also been made so that it is fully responsive and is easy to use.</p> <p>The report explains the theory behind responsive web design and how it has changed the mindsets and working patterns of the web development teams in general. Also the technical requirements and best practices of responsive design are explained in the text. The technical requirements part includes among other topics configuring the measurement units for the site, picture optimization and site testing. The theory is mainly based on the responsive design and usability related books written by Matthew Carver, Phil Dutson and Steve Krug.</p> <p>The final part of the report focuses on building the actual movie site. WordPress as a content management system is explained more in details, and also the reasons why it was chosen for this particular project. The report opens up the website building as a process and specifies which techniques have been used to make the site work in the optimal way on mobile devices.</p> <p>The final result and the goals it was aiming for have been reviewed at the end of the report. The final site uses techniques such as responsive Bootstrap-framework, Less -styling language, Javascript and WordPress custom post types. The site was published at the end of March 2016 under the domain name toisenlainenelokuva.com. Since the release site has been maintained and developed further.</p>	
Keywords Responsive development, mobile optimization, WordPress-website, customer project, web development, modern website	

Sisällys

1	Johdanto	1
	Sanasto	3
2	Responsiivinen suunnittelu vaatii muutosta ajatusmallissa	5
2.1	Sisältö edellä	5
2.2	Sisällönsuunnittelu = hakukoneoptimointi.....	5
2.3	Projektiryhmän vuorovaikutus	6
2.4	Käytettävyys	6
3	Tekniset huomiot responsiivisessa kehityksessä.....	8
3.1	Mittasuhteet	8
3.2	Rakenne	9
3.3	Typografia.....	9
3.4	Kuvat	9
3.4.1	Sopiva kuvaformaatti.....	10
3.4.2	Skaalautuvuus	11
3.5	Latautuminen	12
4	Testaus	13
4.1	Käytettävyystestaus	13
4.2	Toiminnallinen testaus	13
5	WordPress julkaisualustana	16
5.1	Miksi WordPress valittiin juuri tähän projektiin?.....	16
6	Projektin kulku.....	18
6.1	Kehitysympäristö.....	18
6.2	Sivuston suunnittelu ja ulkoasu	19
6.2.1	Tarpeet	19
6.2.2	Rautalangat.....	19
6.2.3	Ulkoasu.....	20
6.3	Koodin rakentaminen	20
6.4	Käytetyt tekniikat.....	21
6.4.1	WordPress-teema	21
6.4.2	WordPress-lisäosat eli pluginit	21
6.4.3	Custom Post Type.....	22
6.4.4	Bootstrap	28
6.4.5	Javascript.....	31
6.4.6	Php	32
6.4.7	Less	34
6.5	Yhteydenpito toimeksiantajan kanssa	34
6.6	Julkaisu.....	35

6.7	Julkaisun jälkeen.....	36
6.7.1	WordPressin suojausasetukset	36
6.7.2	Latautumisen optimointi	37
7	Pohdinta.....	40
	Lähteet	42
	Liitteet.....	44
	Liite 1. Suunnittelukokouksen muistilista	44
	Liite 2. Pelkistetty rautalankamalli.....	45
	Liite 3. Kuvitettu rautalankamalli työpöytäversiosta	46
	Liite 4. Kuvitettu rautalankamalli mobiiliversiosta	47
	Liite 5. Valmis sivusto työpöytänäkymässä.....	48

1 Johdanto

Asetin oppinäytetyöni tavoitteeksi, että pääsen toteuttamaan jonkin konkreettisen työn ja kokeilemaan samalla erilaisia teknisiä ratkaisuja sekä treenaamaan niiden käyttöä. Ehdotin yrittäjäyhtävälleni yhteistä projektia, johon hän tarttui mielellään. Työksi valikoitui verkkosivuston toteuttaminen Toisenlainen-nimiselle elokuvalle, jota ystäväni on suunnitellut ja tehnyt jo useamman vuoden. Sivuston teon lisäksi päätin perehtyä responsiiviseen verkkosivukehitykseen ja kerron oppinäytetyössäni tarkemmin siihen liittyvistä vaatimuksista.

Ystäväni perustama tuotantoyhtiö T2Tuotanto on parin vuoden ajan kuvannut Toisenlainen-nimistä suomalaista elokuvaa vapaaehtoisvoimin. Elokuva voisi kuvailla toimintadraamaksi, jonka punaisena lankana kulkee päähenkilöiden välinen hieman toisenlainen rakkaustarina. Kaikki elokuvan tuotot lahjoitetaan lasten ja nuorten harrastustoimintaa tukevalle hyväntekeväisyysjärjestö Hope Ry:lle. Projektiin onkin matkan varrella lähtenyt mukaan joukko yrityskumppaneita sekä tunnettuja suomalaisia näyttelijöitä tukemaan mielenkiintoista ideaa ja hyvää asiaa. Olen itse ollut ohjaajaystäväni kautta projektissa mukana kuvausavustajana ja seurannut elokuvan edistymistä sen alkumetreistä asti. Tänä keväänä 2016 Toisenlainen sai vihdoinkin ensi-iltansa.

Sivustoprojektin valmistumisaikatauluksi päätettiin huhtikuun puoliväli, jolloin elokuvan oli määrä saada ensi-iltansa. Sivuston tarkoitus oli tehdä elokuvaa ja sen tekijöitä tunnetuksi, markkinoida elokuvaa katsojille sekä houkutellessa uusia yhteistyökumppaneita ja rahoittajia tulevia elokuvaprojekteja varten. Koko elokuvan tuotantoprosessin ajan oli kerätty erilaista kuva- ja videomateriaalia kotisivuja sekä markkinointia ajatellen, mikä tuki loistavasti sivuston sisällön suunnittelua.

Teknisesti tavoite tarkoitti sitä, että sivuston täytyi olla moderni ja toimiva, mikä tänä päivänä tarkoittaa mm. sivujen sujuvaa skaalautuvuutta eri päätelaitteilla. Responsiivinen verkkosivukehitys oli minusta mielenkiintoinen tavoite, sillä mielestäni se on tulevaisuudessa ja jo nyt oletusarvo kaikille uusille sivustohankkeille. Koska responsiivisuus on vielä kohtalaisen tuore käsite, siihen liittyvät metodit ja työskentelytavat kehittyvät jatkuvasti. Ajattelin, että tässä vaiheessa on hyvä ottaa perusasiat haltuun tällä saralla.

Responsiiviseen suunnitteluun ja kehitykseen liittyy paljon teknisesti tärkeitä huomioita kuten se, miten elementit saadaan sopimaan sujuvasti sekä pienelle että isolle ruudulle. Vaikka elementit saisikin skaalautumaan kauniisti laitteeseen kuin laitteeseen, omaksi ongelmakseen muodostuu latautuvuus. Jos isolle HD-resoluutioiselle näytölle tarkoitettua ku-

vat, videot ja tyylit haluaa puristaa pieneen älypuhelimeen, joka kaikenlisäksi käyttää hidasta pakettidataa, sivuston latautuminen kestää kärsimättömän käyttäjän näkökulmasta liian kauan. Latautuvuus ja sen optimointiin liittyvät asiat olivatkin mielestäni aiheen mielenkiintoisin osuus.

Käyn opinnäytetyössäni ensin läpi yleistä tietoa responsiivisesta kehityksestä ja mitä uusia näkökulmia se on tuonut yleiseen sivustokehitykseen. Vanhat työskentelytavat ovat paikoin liian kankeita responsiivisen kehityksen tarpeisiin ja siksi koko työyhteisön täytyy haastaa ajatusmallinsa ja olla valmiita mukautumaan. Lisäksi kerron responsiivisen kehityksen teknisistä vaatimuksista ja hyväksihavaituista käytännöistä. Käyn muunmuassa läpi kuvien asettelua ja optimointia, sivuston mittasuhteiden määrittämistä sekä testausta.

Pohjustuksena varsinaiseen sivuston rakennusprosessiin, kerron käyttämästäni WordPress-sisällönhallintajärjestelmästä ja siitä miksi valitsin juuri sen tähän projektiin. Työni loppuosa käsittelee elokuvasivuston tekoa sekä siinä käyttämiäni tekniikoita, joita pyrin havainnollistamaan koodin sekä kuvien avulla.

Sanasto

Blokata = Estää. Koodillisesti tai sisällönhallinnan kautta estää ei-toivottuja käyttäjien tai mahdollisia hyökkääjien toiminta verkkosivustolla.

Flat Design = Ulkoasukehityksen tyyliä, jossa suositaan puhtaita värejä ja selkeitä linjoja. Värit on ikäänkuin litistetty (flattened) minimiin ja ylimääräiset varjotukset, liukuvärit ja koristelut jätetty pois.

Footer = Alatunniste. Sivuston alaosasta erotettu alimmaisina alue, joka sisältää usein linkkejä ja yhteystietoja.

FTP-ohjelma = Ohjelma, jonka avulla voi ottaa yhteyden palvelimelle ja siirtää sinne turvallisesti tiedostoja.

Header = Ylätunniste tai otsake. Sivuston yläosasta erotettu alue, joka sisältää usein sivuston logon sekä navigaation.

Hover-efekti = Klikattava elementti, yleensä linkki, reagoi kun hiiren vie sen päälle. Usein elementti vaihtaa väriä tai saa alleviivauksen. Elementti muuttuu takaisin normaaliksi kun hiiren vie pois sen päältä.

jQuery = Suosittu Javascript-kirjasto, jonka kirjoitustapa poikkeaa hieman javascriptista. Mahdollistaa monipuolisia käsittelytoimintoja.

Kehittäjä = Koodaaja eli verkkosivujen tekijä.

Koodieditori = Ohjelma, joka helpottaa koodin kirjoittamista ja lukua. Ymmärtää koodin oletettua rakennetta, huomauttaa syntaksivirheistä, jäsentelee koodin eri elementtejä väreillä ja ehdottaa valmiiksi koodiin sopivia osia nopeuttaen kirjoittamista. Editoreihin on usein saatavilla monipuolisia lisäosia helpottamaan kehitystyötä entisestään. Koodieditoreita on useita erilaisia, joista kehittäjä voi valita mieleisensä.

Layout = Yleensä webgraafikon Photoshopin tai Illustratorin kaltaisella ohjelmalla toteutettu malli sivuston ulkoasusta. Kuvaa tarkasti sivuston asettelun, värit, fontit ja tekstuuri.

Mobile first = Ajatusmalli, jonka mukaan verkkosivukehitys aloitetaan mobiilikäyttäjän näkökulmasta ja mobiiliin optimoitua näkymää laajennetaan sitten suurempiin näyttökokoihin.

Rautalankamalli = Raakilemainen hahmotelma tulevan sivuston rakenteesta. Voidaan tehdä joko digitaalisena tai piirtää käsin paperille.

Renderöinti = Raakakoodin prosessointi lopulliseksi verkkosivuksi.

Responsiivisuus = Verkkosivuston skaalautuvuus tietokoneen ruudun ja pienen mobiililaitteen ruudun välillä eli sama sivusto toimii sujuvasti laitteella kuin laitteella.

Skripti = Jollakin ohjelmointikielellä (esimerkiksi javascriptillä) kirjoitettu komentosarja, joka tuottaa ohjelmallisesti halutun tuloksen.

Työpöytäkoko = Kannettavien tietokoneiden ja pöytäkoneiden käyttämä iso näyttökoko. Näytön leveys yli 990 pikseliä.

2 Responsiivinen suunnittelu vaatii muutosta ajatusmallissa

Responsiiviset sivustot ovat tulleet jäädäkseen. Siinä missä ennen täytyi perustella responsiivista ajattelua, nyt täytyykin argumentoida, miksi jokin sivusto kannattaisi tehdä vain työpöytäkoossa. Sivustojen suunnittelijat ovat kohdanneet uusia haasteita, kun sivuston kokonaisvaltaisessa suunnittelussa täytyykin huomioida paitsi eri sivunäkymät myös niiden skaalautuvuus, interaktiot, latautuminen sekä käyttömukavuus kaikilla laitteilla. Responsiivisuus haastaa kehittäjät paitsi teknisesti myös ajatustasolla. (Carver 2015, 1)

2.1 Sisältö edellä

Verkkosivuston tärkein pääoma on aina sen sisältö. Jos sivustolla ei ole mitään mielenkiintoista sisältöä, käyttäjät eivät viihdy siellä kauaa vaikka sivusto olisi kuinka visuaalinen tai rakenteellisesti taidokkaasti toteutettu. Erityisesti responsiivisessa verkkosivusuunnittelussa sisältö ja sen huolellinen kartuttaminen etukäteen korostuvat. Niin Matthew Carver kuin Steve Krugkin puhuvat sisällön puolesta, mutta Phil Dutson on pyhittänyt aiheelle kokonaisen luvun kirjassaan Responsive Mobile Design. Monet vanhemman koulukunnan kehittäjät ja designerit saattavat kokea responsiiviseen suunnitteluun liitetyn termin "mobile first" oudoksi ja vieraaksi - eli kehityksen mobiilikäyttäjälähtöisyys on monille vielä vaikeasti hahmotettava asia. Dutson kuitenkin muistuttaa, että "mobile first" voisi olla yhtä hyvin "content first". Responsiivisessa kehityksessä punainen lanka olisikin keskittyä nimeämään sisällön selkeään ulosantiin. (Dutson 2015, 15-26 sekä Krug 2014, 147)

Mobiililaitteissa on pienet ruudut ja kärsimättömät käyttäjät selaavat sivuja usein kiireessä. Tällöin on tärkeää, että he löytävät etsimänsä sisällön nopeasti ja loogisesti. Tämä tarkoittaa sitä, että sivustolle tuleva sisältö täytyy selvittää hyvin ja suunnitella sitten sivusto niin, että olennaisin, korkeimmalle priorisoitu tieto näytetään ensin tai on muuten helposti saatavilla käyttäjälle. Tästä näkökulmasta mobile first -ajattelu onkin pitkälle juuri sisällön laittamista etusijalle. (Dutson 2015, 15-26)

2.2 Sisällönsuunnittelu = hakukoneoptimointi

Sisältö vaikuttaa olennaisesti myös nykypäivän digibisneksessä erityisen tärkeäksi nostettuun seikkaan eli hakukoneoptimointiin. Paikka hakukoneiden tuloslistausten kolmenkärjessä on haluttu ja tuo tunnetusti kävijöitä sivustolle. Verkkosivun sisällön suunnittelu on tärkeää siksikin, että Googlen ja Bingin kaltaiset hakukonerobotit mittaavat sisällön määrää ja laatua tehdessään listauksiaan. (Dutson 2015, 4)

Sivuston optimaalisuuteen hakukoneiden kannalta vaikuttavat:

- Sivuston sisältöä jaetaan sosiaalisessa mediassa, jossa jaot saavat lisänäkyvyyttä ja ohjaavat uusia kävijöitä sivustolle.
- Sisältö on relevanttia eli se tukee hyvin sivustolla myytävää tuotetta tai palvelua.
- Sivustolla on ulkoisille sivustoille johtavia linkkejä ja ne ovat relevantteja tuotteelle.
- Sisältö on hyvän kielipiinin mukaista ja esitetty helposti luettavasti (ei esim. liian pitkiä lauseita).
- Kuvat ja videot sekä niiden metatiedot ovat relevantteja tuotteen kannalta.
- Sisältö on originaalia eli sitä ei ole kopioitu suoraan mistään muualta.

(Dutson 2015, 4)

2.3 Projektiryhmän vuorovaikutus

Kirjassaan *Responsive Web Design* Matthew Carver kannustaa kehittäjiä ja ulkoasuunnittelijoita läheisempään ja ketterämpään yhteistyöhön (Carver 2015, 48 ja 78). Hänen mukaansa enää ei voida toimia vanhan mallin mukaan, jolloin graafikko työsti pikselin tarkkoja verkkosivu-ulkoasuja (layout) Photoshopilla viikkokausia ja koodaaja toteutti tehdyt mallit sokeasti.

Responsiivisten sivujen tulee mukautua niin moneen eri näyttökokoon etteivät tarkat visuaaliset mallit enää toimi eikä niihin kannata tuhlaa liikaa resursseja. Enemmän tulisi panostaa kehittäjän ja graafikon väliseen yhteistyöhön siten, että kehittäjä tekee ensin sivustosta skaalautuvan koodiprototyypin, joka mallintaa tulevan sivuston käyttäytymistä ja interaktiota. Graafikko puolestaan luo tyyliohjeistuksen, joka määrittää sivustolle tulevat värit, fontit, tekstuurit sekä painiketyylit. Ohjeistuksen perusteella kehittäjä tyyllittelee sivuston ilman turhantarkkoja rajoituksia, niin että tyylit skaalautuvat kauniisti kaikkiin näkymiin. Projektin onnistumisen kannalta kehittäjän ja graafikon kommunikaatio on erityisen tärkeää. Tässä auttaa esimerkiksi lyhyiden tilannekatsauksien pitäminen tarpeeksi usein, jotta varmistetaan projektin eteneminen oikeaan suuntaan. (Carver 2015, 47-61)

2.4 Käytettävyys

Mobiililaitteet kehittyvät nopeasti, joten myös niiden käyttäjät joutuvat jatkuvasti totuttelemaan uudistuneisiin käyttöjärjestelmiin, ominaisuuksiin ja laitteiden kokoihin. Kehitettäessä sivustoja mobiiliin sopiviksi on hyvä muistaa, että käyttäjät saattavat kokea jo itse älylaitteen käytön hankalaksi, puhumattakaan siitä, että he ymmärtäisivät miten suunniteltu sivusto toimii. Mobiilikehitykseen on onneksi ehtinyt muotoutua joitakin standardeja,

kuten että päänavigaatio piilotetaan usein kolmea viivaa kuvaavan ikonin alle. Nämä standardit helpottavat käyttäjiä löytämään etsimänsä sisällön sivustolta, joten näissä hyväksitututuissa käytännöissä pysyminen on suositeltavaa. (Krug 2014, 142-163)

Mobiililaitteiden ja pienten ruutujen myötä käyttäjien ajatusmalli on muokkautunut siihen, että he eivät oleta kaiken sisällön olevan heti näkökentässään. Mobiililaitteiden käyttäjät ovat tottuneet siihen, että heidän täytyy selata sivua alaspäin tai painaa ikoneita löytääkseen haluamansa tiedon. Ylimääräinen työ ei siis ole käyttäjille ongelma, jos vain sivuston käyttöliittymä antaa heille vihjeen, että haluttu tieto todellakin on löydettävissä esimerkiksi selaamalla sivua eteenpäin. (Krug 2014, 148)

Sekä Krug että Dutson (Krug 2014, 151-153 sekä Dutson 2015, 106) tuovat esille, että sivuston käyttöön liittyvät olennaiset visuaaliset vihjeet jäävät usein puuttumaan mobiilinäkymästä. Molemmat mainitsevat olennaisena esimerkkinä hover-efektin, joka työpöytänäköymässä paljastaa klikattavan kohteen hiiren osuessa kohteen päälle. Kosketusnäytöissä ei ole hiirtä, joten hover ja sen antama informaatio jäävät käyttäjälle saamatta. Krug muistuttaa myös, että responsiivinen- ja mobiili-design ovat tuoneet tullessaan entistä pelkistetyemmän ja minimalistisemmän tyyliuuntauksen Flat Designin; painikkeista jätetään varjot sekä korostukset pois sekä suositaan puhtaita värejä ja selkeitä kokonaisuuksia. Vaikka tyyli onkin modernin näköinen ja kevyt toteuttaa, liian kliininen ulkoasu voi jättää käyttäjän epäilemään esimerkiksi onko jokin elementti klikattava vai ei. Tällöin pahimmassa tapauksessa käyttäjä ei ymmärrä klikata tarvittavaa elementtiä, jää ilman haluamansa tietoa ja poistuu sivustolta turhautuneena. Tällaisten virheiden estämiseksi käytettävyydestä onkin erityisen tärkeää läpi koko kehitysprosessin (Krug 2014, 160).

3 Tekniset huomiot responsiivisessa kehityksessä

Responsiivisen sivuston teknistä toteutusta on hyvä miettiä jo sen sisältöä suunniteltaessa. Skaalautuvuus eri laitekokoihin tuo haasteita ja voi aiheuttaa yllätyksiä matkan varrella, ellei toteutusmenetelmiä ole suunniteltu etukäteen. Elementtien rakentamisen ja niiden asettelun lisäksi huomiota kannattaa kiinnittää erityisesti sivuston sujuvaan latautumiseen.

3.1 Mittasuhteet

Responsiivisessa kehityksessä teknisesti olennainen seikka on välttää tarkkoja pikselimäärisiä mittoja (Carver 2015, 100). Sääntö pätee niin elementtien leveyksiä kuin fonttikokojakin määritettäessä. Elementtien leveyksissä tulee käyttää ruudun leveyteen viittavia prosenttimääreitä eli esimerkiksi puolikkaan ruudun levyisen palstan leveydeksi annetaan 50%. Tällöin kokosuhte säilyy, oli näytön leveys sitten 320 pikseliä tai 1200 pikseliä.

Fonteissa kannattaa käyttää em-määrettä. Em perustuu selaimelle määriteltyyn perusfonttikokoon, jonka voi määritellä sivuston CSS-tyylitiedostossa. Oletuksena tämä peruskoko on usein 16px. Tällöin 1em kokoinen fonttikoko tarkoittaa 16px. Kun fonttikokoa lähdetään kasvattamaan, em-arvo suhteutetaan aina periytyvään arvoon eli tässä tapauksessa perusarvoon 16 pikseliin. Tällöin esimerkiksi otsikkofontille voitaisiin määritellä kooksi `font-size: 2em;`, mikä vastaisi 32 pikseliä. Em-arvojen ansiosta fontit skaalautuvat erikokoisille näytöille sujuvammin ja niitä on helppo muokata ilman, että joka näytölle täytyy ilmoittaa pikselin tarkat fonttikoot yksitellen. (Carver 2015, 26, 104-106)

Em-määreitä käyttäessä saa kuitenkin olla tarkkana, sillä ne periyvät ylempältä elementiltä lapsielementeille. Esimerkiksi jos elementin em-perusarvoksi on määritelty 16px ja elementillä on lapsielementti em-arvolla 0.8em ja lisäksi lapsielementillä on lapsielementti myös em-arvolla 0.8em. Tällöin sisimmän lapsielementin em-arvo ei todellisuudessa olekaan määritelty 0.8em vaan 0.64em alkuperäisestä 1em=16px arvosta, sillä lapsielementti suhteuttaa em-arvonsa ylempään elementin em-arvoon, joka tässä tapauksessa on 0.8em. Onneksi CSS3 on mahdollistanut myös ns. rem eli root em -arvon käytön. Tällöin rem-arvo suhteutetaan aina html-perustasolla määriteltyyn peruskokoon ohittaen muut fonttikoon periytyvyudet. (Dutson 2015, 61)

3.2 Rakenne

Rakenteellisesta suunnittelusta Carver mainitsee HTML5:den tukemien semanttisten elementtien käytön. Muunmuassa hakukonerobottien on helpompi lukea ja hahmottaa sivuston rakennetta, kun koodirakenteessa käytetään div-elementtien sijaan kuvaavampia peruselementtejä kuten <header>, <section> ja <aside>. (Carver 2015, 99) Hakukoneoptimoinnin lisäksi semanttisten elementtien käyttö suoraviivaistaa sivuston asettelua ja koodin lukemista ylipäätään.

3.3 Typografia

Sivuston ulkoasun yksi tärkeimpiä kulmakiviä on typografia, sillä fonteilla saa helposti persoonallista ilmettä sivustolle. Yleinen suositus on käyttää webfontteja, jotka on standardoitu toimimaan kaikilla selaimilla. Valitettavasti yleisiä webfontteja on vain muutama, joten niiden avulla ei vielä saavuteta suurta erottuvuutta ulkoasun suhteen. Sivustoille on mahdollista ladata myös omia kustomoituja fontteja tiedostomuodossa, mutta etenkin jos fontteja ja niiden kirjasinleikkauksia on paljon, sivustoa ylläpitävä verkkopalvelin kuormittuu tarpeettomasti. (Dutson 2015, 83-86)

Suositteluin ja käytetyin ratkaisu on käyttää jotakin olemassa olevista fonttipalveluista. Niissä on tarjolla laaja valikoima erilaisia webfontteja, jotka saa käyttöönsä lisäämällä sivuston head-tagin sisään aktivointikoodin sekä määrittämällä ks. fontit CSS-tiedostoihin. Koska fonttipalvelut ylläpitävät fontteja omilla palvelimillaan, isotkaan fonttitiedostot eivät kuormita sivuston palvelinta vaan ne latautuvat helposti ja nopeasti. (Dutson 2015, 91-93)

Eräitä käytettyjä fonttipalveluita ovat esimerkiksi (Dutson 2015, 91-93):

- Google Fonts
- Adobe Typekit
- Fonts.com
- Font Squirrel
- Icon Fonts

3.4 Kuvat

Kuvat aiheuttavat responsiivisessa kehityksessä ehkä eniten päänvaivaa. Yksittäiset muutaman sadan kilontavun kuvat eivät vielä aiheuta ongelmia, mutta kun näitä kuvia on useita, niinkuin nyky sivustoilla usein on, ajaudutaan vaikeuksiin. Raskaat kuvat hidastavat sivustoa dramaattisesti etenkin langattomia yhteyksiä käyttävillä mobiililaitteilla. Ongel-

maksi ei muodostu ainoastaan kuvien koko ja latautuminen vaan myös niiden responsiivisuus. Skaalautumaton kuva saattaa katketa pienellä ruudulla. Skaalautuva kuva taas saattaa kutistua todella pieneksi tai rajautua eri tavalla kuin työpöytäkokoisella näytöllä, jolloin kuvan katsomiskokemus vääristyy. (Dutson 2015, 114-117)

3.4.1 Sopiva kuvaformaatti

Olennaista kuvien optimoinnissa on valita oikea kuvaformaatti kuvan käyttötarkoitusta varten. Näyttävä valokuvaelementti vaatii tiedostotyyplitään erilaisia ominaisuuksia kuin yksivärinen ikoni. Tärkeintä on taata kuvan riittävä laadukkuus ja tarkkuus, mutta pitää samalla tiedostokoko mahdollisimman pienenä. Käytetyimpiä formaatteja ovat GIF, JPG, PNG, SVG sekä WebP. (Dutson 2015, 141-153)

GIF (Graphics Interchange Format) on vanha kuvatyyppi. Sen väriskaala tunnistaa ainoastaan 256 väriä, minkä takia se ei sovi monimutkaisten kuvien tyyppiä. Yksinkertaisille ja vain muutamaa väriä käyttäville grafiikoille sekä ikoneille GIF on kuitenkin usein riittävä tallennusmuoto. GIFin etuja ovat sen tuki kuvien läpinäkyvyyden suhteen sekä sen luomat pienet tiedostokoot. GIF on lisäksi tunnettu erityisesti hauskojen lyhyiden videoiden tai kuvasarjojen formaattina, sillä se tukee myös animoitua kuvaa. Kaikki selaimet tukevat GIFiä. (Dutson 2015, 144)

JPG on usein kameroiden oletuskuvaformaatti ja sopiikin parhaiten värikkäiden, tarkkuutta vaativien, monimutkaisten valokuvien tallennusmuodoksi. Kuvanlaatua säätämällä voi vaikuttaa JPG-kuvan tiedostokokoon. Netissä ei yleensä tarvita tarkinta mahdollista painokelpoista kuvaa, vaan JPG-kuvat kannattaa tallentaa aina vain sen kokoisena ja laatuisena kuin niitä käytetään. Kaikki selaimet tukevat JPG-formaattia. (Dutson 2015, 143)

PNG (Portable Network Format) luotiin yhdistämään GIFin ja JPGin parhaat puolet. Se tukee läpinäkyvyyttä, mutta siinä on lisäksi JPGn laaja väriskaala sekä tarkkuus. Varjopuolena PNGn hienoille ominaisuuksille on, että monimutkaisempien kuvien tallennuskoko saattaa kasvaa suureksi. PNG-kuvia kannattaakin siksi käyttää harkiten. Jos jokin muu tiedostotyyppi kuten JPG tai GIF täyttää saman tarpeen pienemmällä tiedostokoolla, kannattaa PNG korvata niillä. Joillakin selaimilla saattaa olla ongelmia käsitellä PNG-kuvien läpinäkyvyys ominaisuutta. (Dutson 2015, 146)

SVG (Scalable Vector Graphics) tallentaa kuvat vektorimuodossa. Tällöin kuvan kaikki yksityiskohdat ja sen tarkkuus säilyvät vaikka kuvaa skaalaisi rajattomasti. SVG-tyyppi sopii parhaiten graafisiin selkeisiin ikoneihin, kuten käyttöliittymän elementteihin. Esimerkiksi

Font Awesome -palvelu tarjoaa yksiväristä ikonigrafiikkaa nopeasti latautuvassa SVG-muodossa. Font Awesomen grafiikat on helppo ottaa käyttöön ja sen laajasta valikoimasta löytyvät niin tärkeimmät sosiaalisen media ikonit, erilaiset nuolet kuin variaatiot menua kuvaavasta kolmiviivaisesta symbolistakin. Carver kuitenkin varoittaa käyttämästä SVG -tiedostotyyppiä logoissa, sillä korkean kuvalaatusensa takia SVG-kuvien tiedostokoko kasvaa helposti suureksi verrattuna esim. PNG-kuviin (Carver 2015, 121-122).

Uusin kuvatiedostotyyppi on WebP ja siitä on kaavailtu JPGn, GIFn ja PNGn syrjäyttäjää. WebP tarjoaa laajan väriskaalan sekä läpinäkyvyysominaisuuden kuten PNG sekä tukee animoitua kuvaa kuten GIF (Dutson 2015, 148). Parasta WebP-kuvissa on, että ne ovat tiedostokooltaan noin 25% pienempiä kuin JPG tai PNG -formaateilla tallennetut kuvat. Pienestä koostaan huolimatta WebP-kuvien laatu säilyy kuitenkin tarkkana. Tois- taiseksi vain Chrome ja Opera -selaimet tukevat WebP-tiedostomuotoa, minkä takia sen käyttö ei ole vielä kovin suositeltavaa. Uskon kuitenkin, että monipuolisuutensa ansiosta WebP:n käyttö kasvaa rajusti, kun selaintuki ajan myötä paranee. (Google Developers 2016.)

3.4.2 Skaalautuvuus

Kuvat voi skaalata eri näyttökokoihin monella tavalla. Helpoin keino on pienentää kuva CSS-muotoilulla, jolloin kuvan leveys määritetään prosentuaalisesti suhteessa sisältöalueen leveyteen. Tällöin koko sisällön levyinen kuva olisi leveydeltään 100%. Jotta kuva ei veny, sen korkeus kannattaa määrittää automaattiseksi. (Dutson 2015, 118-120)

```
img {  
width: 100%;  
height: auto;  
}
```

Vaikka kuvaa pienentäisi tyylimääreillä, kuvan fyysinen tiedostokoko pysyy kuitenkin samana, kuin missä se on kerran ladattu palvelimelle. Jos isolle laajakuvanäytölle optimoitu kuva ladataan pienelle älypuhelimien näytölle, käytetään turhaan resursseja suuren tiedoston lataamiseen. Pienelle näytölle riittäisi kevyempi, oikeaan kokoon optimoitu kuva. Onkin suositeltavaa tarjota selaimelle aina vain sen kokoista kuvaa, joka on riittävä kyseiseen näyttökokoon. Tämän ehdollisuuden voi toteuttaa esimerkiksi tyylitiedoston media queryllä tai javascriptillä. (Dutson 2015, 118-120)

3.5 Latautuminen

Sivuston latautumisnopeudessa olennaista on selaimen ja palvelimen välillä kulkevien HTTP-kyselyiden määrä, miten raskasta tietoa kyselyiden välillä kulkee ja kuinka nopeasti niiden kuljettama tieto pystytään käsittelemään dataverkossa. Nopealla valokuitukaapelilla isotkin kyselyt latautuvat nopeasti, jolloin sivu ilmestyy käyttäjälle viipymättä. Heikommalla mobiilidatayhteydellä suuret kyselyt sen sijaan tukkivat yhteyden ja pahimmassa tapauksessa käyttäjä saa timeout-errorin yrittäessään ladata sivua. Timeout-error on selaimen antama virheilmoitus tilanteessa, jossa se ei pysty lataamaan sivustoa kohtuullisessa ajassa ja yhteys palvelimeen katkeaa. (Carver 2015, 152)

Etenkin kuvat kannattaa käsitellä paitsi oikeaan kokoon ja formaattiin, myös kompressoida ne esimerkiksi PngGauntlet, JPEGmini tai Radical Image Optimization Toolin avulla. Kompressointityökalut eivät heikennä kuvien laatua, mutta pienentävät niiden tiedostokokoja useille kymmenillä prosenteilla. (Dutson 2015, 148-154)

Myös jokaisesta koodiin erikseen linkitetystä tiedostosta lähtee erillinen kysely palvelimelle. Jos tyyli- ja skriptitiedostoja on mahdollista yhdistellä, kannattaa se tehdä. Lisäksi erilaisilla kääntäjäsovelluksilla kuten Koala-kääntäjällä on mahdollista pakata kooditiedostosta .min-version, joka vie vähemmän tilaa palvelimelta sekä latautuu nopeammin. (Carver 2015, 152)

4 Testaus

Responsiivisessa kehityksessä on monta huomioitavaa asiaa, joten huolellinen testaus nouseekin erityisen tärkeään rooliin. Testauksissa täytyy huomioida paitsi sivuston toimivuus eri päätelaitteilla, myös sivuston käytettävyys (Krug 2014, 160). Vaikka sivusto näyttäisi tabletilla juuri siltä kuin se on suunniteltu, sivuston käyttäminen ei välttämättä ole loogista.

4.1 Käytettävyystestaus

Krug neuvoo yleisesti käytettävyystutkimuksen valmistelussa. Testin pitäjät voivat katsoa samaa ruutua kuin testaaja. Ruudun voi esimerkiksi joko jakaa ruudunjako-ohjelmalla tai ajaa kaapelin avulla erilliselle näytölle, kuitenkin niin ettei se häiritse testaajaa. Testaajaa on hyvä pyytää kertomaan vapaasti ääneen tuntemuksistaan ja ajatuksistaan testattavaa tuotetta kohtaan testin aikana. Tarkentavat kysymykset kannattaa säästää testin loppuun, kun testaaja on jo suorittanut testiin liittyvät tehtävät. (Krug 2014, 160)

Mobiililaitteilla testattaessa kannattaa erityisesti huomioida, että tottumattomille testaajille jo pelkän mobiililaitteen käyttö voi olla haastavaa, mikä saattaa vaikuttaa kokemukseen testattavasta tuotteesta. Testaustilanteessa havainnoinnin kannalta voisi olla mielekäästä pitää mobiililaitte pöydällä vaakatasossa testin ajan, tällöin kuitenkin nousee ongelmaksi, ettei pöydällä lepäävä mobiililaitte ole luonnollinen tapa käyttää laitetta. Virheellinen laitteen käyttötapa voi niinkään vaikuttaa testaustulokseen. Testin seuraamista hankaloittaa myös se, että mobiililaitteen ruudulla ei näy hiirtä, joka kertoisi testin seuraajille mitä testaaja yrittää tehdä. (Krug 2014, 160)

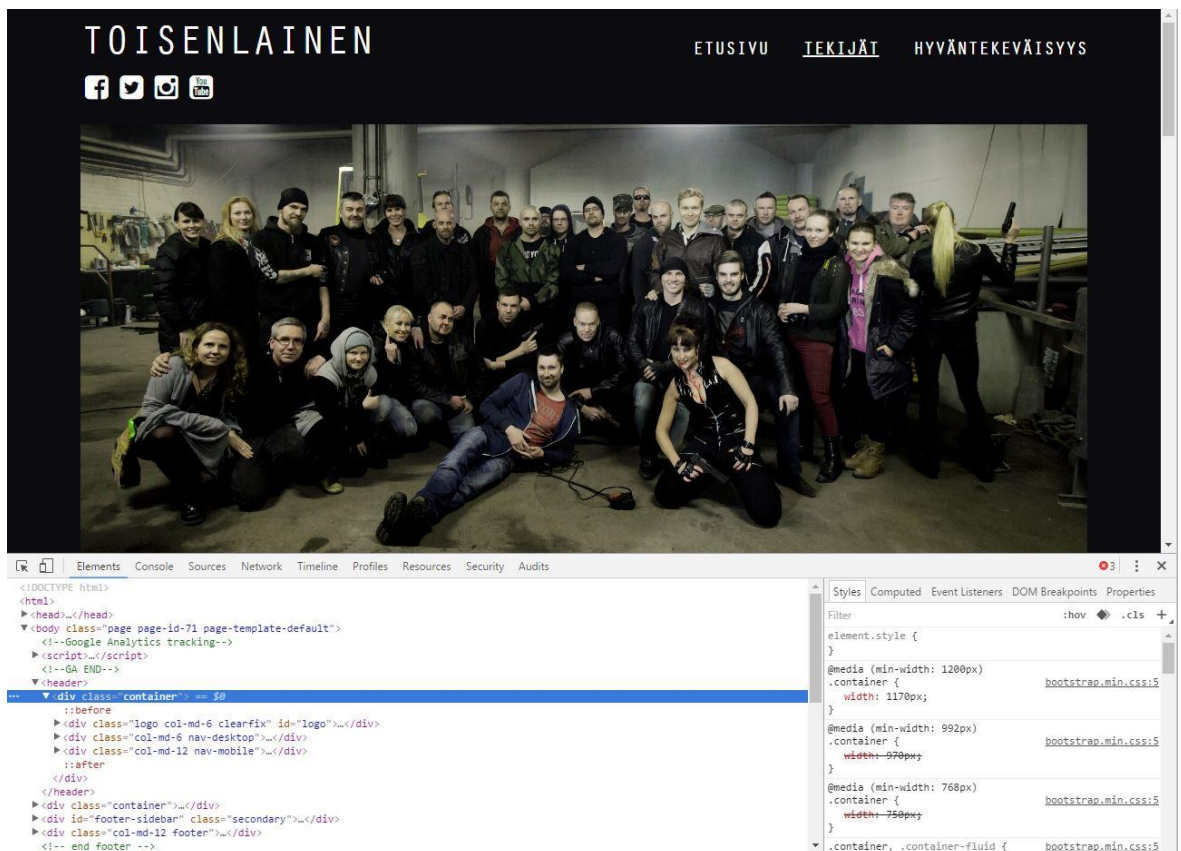
Krug suositteleeikin käyttämään mobiilitestauksessa kameraa, joka kuvaa testaajan kättä ja näyttöä mahdollisimman vakaasti. Tällöin saadaan tarkkailtua myös testaajan sormen liikettä ja mahdollista hapuilua käyttöliittymän suhteen. Krug muistuttaa että käytettävyystestaus on ollut pitkään hyvin työpöytäkonekeskeistä ja testausmenetelmät ovat kehittyneet sen mukaan. Ajan myötä käytettävyyttä testaavat työkalut kuitenkin todennäköisesti kehittyvät myös mobiilitestaukseen sopivammiksi. (Krug 2014, 160)

4.2 Toiminnallinen testaus

Responsiivisten sivustojen toimivuutta täytyy testata sekä eri näyttökoissa että eri selaimilla. Testeissä kannattaa käyttää ainakin Firefox, Chrome sekä Internet Explorer -selaimia sekä mahdollisesti Safaria. Sekä Chromesta että Internet Explorerista löytyy valmiina kehittäjätyökalut, Firefoxiin taas voi asentaa vastaavan Firebug-lisäosan. Näiden

työkalujen avulla sivustoja voi tutkia tarkemmin ja mallintaa niiden responsiivista käyttäytymistä. (Dutson 2015, 212)

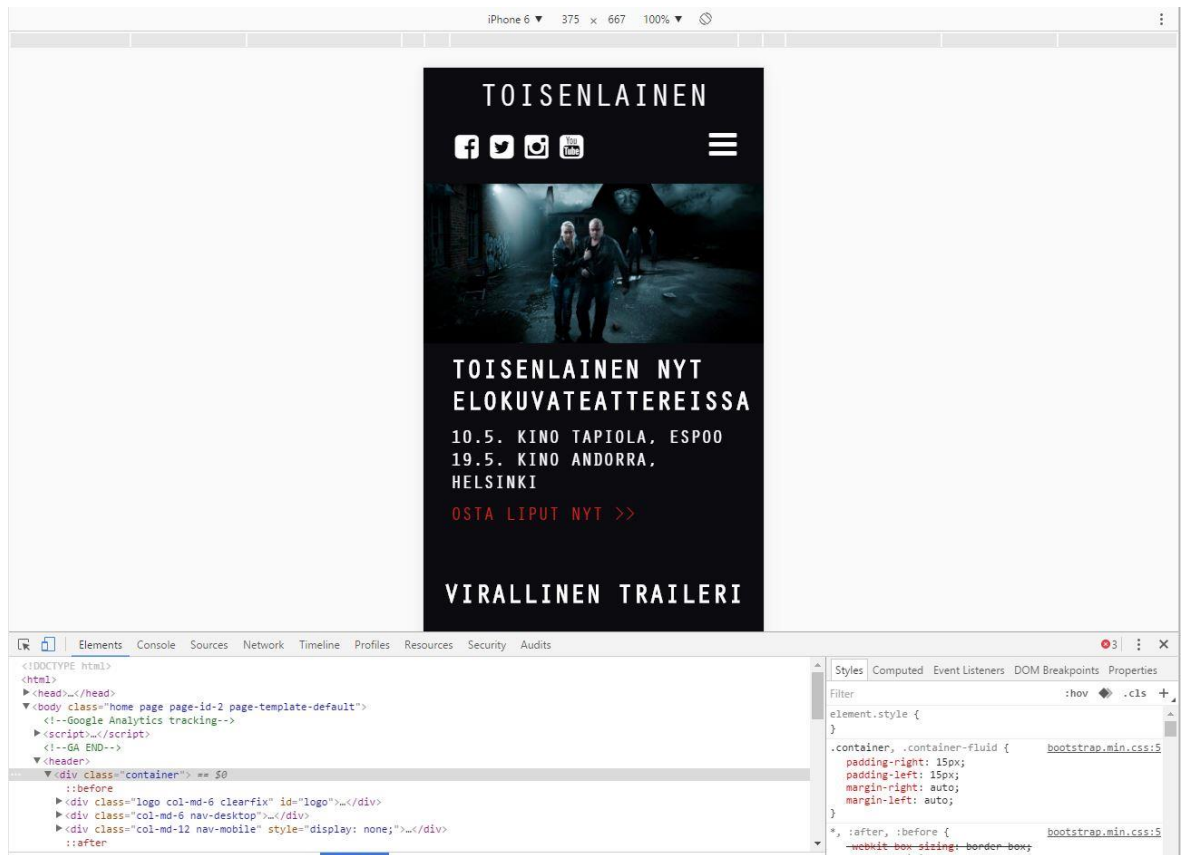
Kehitystyökalut avautuvat selaimen yhteyteen. Toisessa ikkunassa ne näyttävät sivuston koodirakenteen. Toisessa ikkunassa elementtien arvoja voi muuttaa ja testata eri vaihtoehtoja. Muutokset eivät ole pysyviä vaan katoavat selaimen päivittyessä. Työkaluilla voi myös tarkastella selaimen ja palvelimen välillä kulkevia kyselyjä ja ladattavia tiedostoja. (Dutson 2015, 213) Kehittäjätyökalut ovat helppo ja nopeakäyttöinen apuväline niin joka päiväisessä kehitystyössä kuin virhetilanteiden selvittelyssäkin. Kuvassa 1 esimerkki Chromen kehittäjätyökalunäkymästä.



Kuva 1. Chromen kehittäjätyökalut (Kuvakaappaus: toisenlainenelokuva.com)

Ennen oli tapana skaalata selaimia pienemmäksi ja tutkia siten sivuston responsiivisuutta kehittäjätyökaluilla. Tämä oli melko epätarkka työskentelytapa. Nykyään kehittäjätyökaluissa on responsiivisuuspainike, jonka kautta voi säätää ja testata responsiivisia näkymiä. Työkalu tietää suosituimpien mobiililaitteiden mitat valmiiksi ja siinä voikin valita vaikka Samsung S5 -puhelimien resoluution vaakasuunnassa. Näin lähes mikä tahansa mobiililaitte on kehittäjän saatavilla ja tuotetta voidaan siten testata tarkasti. (Dutson 2015, 216)

Kuvassa 2 on Chrome selaimen responsiivinen kehitysnäkymä iPhone 6 -puhelimella.



Kuva 2. Chromen responsiivinen kehitysnäkymä (Kuvakaappaus: toisenlainenelokuva.com)

Vaikka selainten testaustyökalut ovat kehittyneitä ja mahdollistavat eri mobiililaitteiden mallinnuksen, sekä Carver, Dutton että Krug korostavat etteivät työkalut kuitenkaan korvaa oikeilla päätelaitteilla tapahtuvaa testausta. Kaikkia mahdollisia mobiililaitteita ei tietenkään aina ole saatavilla. Carver suosittelee, että kattavan testituloksen saamiseksi responsiivisuutta kannattaisi testata mahdollisimman monella seuraavista laitteista (Carver 2015, 146):

- Vanhalla ja uudella Android-puhelimella
- Vanhalla ja uudella iPhonella
- Windows-puhelimella
- iPadilla
- Android-tabletilla

5 WordPress julkaisualustana

WordPress on avoimen lähdekoodin julkaisualusta, joka on alun perin suunniteltu blogikirjoittamista varten. WordPress on luotu Php ja SQL -kielillä ja se käyttää Gnu Public Licensiä. Alkunaan WordPress sai vuonna 2003 (Wordpress.org 2016.) WordPressin alkuperäinen tarkoitus oli tuottaa yksinkertainen käyttöliittymä, jolla voisi kirjoittaa tekstejä internetiin, julkaista valmiita kirjoituksia ja muotoilla julkaistua sisältöä yksinkertaisesti helposti luettaviksi näkymiksi. Järjestelmä on kuitenkin tästä sittemmin monimutkaistunut niin, että se mahdollistaa useine muunnelmineen mm. erilaisia hakutoimintoja, valinta- ja järjestelymahdollisuuksia sekä sisältöön liitettävän metadatan käytön. (Damstra, Stern & Williams 2010, 1-8)

WordPressille tunnusomaista on laaja, globaali ja eläväinen yhteisö sekä sen mahdollistama kehitys ja tuki. Avoimen lähdekoodin järjestelmässä kaikki voivat osallistua järjestelmän parantamiseen ja kehittämiseen. Samalla huonot tai toimimattomat ominaisuudet ja lisäosat karsiutuvat pois, kun yhteisö raportoi niistä julkisesti ja vaatii korjauksia. Kun kaikki pääsevät puuhailemaan vapaasti avoimen koodin kanssa, myös tukea löytyy helposti muilta yhteisön jäseniltä, jotka ovat jo painineet saman ongelman kanssa ja selättäneet vaikeudet. Juuri näiden ominaisuuksien takia WordPress kasvaa ja kehittyy jatkuvasti yhteisönsä mukana ja sen tarpeita kuunnellen. (Damstra, Stern & Williams 2010, 1-8) BuiltWith-sivuston mukaan peräti 50% jotakin julkaisujärjestelmää käyttävistä verkkosivustoista käyttää juuri Wordpressiä, mikä tekee siitä ylivoimaisesti suosituimman sisällönhallintajärjestelmän (BuiltWith 2016).

5.1 Miksi WordPress valittiin juuri tähän projektiin?

Alusta asti minulle ja toimeksiantajalleni oli selvää, että haluamme käyttää tässä projektissa Wordpressiä. Se oli meille molemmille entuudestaan tuttu järjestelmä, olimmehan työskennelleet sisällönhallinnan parissa ja myös tehneet WordPress-pohjaisia sivuja itse. Olin viimeksi työharjoittelussani päässyt kehittämään useampiakin yrityskäyttöön tulleita WordPress-sivustoja ja halusin myös siksi päästä soveltamaan itsenäisesti näissä projekteissa oppimiani tekniikoita.

WordPressissä oli muutenkin monia projektin kannalta olennaisia etuja (Damstra, Stern & Williams 2010, 317-320):

- WordPress on maksullisia lisäosia lukuunottamassa täysin ilmainen, mikä sopii erityisen hyvin tämän projektin kaltaiseen harjoitustyöhön.
- WordPress tarjoaa helppokäyttöisen sisällönhallintapaneelin, joten minun kehittäjänä on helppo luovuttaa valmis tuote asiakkaalleni ja hän voi päivittää sivustoa itse.
- Monet valmiit teemat, lisäosat ja laajat yhteisön luomat ohjefoorumit mahdollistavat verkkosivujen teon tarvittaessa nopeallakin aikataululla.
- Sivustolla on hyvät jatkokehitysmahdollisuudet, sillä WordPress tukee alisivustoja sekä mm. sosiaalisen median ja RSS-tietovirtojen käyttöä.

6 Projektin kulku

Kehitysprojekti alkoi tammikuussa 2016 suunnittelukokouksella. Työhön kuului sivuston suunnittelu alusta asti, tekniikoiden kartoittaminen, ulkoasusuunnittelu, tekninen toteutus, sisällöntuotanto sekä lopulta ylläpito. Sivusto julkaistiin maaliskuun vaihteessa.

6.1 Kehitysympäristö

Rakensin sivustoa omalla kannettavalla tietokoneellani, johon asensin paikalliseksi kehitysympäristöksi WAMPin. Koodieditorina käytin Komodo Editiä ja tiedostojen kääntämiseen sekä pakkaamiseen Koala-kääntäjää.

WAMP on ohjelmistokokonaisuus, joka on mukailtu Linux-pohjaisesta LAMPista. Molemmat ohjelmistokokonaisuudet sisältävät Apache-palvelimen, MySQL-tietokantarajapinnan sekä PHP-, Perl- ja Python-komentosarjakielen (WampServer 2016.). WAMPin ero LAMPiin on se, että siihen sisältyy Windows-käyttöjärjestelmä siinä missä LAMPiin kuuluu Linux. LAMP on enemmän suosittu virtuaalipalvelimien ympäristönä, kun taas monet suosivat WAMPia nimenomaan paikallisena kehitysympäristönä verkkosivujen teossa.

WAMP on helppo ladata ja asentaa koneelle. Se luo haluttuun hakemistoon kansiorakenteen, jossa voi muokata asetuksia ja verkkosivun rakennetta. Itse työskentelin WAMPilla siten, että pudotin WAMPin www-kansioon WordPressin tiedostot ja tein WAMPin MyPhpAdminilla tietokannan. Tämän jälkeen pystyin simuloimaan oikeaa WordPress-asennusta omalla koneellani. WAMP täytyy aina käynnistää sen sisältämällä wampmanager.exe -ohjelmalla, minkä jälkeen simulaatio onnistuu. Mielestäni WAMPissa on parasta juuri sen selkeä käyttöliittymä kansiorakenteineen. Tiedostojen muokkaus ja siirtely toimii kuin minkä tahansa kansioden ja tiedostojen kanssa.

6.2 Sivuston suunnittelu ja ulkoasu

6.2.1 Tarpeet

Aloitin sivuston suunnittelun kokouksella toimeksiantajan kanssa. Kävimme läpi mitä ajatuksia meillä molemmilla oli sivuston tarpeista ja niiden prioriteeteista. Liitteessä 1 on suunnittelukokouksessa laadittu taulukko, jonka pohjalta lähdin toteuttamaan sivustoa. Taulukon ensimmäiseen sarakkeeseen on määritelty asiat, joita sivulla täytyi olla. Toiseen sarakkeeseen on lisätty huomautuksia ja muistiinpanoja myöhempiä kokouksia varten. Kolmanteen sarakkeeseen on merkitty asioita, joita sivustolla voisi olla, mutta jotka eivät ole ainakaan alkuvaiheessa välttämättömiä. Viimeiseen sarakkeeseen keräsimme muutamia olemassa olevia sivustoja referensseiksi. Osasta referenssisivustoista oli tarkoitus ottaa mallia, osa taas oli kerätty varoittaviksi esimerkeiksi.

6.2.2 Rautalangat

Kokouksen pohjalta mietin sivuston sisältöä ja sen asettelua sekä työstin sivustosta rautalankamallit. Piirsin alustavan hahmotelman ensin paperille ja tein sitten siitä version Photoshopilla, jotta elementtien asettelu ja siirtely paikasta toiseen olisi helpompaa. Piirsin Photoshopilla ensin työpöytänäkymän, koska siitä oli mielestäni helpoin lähteä liikkeelle. Tarkoitukseni oli saada nopeasti jotain kättä pitempää, jonka avulla voisin jatkaa keskustelua toimeksiantajan kanssa. Aiempien kokemusteni pohjalta pidin kokoajan mielessäni, miten elementit tulisivat skaalautumaan mobiilinäkymään. Pelkistetty rautalankamalli löytyy liitteestä 2.

Olen aikaisemmissa projekteissa huomannut, että asiakkaan on joskus vaikea saada rautalankavedoksista konkreettista käsitystä tulevasta sivustosta. He eivät välttämättä osaa visualisoida mielessään lopullista sisältöä karkeisiin malleihin. Siksi tein rautalankamallista sekä työpöytä- että mobiiliversiot, joissa käytin sivustolle suunniteltua värimaailmaa ja muutamia mallintavia kuvia. Tyylliteltyt rautalankamallit löytyvät liitteistä 3 ja 4. Mallien ei ollut tarkoitus olla valmiita sivuston layouteja, vaan edustaa konkreettisempaa tunnelmaa ja käyttökokemusta, jota sivuston kanssa lähdeittäisiin tavoittelemaan. Kun seuraavassa kokouksessa esittelin mallit toimeksiantajalle, keskustelu oli niiden avulla helppoa ja tiesimme olevamme samalla aaltopituudella projektin suhteen.

6.2.3 Ulkoasu

Minun oli tarkoitus tehdä rautalankamallien pohjalta lopulliset sivustolayoutit, sitten kun saisin toimeksiantajalta tarkempaa tietoa toivotuista väreistä, fonteista ja kuvista. Tiedossa oli, että toimeksiantajan graafikkotuttava oli työstämässä julistetta elokuvaa varten ja odotin, että voisin kopioida ulkoasuun tyyliä siitä.

Koska tätä projektia tehdään vapaa-ehtoisesti ja ilman korvausta, toisinaan arki ja leipätyö täytyy laittaa aikatauluissa edelle. Julisteen tekijälle osui kiireitä enkä ehtinyt saada häneltä lopullista julistetta ajoissa. Minulla ja toimeksiantajalla oli kuitenkin vahva käsitys sivuston toivotusta tunnelmasta. Sivustosta haluttaisiin tummapohjainen, jolloin tekstien olisi oltava vaaleita. Pääosassa sivulla olisivat visuaaliset kuvat ja trailerivideo, jotka loisivat ilmeeseen loppusilauksen. Lisäksi toimeksiantaja oli määrittellyt viralliseksi fontiksi Orator STD:n. Näillä tiedoilla aloitin ulkoasun kokoamisen suoraan koodiin. Määrittelin fontin, valitsin taustaväriksi siniseen vivahtavan mustan, tekstien väriksi valkoisen ja huomioväriksi vereen viittaavan punaisen, jota olin nähnyt käytettävän julisteen mallivedoksessa.

Loppu olikin tekemistä ja testausta. Koska tein itse sekä ulkoasun suunnittelun että sen koodaamisen, tyylejä oli helppo muuttaa mikäli huomasin, ettei jokin toiminut. Vaihdoin esimerkiksi huomiovärin sävyä useankin otteeseen, kun näin ettei liian tumma punainen erottunut tarpeeksi. Lisäsin myös pohjaväriin lisää sinisyyttä tuomaan enemmän kontrastia. Lisäksi muuttelin fonttien kokoa sen perusteella, mikä mielestäni toimi parhaiten milläkin päätelaitteella. Valmis lopputulos on esitetty liitteessä 5.

6.3 Koodin rakentaminen

Käytin projektin pohjana WordPressin valmista teemaa, jossa ei ollut aluksi mitään koodirakennetta. Teema sisälsi ainoastaan paikat tyhjälle otsakkeelle (header) ja alatunnisteelle (footer). Koodasin etusivun koodirunkoon osioita aluksi tyhjiillä section-elementeillä, jotta sain sivulle jonkinlaisen alustavan järjestyksen. Seuraavaksi asettelin paikoilleen headerin ja footerin elementteineen ja annoin niille tyyliä, jotta ne asettuisivat suunnitelmieni mukaan.

Enemmän huomiota vaativien elementtien koodaus eteni niin, että tein kaikkiin osioihin jonkinlaisen alustavan pohjan. Sitten aloitin siitä, minkä toteutuksesta minulla oli selkein käsitys ja etenin elementti kerrallaan. Haastavimmat osat olivat karuselli ja näyttelijät-osio.

Niitä täytyi palata viilaamaan vielä aivan viimeisinäkin päivinä ennen julkaisua. Tarkoitukseni sivuston rakentamisessa oli tehdä mahdollisimman paljon itse ja minimoida valmiiden elementtien ja lisäosien käyttö.

6.4 Käytetyt tekniikat

6.4.1 WordPress-teema

Projektissa on käytetty pohjana WP Speed code Blank Theme -nimistä WordPress-teemaa (Ordinarycoder.com 2016.), joka sisältää teeman rakenteen headereineen ja footereineen sekä valmiiksi asennetun jQueryn, Bootstrapin, Font Awesomen ja Less:in.

Vaikka teeman tiedostorakenne on valmiina, itse tiedostot kuten header ja single.php ovat käytännössä tyhjiä, mikä jättää koodaajalle vapaat kädet kustomoida teemaa rajattomasti tarpeidensa mukaan.

Halusin käyttää projektissani tätä teemaa juuri siksi, että se antaa sivustolle nopeasti hyvän pohjan, mutta ei rajoita sen rakennetta millään tavalla. Tutustuin teemaan alunperin työharjoitteluni aikana sillä ohjaajani käytti ks. teemaa sivustokehityksessä. Harjoitteluni aikana teemaa käytettiin useissa sivustoprojekteissa ja työskentely sen kanssa osoittautui helpoksi. Ennen opinnäytetyöprojektia käytin teemaa myös oman portfoliosivustoni pohjana, joten teema oli varsin hyväksi havaittu.

6.4.2 WordPress-lisäosat eli pluginit

Halusin käyttää sivustolla mahdollisimman vähän valmiita lisäosia. Kokemukseni mukaan jos sivustolla on paljon lisäosia, ne saattavat sotkea toistensa toimintaa ja niitä päivitettäessä seuraukset voivat olla arvaamattomat. Valmiilla sivustolla on käytössä 5 lisäosaa: WordFence Security, Maintenance Mode, Lazy Load, Yoast SEO sekä Advanced Custom Fields.

WordFence Security -lisäosa hoitaa sivuston tietoturvaa ja ilmoittaa mm. epäilyttävästä liikenteestä sivulla. Lazy Load optimoi kuvien latautumista jQueryn avulla. Sen ansiosta yksittäinen kuva latautuu vasta kun se ilmestyy käyttäjän näkymään, mikä nopeuttaa koko sivuston latautumista. Advanced Custom Fields automatisoi kustomoitujen sisältötyyppien käsittelyä. Yoast Seo puolestaan auttaa hakukoneoptimoinnissa. Sen avulla artikkeleille ja julkaisuille voi määrittää avainsanoja ja kuvauksia, jotka nostavat koko sivuston hakukonelöydettävyttä.

Maintenance Mode -lisäosa oli käytössä, kun sivustoa ei ollut vielä julkaistu, mutta minun sekä toimeksiantajan piti päästä päivittämään tietoja sivustolle. Maintenance Mode esittää kirjautumattomalle käyttäjälle staattisen sivun, joka kertoo sivuston olevan huoltotilassa. Kirjautuneet käyttäjät voivat puolestaan käyttää sivustoa normaalisti. Olisin voinut poistaa Maintenance Mode -lisäosan sivustolta julkaisun jälkeen, mutta mielestäni se on hyvä olla sivustolla valmiina mahdollisten ongelmatilanteiden tai päivitystöiden varalta. Jos sivusto joudutaan yhtäkkiä sulkemaan käyttäjiltä virhetilanteen takia, riittää että aktivoi jo valmiiksi asennetun lisäosan.

6.4.3 Custom Post Type

Mielestäni sisällönhallintajärjestelmän tärkein tehtävä on mahdollistaa se, että kenen tahansa on helppo päivittää sivustoa WordPressin käyttöliittymän kautta. Alusta asti oli selvää, että elokuvan sivusto tulee sisältämään toistuvia elementtejä kuten näyttelijäkuvauksia, joiden määrä voi vaihdella ja joita täytyy päästä muokkaamaan helposti. Keskusteltuani toimeksiantajan kanssa päätimme, että näyttelijöiden kuvaukset ovat aina rakenteeltaan samanlaisia, sisältäen kuvan, näyttelijän nimen, roolihahmon nimen sekä lyhyen kuvaustekstin.

WordPressissä on oletuksena viisi erilaista sisältötyyppiä, joita voi hallinnoida ohjausnäytössä:

- Artikkelit (Post)
- Sivut (Page)
- Liitteet (Attachment)
- Korjausversio (Revision)
- Navigaatio menu (Navigation menu)

WordPress kuitenkin antaa mahdollisuuden luoda myös omia kustomoituja sisältötyyppejä (Custom Post Type). Custom Post Type määritetään `functions.php` -tiedostossa `register_post_type()` -funktion avulla. WordPressiin on myös ladattavissa lisäosia (plugin), joiden avulla sisältötyyppejä voi rakennella graafisen käyttöliittymän avulla. Tällä sivustolla hyödynsin osittain molempia tekniikoita.

Latasin sivustolle Advanced Custom Fields -lisäosan helpottamaan työskentelyä. Lisäosa tarjoaa suoraan sisällönhallintajärjestelmään graafisen käyttöliittymän, jossa voi luoda uuden sisältötyypin ja valita siihen halutut kentät sekä ominaisuudet. Itse tein kuitenkin niin, että vain aktivoin Advanced Custom Fields -lisäosan, jolloin se käsittelee määriteltyjä si-

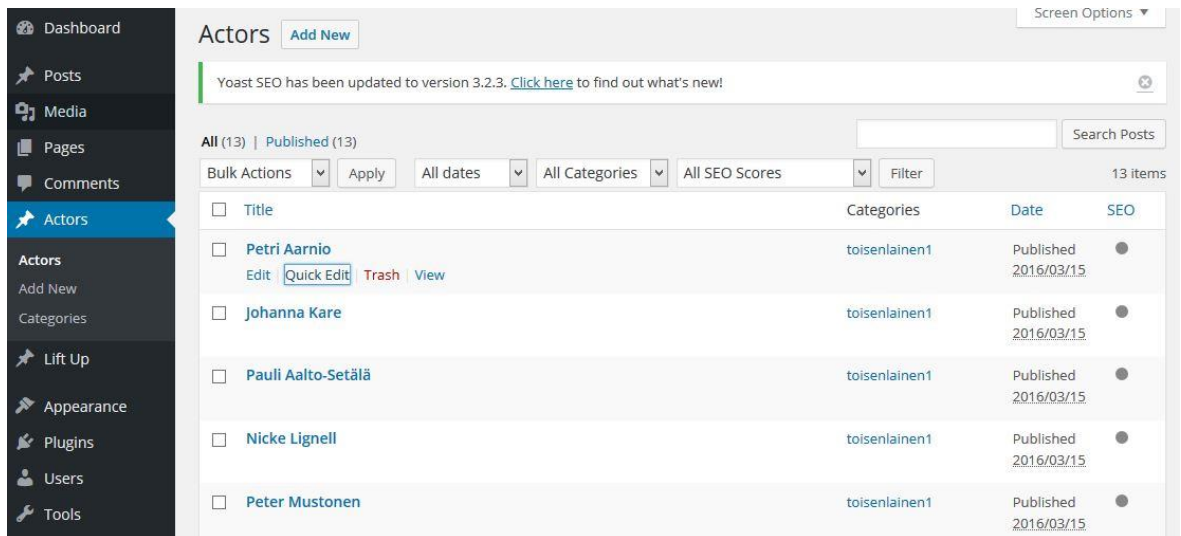
sältötyyppejä automaattisesti. Varsinaiset sisältötyypin määrittelyt tein functions.php -tiedostoon, sillä se oli mielestäni konkreettisempi tapa. Advanced Custom Field -lisäosa kuitenkin pelasti minulta aikaa ja vaivaa, ettei minun tarvinnut kirjoittaa koodiin kaikkia Custom Post Type -käsittelylauseita.

Näyttelijät-sisältötyyppi näyttää functions.php- tiedostossa tältä:

```
add_action( 'init', 'create_post_type' );
function create_post_type() {
    register_post_type( 'actors',
        array(
            'labels' => array(
                'name' => __( 'Actors' ),
                'singular_name' => __( 'Actor' ),
            ),
            'taxonomies' => array('category'),
            'public' => true,
            'has_archive' => true,
            'rewrite' => array('slug' => 'actors'),
            'supports' => array(
                'title',
                'editor',
                'excerpt',
                'thumbnail',
                'custom-fields',
                'revisions'
            ),
        )
    );
}
```

Koodista voi lukea, että määritellyn sisältötyypin nimi on Actors ja yksittäisen näyttelijäelementin nimi Actor. Elementtiin on määritelty title, editor, excerpt, thumbnail, custom-fields sekä revision -kentät.

Ohjausnäkyvässä Näyttelijät-elementti muistuttaa tavallista artikkelia. Kuvasta 3 näkyy, että Actors-sisältötyyppi on WordPressin ohjausnäkyvässä vasemmalla listauksessa muiden sisältötyyppien kanssa. Actors-välilehdellä näkyvät kaikki lisätyt näyttelijät (actor). Välilehdeltä voi myös lisätä uusia näyttelijöitä (Add New) tai muokata nykyisiä samalla tavalla kuin tavallisia WordPressin artikkeleita muokataan.



Kuva 3. Näyttelijät-listaus (Kuvakaappaus: toisenlainenelokuva.com. 2016)

Yksittäisen näyttelijän muokkausnäkyminen kuvassa 4 näyttää samalta kuin tavallisen WordPress-artikkelin muokkausnäkyminen. Otsikkokenttään syötetään näyttelijän nimi, excerpt- eli ingressikenttään tulee roolihahmon nimi, sisältökenttään kuvausteksti ja featured imageen täytetään kuva näyttelijästä. Olisin voinut muokata kenttien nimet kuvaavammiksi, esimerkiksi excerpt-kentän nimi olisi tässä tapauksessa roolihahmo. Nimeäminen ei ollut kuitenkaan nyt ensisijainen asia, koska vain minä ja toimeksiantaja muokkaamme sivustoa ja tiedämme mitä eri kohtiin tulee täyttää. Nimeämisen jättäminen väliin oli siis tietoinen valinta ja saatan korjata nimet myöhemmin.

The screenshot shows the WordPress 'Edit Post' interface for a post titled 'Johanna Kare'. The left sidebar contains navigation options: Dashboard, Posts, Media, Pages, Comments, Actors (highlighted), and a sub-menu for Actors including Add New, Categories, Lift Up, Appearance, Plugins, Users, Tools, Settings, Custom Fields, SEO, and Collapse menu. The main content area has a title field with 'Johanna Kare', a permalink field with 'http://toisenlainenelokuva.com/actors/johanna-kare/', and an 'Add Media' button. Below is a rich text editor with a Paragraph block containing two paragraphs of text. The first paragraph describes Johanna as a photography enthusiast and introverted person. The second paragraph describes her role as Mari in a film. Below the editor is a word count of 35 and a note that the post was last edited on March 29, 2016. An 'Excerpt' field contains the word 'Mari'. The right sidebar features a 'Featured Image' section with a photo of Johanna Kare and a 'Publish' section with options for 'Preview Changes', 'Status: Published', 'Visibility: Public', 'Revisions: 6', 'Published on: Mar 15, 2016 @ 20:29', and buttons for 'Move to Trash' and 'Update'.

Kuva 4. Näyttelijä-sisältötyypin muokkausnäkyminen (Kuvakaappaus: toisenlainenelokuva.com. 2016)

Toinen kustomoitu sisältötyyppi on etusivun karuselli ja siihen tehtävät nostot eli lift upit. Nostoilla on tarkoitus korostaa tärkeää sisältöä kuten tietoa ensi-illasta tai yhteistyökumppaneista. Jos nostoja on julkaistu enemmän kuin yksi, karuselli esittää yhden noston kerrallaan. Nostoon liittyy aina kuva ja otsikko. Lisäksi nostoon voi lisätä tarvittaessa linkin sekä lisää tekstiä. Linkin tarkoitus on, että esimerkiksi elokuvanäytöksistä kertova nosto voidaan ohjata suoraan ulkoiselle lipunmyyntisivustolle. Nostoja voi lisätä, poistaa ja muokata helposti ohjausnäkyvän kautta ja julkaistut nostot esitetään automaattisesti karusellissa.

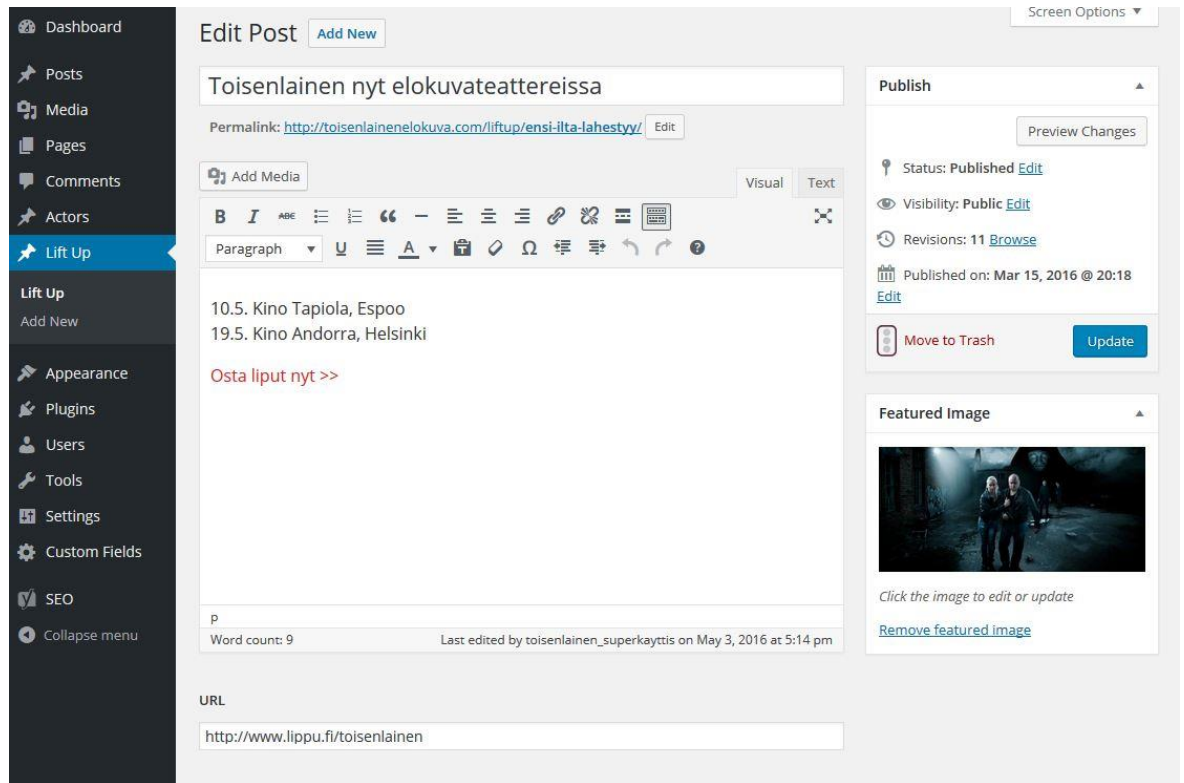
Loin nostoista uuden sisältötyypin samalla tavalla kuin näyttelijöistä. Kuvassa 3 Lift Up -niminen sisältötyyppi näkyy WordPressin ohjausnäkyvässä Actors-tyypin alla kuten muutkin sisältötyypit. Nostot-sisältötyypin luontilauseet functions.php-tiedostossa näyttävät myös samalta kuin Näyttelijät-elementin luontilauseet:

```
add_action( 'init', 'create_post_type' );
function create_post_type() {
    register_post_type( 'liftup',
        array(
            'labels' => array(
                'name' => __( 'Lift Up' ),
                'singular_name' => __( 'Lift Up' ),
            ),
            'public' => true,
            'has_archive' => true,
            'rewrite' => array('slug' => 'liftup'),
            'supports' => array(
                'title',
                'editor',
                'excerpt',
                'thumbnail',
                'custom-fields',
                'revisions'
            ),
        )
    );
}
```

Nostoihin tarvittiin kuitenkin myös kenttä linkkiä varten. Määritin kentän functions.php-tiedostossa `register_field_group()` -funktiolla. Määritelty kenttä on tyypiltään tavallinen tekstikenttä. Sille on määritelty kuvaava otsikko URL. Lisäksi kentälle on määritelty, että se esiintyy vain liftup-sisältötyyppien yhteydessä. Määritys näyttää tältä:

```
if(function_exists("register_field_group"))
{
    register_field_group(array (
        'id' => 'acf_lift-up-target-url',
        'title' => 'Lift Up target URL',
        'fields' => array (
            array (
                'key' => 'field_56d6c0969c9d8',
                'label' => 'URL',
                'name' => 'target_url',
                'type' => 'text',
                'post_type' => array (
                    0 => 'all',
                ),
                'allow_null' => 0,
                'multiple' => 0,
            ),
        ),
        'location' => array (
            array (
                array (
                    'param' => 'post_type',
                    'operator' => '==',
                    'value' => 'liftup',
                    'order_no' => 0,
                    'group_no' => 0,
                ),
            ),
        ),
        'options' => array (
            'position' => 'normal',
            'layout' => 'no_box',
            'hide_on_screen' => array (
            ),
        ),
        'menu_order' => 0,
    ));
}
```


Kuva 5 esittää nostojen muokkausnäkömön WordPressin hallintapaneelissa. Näkömön muistuttaa minkä tahansa artikkelin muokkausnäkömön, mutta alalaidassa oleva URL-kenttä on määritelty vain nostoille. URL-kenttään lisätään linkki, jonne nostoa napsauttamalla pääsee, otsikko-kenttään syötetään noston tärkein sanoma eli otsikko, tekstikenttään voi antaa lisätietoja kuten kuvassa 5 olevat näytösajat. Featured image -kenttään annetaan nostoon liittyvä kuva.



Kuva 5. Lift Up-nostojen muokkausnäkömön (Kuvakaappaus: toisenlainenelokuva.com. 2016)

Nostot kootaan automaattisesti etusivun karuselli-elementtiin, joka käyttää Owl Carousel -nimistä Javascript-lisäosaa. Karusellin toiminnasta lisää tietoa edempänä javascript-luvussa.

6.4.4 Bootstrap

Bootstrap on HTML:stä, CSS:stä ja javascriptista koostuva viitekehys(framework), joka on tarkoitettu responsiivisten sivustojen ja mobiiliapplikaatioiden kehitykseen. Omien sivujensa mukaan se on tämän hetken käytetyin koodikehikko ks. tarkoitukseen (getbootstrap.com 2016.). Suosionsa ja laajan käyttöasteensa takia minulle oli melko itseselvää valita Bootstrap tähän projektiin.

Bootstrap määrittää selainikkunan 12 pystysuuntaiseen sarakkeeseen (column). Elementteille voi antaa luokka-määreillä (class) tiedon, kuinka monen sarakkeen leveyttä ne voivat käyttää. 12 sarakkeen leveys tarkoittaa täysleveää elementtiä, 6 sarakkeen leveys puolikkaan ruudun kokoista elementtiä, 3 saraketta 1/4 näytön levyistä elementtiä ja niin edelleen.

Bootstrap jakaa näyttöjen leveydet neljään kategoriaan: isoihin ja tavallisiin tietokoneen näyttöihin, tabletteihin sekä puhelimiin. Taulukossa 1 näkyvät näyttöjen pikselikoot, minkä tyyppiset laitteet käyttävät ks. kokoisia näyttöjä sekä mikä Bootstrapin luokka-määre niitä koskee.

Taulukko 1. Bootstrapin leveysluokat

Luokka-määre	Selaimen leveys	Laite
.col-xs-	<768px	Puhelimet
.col-sm-	>768px	Tabletit
.col-md-	>992px	Tavalliset näytöt
.col-lg-	>1200px	Laajakuvanäytöt

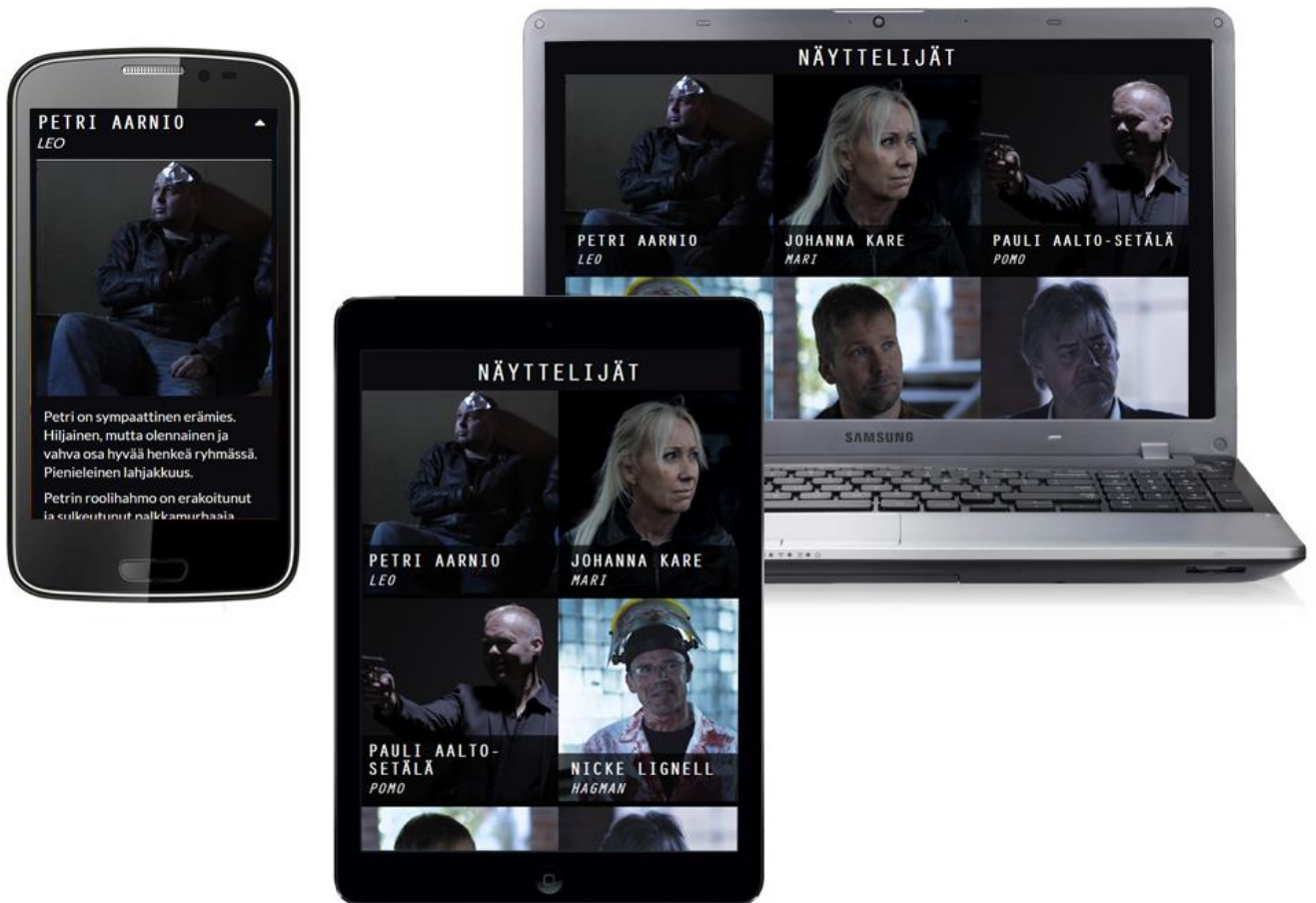
Hyödynsin Bootstrapia esimerkiksi näyttelijät-elementin asettelussa. Koodi näyttää tältä:

```
<div class="actor col-xs-12 col-sm-6 col-md-6 col-lg-4">
  <div class="actor-preview">
    <h2><?php the_title();?></h2>
    <h3><?php the_excerpt();?></h3>
    <i class="fa fa-caret-up actor-slide"></i>
  </div>

  <?php echo get_the_post_thumbnail();?>
  <div class="actor-info">
    <h2><?php the_title();?></h2>
    <h3><?php the_excerpt();?></h3>
    <p><?php the_content(); ?></p>
  </div>
</div>
```

Ensimmäisen rivin luokka-määreistä huomaa, että isolla näytöllä yksi näyttelijä vie neljä sarakkeen leveyttä eli 1/3 sivun leveydestä (col-lg-4), jolloin näyttelijöitä asettuu 3 rinnan. Pienemmällä näytöllä yksi näyttelijäesittely vie puolet näytöstä (col-md-6, col-sm-6), jolloin näyttelijöitä on rinnakkain kaksi. Pienimmällä näytöllä yksi näyttelijä (col-xs-12) käyttää koko 12 saraketta eli täyden ruudun leveyden. Tämä on yksi tapa skaalata elementtejä

mobiiliin ja työpöytäkoon välillä Bootstrapia hyödyntäen. Kuva 6 esittää, miltä Näyttelijät-
osio näyttää erikokoisilla näytöillä.



Kuva 6. Näyttelijät-osio eri laitteilla (Projektin ohessa syntynytä kuvituskuva)

Bootstrapin suosion takana on sen kyky tehdä responsiivisesta ulkoasukehityksestä helppoa ja nopeaa. Laajan käyttöasteensa ansiosta Bootstrap on oletuksena useiden valmiiden sivustoteemojen pohjana (startbootstrap.com 2016.). Teemasta sitten riippuu kuinka paljon sivuston rakennetta on tehty valmiiksi tai kuinka paljon kehittäjän täytyy itse koodata elementtejä paikoilleen. Tässä projektissa käyttämässäni teemapohjassa Bootstrap oli asennettu valmiiksi, mutta mitään valmista koodirakennetta ei ollut. Sain Bootstrapin elementit kuitenkin heti käyttöni, kun aloin rakentamaan sivustoa puhtaalta pöydältä.

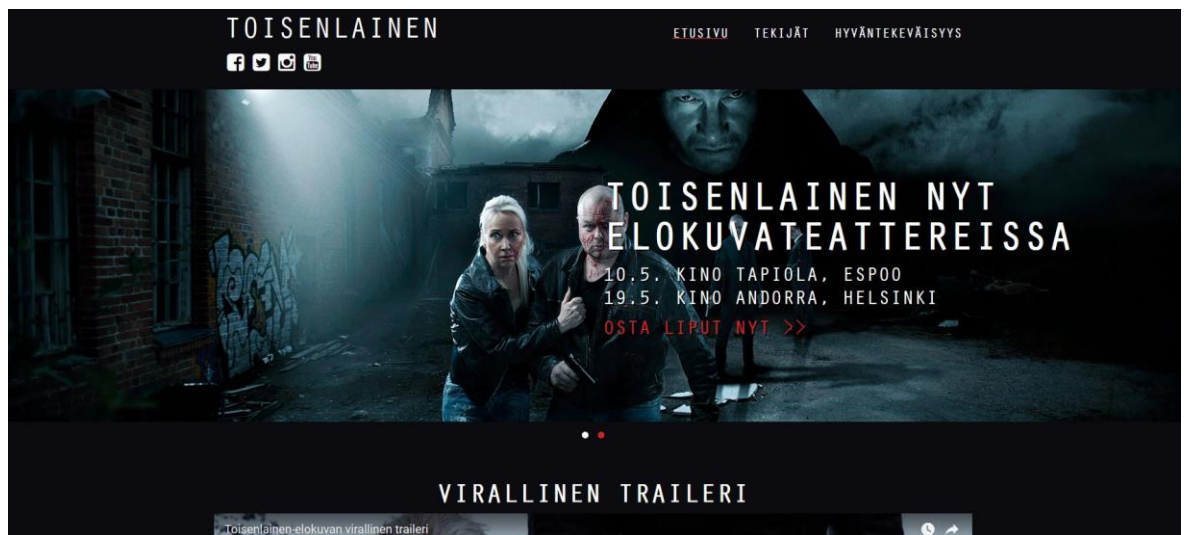
6.4.5 Javascript

Käytin sivustolla yhtä ulkoista Owl Carousel -nimistä javascript-lisäosaa etusivun karusellelementtiä varten. Karuselli kokoaa automaattisesti kaikki Nosto-sisältötyyppejä edustavat julkaisut. Sisältötyyppi on määritelty erikseen Custom Post Type -määrittelyllä.

Toimiakseen Owl Carousel lataa tiivistetyn owl.carousel.min.js -skriptitiedoston sivun footerissa. Lisäksi elementti vaatii määrittelyskriptin, jonka sijoitin erilliseen scripts.min.js tiedostoon, joka latautuu myös sivuston footerissa. Määrittelyskripti näyttää tältä:

```
$(document).ready(function() {  
    $('#carousel #liftups').owlCarousel({  
        items:            1,  
        loop:              true,  
        autoplay:         true,  
        autoplayTimeout:  2000,  
        smartSpeed:       1700,  
        autoplayHoverPause: true,  
    });  
});
```

Määrittelyssä kerrotaan, mitkä elementit halutaan sitoa karuselliin. Olen sitonut aiemmin tekemäni nostot käyttämään liftups id:tä, jota käytän tässä määrittelyssä. Koodissa säädetään myös karusellin asetuksia eli tässä tapauksessa se esittää yhden elementin (noston) kerrallaan ja vaihtaa elementtejä automaattisesti (autoplay). Automaattinen esitys pysähtyy hiiren hoverilla ja esitykselle on asetettu nopeus. Kuvassa 7 näkyy miten karuselli renderöityy sivulle.



Kuva 7. Etusivun karuselli (Kuvakaappaus: toisenlainenelokuva.com. 2016)

Owl Carouselin ohella käytin kustomoitua javascriptiä, jonka kirjoitin jQuery-muodossa scripts.min.js-tiedostoon. Sijoitin tiedostoon lausekkeet, joilla mobiilinavigaatio aukeaa menu-ikonin painamalla. Käytin avautumiseen toggle-funktiota. Määritin tiedostoon myös, että sivuston alaosassa olevaa Alkuun-linkkiä painamalla näkymä kelautuu takaisin sivun yläosaan.

Näyttelijät-osion mobiiliversio tuotti minulle eniten päänvaivaa. Tarkoitus oli, että mobiilikossa näyttelijöistä esitetään vain nimet ja tarkempi kuvaus kuvineen avautuu nimeä ja nuoli-ikonin klikkaamalla. Tein mobiilinäkymän määrittelyt myös scripts.min.js-tiedostoon.

Aluksi määritin, että koko ehdollisuus on voimassa vain tablettikokoisilla ja sitä pienemmillä ruuduilla (< 767). Tämän jälkeen määritin, että piilotettu esittely (actor-info) ja kuva aukeaisivat näyttelijän nimeä ja nuoli-ikonin klikattaessa (actor). Erityisen hankalaa koodilogiikassa oli saada vain klikatun näyttelijän tiedot avautumaan ja sulkeutumaan loogisesti. Päädyin lopulta käyttämään oheista koodirakennetta:

```
if ($(window).width() < 767) {
    $('.actor-slide').removeClass("clicked");
    $('.actor').click(function(){
        $( this ).children('.actor-info').slideToggle('fast');
        $( this ).children('img').css('display', 'block');
        $( this ).children('img').css('height', 'auto');

        if ($( this ).find('.actor-slide').hasClass("clicked")) {
            $( this ).find('.actor-slide').removeClass("clicked");
            $( this ).children('img').css('display', 'none');
        }
        else{
            $( this ).find('.actor-slide').addClass("clicked");
        }
    });
}
```

6.4.6 Php

WordPress koostuu php:stä ja myös sen käyttämää teemaa ja sisältöä voi ohjalla php-komennoilla. Käytin php:tä määritellessäni, miten luomani Custom Post Type -julkaisut tulostetaan koodissa. Esimerkiksi Nosto-tyyppisten julkaisujen sisältö on haettu koodin käsiteltäväksi `get_field()`, `get_the_post_thumbnail()`, `the_title()` ja `the_content()` -komennoilla. Samaa tekniikkaa on käytetty myös esim. Näyttelijät-elementin kanssa.

Karusellissa esitettävien nostojen määrä vaihtelee tarpeen mukaan, joten minun piti määrittää ehdollisuus, joka tutkii onko nostoja yksi, useampi kuin yksi vai ei yhtään. Toteutin tämän ehdollisuuden php:llä `wp_count_posts()` -funktiolla. Koodin mukaan jos lift-ups -

tyyppisiä julkaisuja on yksi, se sidotaan carousel-single luokkaan ja esitetään paikallaan olevana elementtinä ilman karusellin pyörimisominaisuuksia. Jos lift-ups -tyyppisiä julkaisuja on useampia, julkaisut käyttävät carousel-luokkaa, joka luo pyörivän karusellimaisen toiminnallisuuden. Nostot hakeva koodi ehdollisuuksineen on esitetty kuvassa 8.

```

3 <?php
4     $count_posts = wp_count_posts('liftup')->publish;
5     if($count_posts == 1) {?>
6         <div class="carousel-single">
7             <div id="liftups" class="owl-carousel">
8                 <?php
9                     $args = array( 'post_type' => 'liftup', 'posts_per_page' => 10 );
10                    $loop = new WP_Query( $args );
11                    while ( $loop->have_posts() ) : $loop->the_post();?>
12                        <div class="liftup">
13                            <a href="<?php echo get_field( 'target_url' );?>">
14                                <?php echo get_the_post_thumbnail();?>
15                                <div class="container">
16                                    <div class="lift-up-wrapper col-md-6 col-sm-12">
17                                        <h2><?php the_title();?></h2>
18                                        <?php the_content();?>
19                                    </div>
20                                </div>
21                            </a>
22                        </div>
23                    <?php
24                        endwhile;
25                    ?>
26                </div>
27            </div>
28        </div>
29
30    <?php } else if ( $count_posts > 1 ) { ?>
31        <div class="carousel">
32            <div id="liftups" class="owl-carousel">
33                <?php
34                    $args = array( 'post_type' => 'liftup', 'posts_per_page' => 10 );
35                    $loop = new WP_Query( $args );
36                    while ( $loop->have_posts() ) : $loop->the_post();?>
37                        <div class="liftup">
38                            <a href="<?php echo get_field( 'target_url' );?>">
39                                <?php echo get_the_post_thumbnail();?>
40                                <div class="container">
41                                    <div class="lift-up-wrapper col-md-6 col-sm-12">
42                                        <h2><?php the_title();?></h2>
43                                        <?php the_content();?>
44                                    </div>
45                                </div>
46                            </a>
47                        </div>
48                    <?php
49                        endwhile;
50                    ?>
51                </div>
52            </div>
53        </div>
54    <?php }?>

```

Kuva 8. Nosto-elementtien ehdollinen käsittely koodissa. (Kuvakaappaus projektin koodista.)

6.4.7 Less

Less on tyylikieli CSS:stä kehitetty laajennettu versio, joka on rakennettu Javascriptillä ja perustuu avoimeen lähdekoodiin (Wikipedia 2016.). Tavallisesta CSS:stä poiketen se mahdollistaa muuttujien, operaattoreiden ja funktioiden käytön sekä niiden yhdistelyn ja hierarkisen jäsentelyn. Erityisesti sisäkkäisen hierarkian käyttö helpottaa tyylikoodin lukua ja tyylimuuttujat puolestaan nopeuttavat muutosten tekoa. Esimerkiksi muuttaessani punaista huomioväriä, riitti, että muutin määriteltyä värimuuttujan arvoa yhdessä paikassa. Päivitys tuli voimaan kaikkialla, missä kyseinen värimuuttuja oli käytössä.

Internet-selain ei lue Lessiä suoraan, vaan se täytyy kääntää joko palvelimelle asennettavalla kääntäjällä tai erillisellä ohjelmalla paikallisesti. Tässä projektissa työskentelin Lessillä vain omassa paikallisessa kehitysympäristössäni. Käytin Koala-kääntäjää, joka loi Less-tiedostoista tiivistetyn CSS-tiedoston, jonka sitten siirsin palvelimelle. Näin tyylimuutoksia oli helppo tehdä ja testata paikallisesti, eikä palvelimelle kertynyt turhia tiedostoja vaan ainoastaan yksi tiivistetty CSS.

Bootstrap ei huomioi responsiivisessa jaottelussaan isoja ja pieniä puhelimia erikseen eikä juuri huomioi vaakaan käännettä tablettia, jonka näyttö on melko iso. Näiden puutteiden takia päädyin käyttämään media query -määrittelyjä Less-tiedostoissani optimoidakseni tyylit myös käännetylle tabletille (koko 992px > 720px), isommalle puhelimelle tai pienelle tabletille (koko 719px > 500px) sekä pienelle älypuhelimelle kuten iPhone 4:lle (koko < 499px). Pääosan responsiivisista määreistä tein siis ao. jaottelun mukaan:

```
@large-desktop: ~"only screen and (min-width: 1200px) and (max-width: 3199px)";
@desktop: ~"only screen and (min-width: 992px) and (max-width: 1199px)";
@tablet: ~"only screen and (min-width: 720px) and (max-width: 991px)";
@phone: ~"only screen and (min-width: 500px) and (max-width: 719px)";
@small-phone: ~"only screen and (max-width: 499px)";
```

6.5 Yhteydenpito toimeksiantajan kanssa

Työstövaiheen aikana tapasin toimeksiantajan kanssa suunnilleen kerran viikossa. Pidimme palaverit melko lyhyinä ja rentoina, mutta niiden ansiosta pysyimme hyvin selvillä projektin etenemisestä. Esittelin tapaamisissa edellisen tapaamisemme jälkeen tekemäni muutokset, jonka jälkeen neuvottelimme mitkä asiat otettaisiin seuraavaksi työnalle.

Tapaamiset olivat minulle hyvä tilaisuus kysyä tarkennuksia sivustolle tulevasta sisällöstä ja tiedustella tilannetta tarvittavien materiaalien suhteen. Esimerkiksi Näyttelijät-elementin

rakenne määräytyi tarkemmin vasta, kun olin keskustellut toimeksiantajan kanssa tiedoista, jotka hän halusi osiossa esittää. Keskustelun pohjalta osasin varata asetteluun tarpeeksi tilaa tiedoille sekä kuvalle.

Tapaamisten lisäksi sekä minä että toimeksiantaja ilmoitimme aina sähköpostilla isomista asioista, kuten domain-nimen hankinnasta tai että sivusto oli siirretty palvelimelle huoltotilaan. Mielestäni työskentely toimi sujuvasti koko projektin ajan. Omia haasteitaan aiheutti tietysti toimeksiantajan muiden töiden aiheuttama kiireinen aikataulu, mutta lopulta löysimme aina aikaa tapaamisille tarpeen mukaan.

6.6 Julkaisu

Sivuston julkaisupäivämääräksi oli suunniteltu elokuvan ensi-iltapäivä 14.4.2016. Maaliskuun lopulla sivusto alkoi kuitenkin olla rakenteellisesti jo niin hyvässä vaiheessa, että kun toimeksiantaja ehdotti julkaisun aikaistamista, pyyntö oli toteutettavissa. Julkaisu-aikataulun muutoksen syy oli elokuvan markkinoinnissa. Ensi-illasta haluttiin tiedottaa lehdistölle ja yhteistyökumppaneille jo paria viikkoa etukäteen, joten oli loogista, että myös elokuvan kotisivut olisivat silloin esittelykunnossa.

Toimeksiantaja oli jo aiemmin hankkinut sivustolle domainin sekä webhotellista palvelintilan. Käytin WinSCP FTP-ohjelmaa, jolla siirsin kooditiedostot suoraan palvelimella olevaan kansioon. Sivuston rakenteet ja tyylit olivat vielä kesken ja sisällöt puuttuivat, joten asensin sivustolle WordPress -lisäosan nimeltä Maintenance Mode. Lisäosan aktivoimalla sivulle saa käyttöön ns. huoltotilan, jolloin sivun todellinen sisältö näkyy vain kirjautuneille käyttäjille eli admineille. Ulkopuolinen käyttäjä näkee vain määritellyn huoltoilmoitussivun. Huoltotilassa minulla ja toimeksiantajalla oli työrauha rakentaa sivustoa eikä käyttäjien nähtäväksi päätynyt liian keskeneräistä tietoa.

Ennen julkaisua testasin sivustoa eri päätelaitteilla. Selaimissa on kyllä kehittäjän työkaluja, joilla voi mallintaa sivustojen responsiivisuutta, mutta toimivuuden ja käyttökokemuksen kannalta on järkevää testata sivustoa mahdollisuuksien mukaan myös oikeilla kohdelaitteilla. Onneksi minulta sekä toimeksiantajalta löytyi kattava valikoima niin Android kuin IOS -pohjaisia laitteita, joilla selailimme sivustoa läpi ja tarkistimme, että kaikki näkymät toimivat oikein.

Päätelaitetestauksen lisäksi ajoin sivuston läpi erilaisista nopeustesteistä. Käytin testeissä Googlen, GTMetrixin sekä Pingdomin nopeustestejä. Testityökaluihin syötetään sivuston osoite ja työkalu testaa kuinka nopeasti sivusto vastaa kutsuihin ja latautuu. Testien pe-

rusteella työkalu ilmoittaa luvullisen arvosanan ja kertoo korjausehdotuksia, joilla arvostaa voisi parantaa. Esimerkiksi Googlen testityökalu antaa myös erilliset arvostamat työpöytä- sekä mobiilitesteille. Mielestäni testeissä oli hyvä käyttää useita eri testityökaluja, sillä osa työkaluista antoi hieman toisistaan poikkeavia tuloksia. Tulosten keskiarvoa tutkimalla sain luotettavamman käsityksen sivuston toimivuudesta.

Testausten ja rakenteellisten viilausten lisäksi julkaisuvaihe oli pääasiassa sisällönsyöttämisestä ja hiomista. Kun kaikki oli kunnossa, poistimme huoltotilan käytöstä ja aloitimme sivuston markkinoinnin sosiaalisessa mediassa.

6.7 Julkaisun jälkeen

Alusta asti oli sovittu, että sivuston päivitys ja ylläpito jatkuu julkaisupäivän jälkeenkin. WordPressiin ja sen lisäosiin tulee aika-ajoin päivityksiä, joiden asentaminen on suositeltavaa. Sivustolle asentamani WordFence-lisäosa ilmoittaa sähköpostitse aina, kun uusia päivityksiä on saatavilla, mikä onkin ollut erittäin kätevää, ettei sivustoa tarvitse käydä vahtimassa päivittäin.

6.7.1 WordPressin suojausasetukset

Asennuksen yhteydessä lisäsin sivustolle WordFence -lisäosan, joka skannaa sivuston liikennettä ja ilmoittaa epäilyttävistä muutoksista tai epäonnistuneista sisäänkirjautumisyrittämisistä. Jonkin aikaa sivuston julkaisemisen jälkeen sain sähköpostiini useita ilmoituksia käyttäjistä, jotka olivat yrittäneet kirjautua sivustolle liian monta kertaa epäonnistuneesti ja heidät oli estetty. Kyseessä oli palvelunestohyökkäys, eli haittaohjelmien saastuttamat tietokoneet yrittivät murtaa sivuston salasanaa ja tunkeutua sen tiedostoihin. WordPressin omien sivujen sekä useiden muiden internet-lähteiden mukaan hyökkäykset kohdistuvat erityisesti sivustoille, joissa on vanhentunut WordPress-versio ja päivittämättömiä lisäosia. Paras tapa suojautua hyökkäyksiltä onkin pitää sivusto ajantasalla ja näyttää, että sitä ylläpidetään. (Wordpress 2016.) Päivitin WordPressin sekä käyttämäni lisäosat uusimpaan versioon.

Hyökkäykset loppuivat, mutta sain jälleen WordFencen ilmoituksen, jonka mukaan WordPressin sisäisiä (core-) tiedostoja oli muokattu. Muokkaukset eivät olleet vaikuttaneet sivuston toimintaan, mutta joku oli päässyt käsiksi wp-includes-kansion tiedostoihin. Päivitin palvelimelle .htaccess ja wp-config.php -tiedostoja lukuunottamatta koko WordPress-

asennukseni, jotta sain hävitettyä muokatut tiedostot. Lisäksi luin lisää WordPressin suojasasetuksista sen omilta sivuilta.

Tiedostoja voi suojata .htaccess-tiedoston avulla. Muutokset wp-config.php-tiedostoon voi estää lisäämällä estokoodin .htaccess-tiedoston yläosaan (Wordpress 2016.):

```
<files wp-config.php>
order allow,deny
deny from all
</files>
```

Lisäksi wp-includes -kansio on mahdollista suojata oheisella koodilla, joka lisää myös .htaccess-tiedostoon. (Wordpress 2016.)

```
# Block the include-only files.
<IfModule mod_rewrite.c>
RewriteEngine On
RewriteBase /
RewriteRule ^wp-admin/includes/ - [F,L]
RewriteRule !^wp-includes/ - [S=3]
RewriteRule ^wp-includes/js/tinymce/langs/.\.php - [F,L]
RewriteRule ^wp-includes/theme-compat/ - [F,L]
</IfModule>
```

Siltä varalta, että hakkerit pääsisivät kirjautumaan WordPressin hallintapaneeliin, tein muutoksen myös wp-config.php-tiedostoon. Muutos estää etteivät admin-käyttäjät voi muokata tiedostoja hallintapaneelin kautta. (Wordpress 2016.) Estokoodi näyttää tältä:

```
define('DISALLOW_FILE_EDIT', true);
```

6.7.2 Latautumisen optimointi

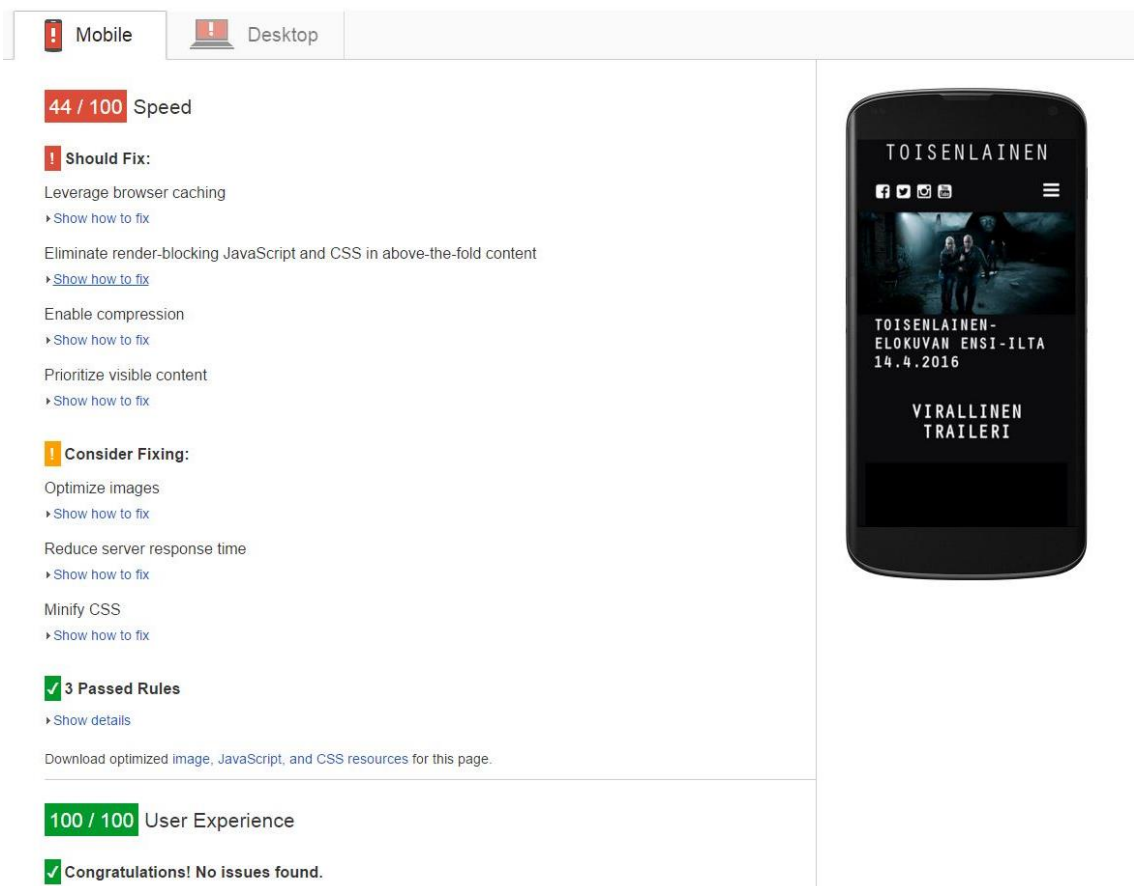
Ylläpidon ja seurannan lisäksi sivustoa voi toki aina kehittää. Mielestäni sivuston latautumisenopeudessa oli parantamisen varaa, joten tein muutamia korjauksia siihen liittyen.

Googlen latautumisenopeutta testaava työkalu PageSpeed oli antanut sivuston julkaisun aikaan mobiililaitteen suorittaman latautumisen arvosanaksi 41/100 ja pöytäkoneen suorittaman latautumisen arvosanaksi 52/100. Sivuston etusivulla on paljon kuvia, joita työkalu suositteli pienentämään. Olin jo aiemmin optimoinut kuvia, mutta ajoin ne vielä PngGauntlet ja Radical Image Optimization Tool (RIOT) -sovellusten läpi. PngGauntlet käsittelee png-kuvia, RIOT puolestaan sopii jpg-kuville. Sovellukset eivät heikennä kuvien laatua,

mutta pakkaavat kuvia pienentäen niiden tiedostokokoa. Kompressoitujen kuvien vaihtaminen sivustolle paransi testituloksia jo hieman.

Kuvien lisäksi optimoin css- ja javascript-tiedostoja. Siirsin kaikki mahdolliset tiedostot latautumaan vasta sivuston footer-osiossa. Sen sijaan, että lataisin kaikki javascript- ja css-tiedostot yksitellen siirsin tiedostoja niin, että mahdollisimman moni niistä latautuisi keskitetysti kompressoitujen scripts.min.js- ja style.min.css-tiedostojen kautta. Näillä keinoilla sain karsittua muutaman turhan kyselyn palvelimelta. Kaikkia tiedostoja en kuitenkaan voinut optimoida näin, sillä esimerkiksi Bootstrapin tyylitiedoston siirtäminen keskitettyyn css-tiedostoon aiheutti koko ulkoasun hajoamisen.

Pääasia oli kuitenkin, että sivuston latautumisenopeus parani tekemilläni muutoksilla. Muutosten jälkeen mobiililatautumisen arvosana oli noussut 41:stä 44:ään ja pöytäkoneen latautuminen 52:sta 57:ään. Kuvista 9 ja 10 näkyy arvosanojen muutos.



Mobile Desktop

44 / 100 Speed

! **Should Fix:**

- Leverage browser caching
→ [Show how to fix](#)
- Eliminate render-blocking JavaScript and CSS in above-the-fold content
→ [Show how to fix](#)
- Enable compression
→ [Show how to fix](#)
- Prioritize visible content
→ [Show how to fix](#)

! **Consider Fixing:**

- Optimize images
→ [Show how to fix](#)
- Reduce server response time
→ [Show how to fix](#)
- Minify CSS
→ [Show how to fix](#)

✓ **3 Passed Rules**
→ [Show details](#)

[Download optimized image, JavaScript, and CSS resources for this page.](#)

100 / 100 User Experience

✓ **Congratulations! No issues found.**

Kuva 9. Google PageSpeed -testin tulokset mobiililatautumisesta muutosten jälkeen. (Kuvakaappaus Google Developers PageSpeed Insight-sivustolta.)

Mobile Desktop

57 / 100 Suggestions Summary

! Should Fix:

- Enable compression
 - › Show how to fix
- Leverage browser caching
 - › Show how to fix
- Optimize images
 - › Show how to fix
- Eliminate render-blocking JavaScript and CSS in above-the-fold content
 - › Show how to fix


! Consider Fixing:

- Prioritize visible content
 - › Show how to fix
- Reduce server response time
 - › Show how to fix
- Minify CSS
 - › Show how to fix

✓ 3 Passed Rules

- › Show details

Download optimized image, JavaScript, and CSS resources for this page.



Kuva 10. Google PageSpeed -testin tulokset pöytäkoneen latautumisesta muutosten jälkeen. (Kuvakaappaus Google Developers PageSpeed Insight-sivustolta.)

7 Pohdinta

Opinnäytetyöni tavoite oli toteuttaa responsiivinen sivusto Toisenlainen-elokuvalle, oppia uusia tekniikoita ja harjoitella responsiivista sivustokehitystä prosessina. Pääsinkin käyttämään työssäni monipuolisia tekniikoita, mikä oli mielestäni erittäin opettavaista. Erityisen mielenkiintoista oli kokonaisuuden hallinta aina sivuston suunnittelusta ja ulkoasun mallinnuksesta lopulliseen toteutukseen sekä ylläpitoon. Yllätyksiä ja uutta opittavaa tuli vielä sivuston julkaisun jälkeenkin.

Näin jälkepäin projektin haasteena oli aikataulun looginen hallinta. Kun olin aloittanut sivuston suunnittelun yhdessä toimeksiantajan kanssa, keskityin kokonaan sivuston rakentamiseen ja kehitykseen, enkä juurikaan siinä vaiheessa etsinyt uutta tietoa projektin tueksi. Minulla oli alusta asti mielikuva, miten aion toteuttaa sivuston, joten puskin projektia eteenpäin ja uppouduin koodauksen pariin. Samalla lykkäsin kirjojen lukemista ja teorian etsimistä myöhemmäksi. Pelkäsin, että sivuston kanssa saattaisi tulla teknisiä ongelmia ja siksi halusin pysyä projektin suhteen mahdollisimman hyvin aikataulussa. Teknisiä vaikeuksia ei lopulta tullut ja sivusto valmistuikin etuajassa.

Julkaisun jälkeen minulla oli enemmän aikaa tutkia responsiivista kehitystä ja perehtyä suositeltuihin käytäntöihin. Vaikka responsiivinen kehitys oli minulle melko tuttua entuudestaan, löysin tutkimusvaiheessa uutta tietoa ja vinkkejä aiheesta. Jos olisin perehtynyt teoriaan enemmän etukäteen, olisin todennäköisesti tehnyt jotkin asiat sivuston kehityksessä toisin ja selkeämmin. Joitain muutoksia, kuten tyyli- ja skriptikoodien yhdistelyä sekä kuvien kompressointia teinkin sivuston julkaisemisen jälkeen, kun olin lukenut tarkemmin optimointikäytännöistä. Mielestäni sivustojen kehityksen ei ylipäätäänkään pitäisi päättyä julkaisupäivään, joten siinä mielessä korjaukset olikin parempi tehdä hieman myöhemmin kuin jättää kokonaan tekemättä.

Teoriataustaa lukiessani minusta oli positiivista huomata, että olin osannut toteuttaa suurimman osan ominaisuuksista suositellulla tavalla ja prosessini oli noudattanut kirjojen ohjeita. Esimerkiksi, kuten ensimmäisessä luvussa kerroin, responsiivinen suunnittelu vaatii, että ensin suunnitellaan sivuston sisältö ja sitten vasta rakenne. Olin noudattanut tätä mallia toimeksiantajani kanssa, sillä ensimmäisessä kokouksessa kävimme tarkasti läpi juurikin sivustolle tulevaa sisältöä ja vasta sen jälkeen suunnittelin ehdotuksen rakenteesta.

Myös tapani työstää ulkoasua suoraan koodiin mukaili hyvin Carverin suosittamaa tapaa, jossa tiukat ulkoasumallit hylätään ja koodaaja sekä graafikko työskentelevät tiiviisti yhdessä. (Carver 2015, 48 ja 78)

Tutkimuksessani huomasin myös, että responsiivisuus ei haasta tekijäänsä vain teknisesti vaan myös ajatuksellisesti. Teoriatausta vahvasti oletukseni, että responsiivisuuden hahmottaminen tuottaa vaikeuksia etenkin vanhemman koulukunnan koodareille sekä ulkoasusuunnittelijoille. Uuden tekniikan hahmottaminen vaatii kokonaan uudenlaisen ajatusmallin sisäistämistä sekä syventymistä erilaisiin työskentelytapoihin.

Sekä minä että toimeksiantajani olemme saaneet sivustosta positiivista palautetta ja se on tukenut hyvin elokuvan tämän hetkistä markkinointia yhdessä sosiaalisen median kanssa. Vaikka nyt jälkepäin uskon, että sivuston koodirakenteen olisi voinut rakentaa vielä selkeämmäksi ja loogisemmaksi, mielestäni sain kuitenkin lyhyessä ajassa tehtyä siitä toimivan kokonaisuuden, joka täyttää sille asetetut tavoitteet ja on uskottava kilpakumppaneihin verrattuna.

Minulle sivuston tekeminen opinnäytetyönä oli juuri sellainen projekti, jota kaipasin. Itselle läheiseen asiaan oli helppo tarttua ja panostaa laatuun. Kehittymisen kannalta verkkosivusto tarjosi loistavan pelikentän harjoitella ja kokeilla erilaisia teknisiä ratkaisuja. Ajatukseni oli myös, että valmiista sivustosta saisin mielenkiintoisen työnäytteen sekä totta kai arvokasta kokemusta tulevia projekteja varten. Tässä mielessä koen onnistuneeni.

Lähteet

BuiltWith.com 2016. CMS Usage Statistics. Luettavissa: <http://trends.builtwith.com/cms>.
Luettu: 14.03.2016

Carver, M. 2015. The Responsive Web. Manning Publications Co. USA

Damstra, D., Stern, H. & Williams B. 2010. Professional WordPress: Design and Development. Wiley Publishing. USA

Dutson, P. 2015. Responsive Mobile Design. Designing for Every Device. Pearson Education. USA

Getbootstrap.com 2016. Bootstrap. Luettavissa: <http://getbootstrap.com/>. Luettu: 21.3.2016

Google Developers 2016. Which web browsers natively support WebP Luettavissa: https://developers.google.com/speed/webp/faq#which_web_browsers_natively_support_webp. Luettu: 26.4.2016

Google Developers 2016. PageSpeed Insight. Luettavissa: <https://developers.google.com/speed/pagespeed/insights/>. Luettu: 22.5.2016

Krug, S. 2014. Don't Make Me Think, Revisited - A Common Sense Approach to Web Usability. New Riders. USA

Lesscss.org 2016. Getting started. Luettavissa: <http://lesscss.org/>. Luettu 21.3.2016

Ordinarycoder.com 2016. WordPress blank bootstrap theme (lightweight). Luettavissa: <http://www.ordinarycoder.com/wp-speed-code-blank-theme/>. Luettu: 21.3.2016

Peltomäki, V. 2014. Bootstrap Framework web-suunnittelun työkaluna 2014. Luettavissa: http://www.theseus.fi/bitstream/handle/10024/74557/Peltomaki_Veera.pdf?sequence=1.
Luettu: 21.3.2016

Startbootstrap.com 2016. Start Bootstrap. Luettavissa: <http://startbootstrap.com/>.
Luettu: 21.3.2016

WampServer 2016. Start with WampServer. Luettavissa: <http://www.wampserver.com/en/>.
Luettu: 17.5.2016

Toisenlainenelokuva.com. 2016. Luettavissa: <http://toisenlainenelokuva.com/>.
Luettu: 22.5.2016

Wikipedia.org 2016. Bootstrap (front-end framework). Luettavissa: https://en.wikipedia.org/wiki/Bootstrap_%28front-end_framework%29. Luettu: 21.3.2016

Wikipedia.org 2016. LAMP. Luettavissa: <https://fi.wikipedia.org/wiki/LAMP>.
Luettu: 30.3.2016

Wikipedia.org 2016. LAMP (software bundle). Luettavissa: https://en.wikipedia.org/wiki/LAMP_%28software_bundle%29. Luettu: 30.3.2016

Wikipedia.org 2016. Less (stylesheet language) Luettavissa: https://en.wikipedia.org/wiki/Less_%28stylesheet_language%29. Luettu: 21.3.2016

Wordpress.org 2016. About WordPress. Luettavissa: <https://wordpress.org/about/>.
Luettu: 14.03.2016.

Wordpress.org 2016. Hardening WordPress. Luettavissa: http://codex.wordpress.org/Hardening_WordPress. Luettu: 27.4.2016

Wordpress.org 2016. Post Types. Luettavissa https://codex.wordpress.org/Post_Types.
Luettu: 29.4.2016

Liitteet

Liite 1. Suunnittelukokouksen muistilista

Tarpeet	tarkennukset	Nice to have	Referenssit
Nostokaruselli		making of -materiaalia	http://www.thehungergames.movie/#/mj2?lang=fi-en
navigaatio		haastatteluja	http://www.everest-movie.co.uk/
Some-kanavat			Varoittava esimerkki! http://www.helsinkifilmi.fi/elokuvat/napapiirinsankarit/
Some-päivitykset			http://www.solar-films.com/elokuvat/kaikki/luokkakokous/fi_FI/synopsis/
Visuaalinen ilme	tarvitaan julistemateriaalit graafikolta		
tapahtumat			
uutiset			
ota yhteyttä			
<i>tekijät</i>	<i>mitä tietoja halutaan kertoa</i>		
trailerit/teaserit			
<i>näyttelijäesittelyt</i>	<i>mitä tietoja halutaan kertoa</i>		
<i>sponsorit!</i>			
kuvia			
rahoita			
<i>hahmot</i>	<i>mitä tietoja halutaan kertoa</i>		
elokuvan esittely			

Liite 2. Pelkistetty rautalankamalli

Toisenlainen-elokuva
[somelogot]

Etusivu Tekijät Rahoita Ota yhteyttä

ENSI-ILTA
14.4.2016

Traileri

Synopsis

Mauris vitae cursus odio, sed venenatis massa. Donec rhoncus lacus ut accumsan accumsan. Nullam tempor pellentesque sodales. Suspendisse potenti. Quisque vestibulum, sapien eget molestie rhoncus, ex metus convallis nisi, vel placerat nisl quam eu tellus. Aliquam rhoncus magna at ligula rhoncus, sed placerat diam ullamcorper.

Näyttelijät

Johanna Kare Mari	Johanna Kare Mari Pääosa näyttelijä. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent non sapien odio. Vestibulum massimus fringilla nulla, id dignissim nunc ullamcorper vitae. Nunc malesuada laoreet a augue a blandit. Praesent laculis, felis eget ornare laoreet, orci lectus pulvinar.	Johanna Kare Mari	Johanna Kare Mari
Johanna Kare Mari	Johanna Kare Mari	Johanna Kare Mari	Johanna Kare Mari

Tekijät

Toisenlainen toteutettiin vapaaehtoisvoimin. Mauris vitae cursus odio, sed venenatis massa. Donec rhoncus lacus ut accumsan accumsan. Nullam tempor pellentesque sodales. Suspendisse potenti. Quisque vestibulum, sapien eget molestie rhoncus, ex metus convallis nisi, vel placerat nisl quam eu tellus.

[TUTUSTU TEKJÖIHIN](#)

Kiitämme yhteistyöstä

alepa

MILLIKLUBI
BAR AND DISCO

ym.

Liite 3. Kuvitettu rautalankamalli työpöytäversiosta

Toisenlainen-elokuva
[somegot]

Etusivu Tekijät Rahoita Ota yhteyttä

ENSI-ILTA 14.4.2016









Traileri

Trailer video player showing a scene from the movie.

Synopsis

Mauris vitae cursus odio, sed venenatis massa. Donec rhoncus lacus ut accumsan accumsan. Nullam tempor pellentesque sodales. Suspendisse potenti. Quisque vestibulum, sapien eget molestie rhoncus, ex metus convallis nisi, vel placerat nisl quam eu tellus. Aliquam rhoncus magna at ligula rhoncus, sed placerat diam ullamcorper.

Näyttelijät

 <p>Johanna Kare Mari</p>	 <p>Johanna Kare Mari</p>	 <p>Johanna Kare Mari</p>	 <p>Johanna Kare Mari</p>
 <p>Johanna Kare Mari</p>	 <p>Johanna Kare Mari</p>	 <p>Johanna Kare Mari</p>	 <p>Johanna Kare Mari</p>

Tekijät

Toisenlainen toteutettiin vapaaehtoisvoimin. Mauris vitae cursus odio, sed venenatis massa. Donec rhoncus lacus ut accumsan accumsan. Nullam tempor pellentesque sodales. Suspendisse potenti. Quisque vestibulum, sapien eget molestie rhoncus, ex metus convallis nisi, vel placerat nisl quam eu tellus.

[TUTUSTU TEKIJÖIHIN](#)


Kiitämme yhteistyöstä

alepa **MILLIKLUBI**
BAR AND DISCO

ym.

Liite 4. Kuvitettu rautalankamalli mobiiliversiosta


Toisenlainen- elokuva



< **ENSI-ILTA** >
1.3.2016

• ○ •

Synopsis



Mauris vitae cursus odio, sed venenatis massa. Donec rhoncus lacus ut accumsan accumsan. Nullam tempor pellentesque sodales. Suspendisse potenti. Quisque vestibulum, sapien eget molestie rhoncus, ex metus convallis nisi, vel placerat nisi quam eu tellus.

LUE LISÄÄ

Tekijät



Johanna Kare

Pääosa näyttelijä. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Praesent non sapien odio. Vestibulum maximus fringilla nulla, id dignissim nunc ullamcorper vitae. Nunc malesuada laoreet augue a blandit. Praesent iaculis, felis eget ornare laoreet, orci lectus pulvinar libero.



Tekijät

Mauris vitae cursus odio, sed venenatis massa. Donec rhoncus lacus ut accumsan accumsan. Nullam tempor pellentesque sodales. Suspendisse potenti. Quisque vestibulum, sapien eget molestie rhoncus, ex metus convallis nisi, vel placerat nisl quam eu tellus. Aliquam rhoncus magna at ligula rhoncus, sed placerat diam ullamcorper.

[LUE LISÄÄ](#)

Liite 5. Valmis sivusto työpöytä näkymässä

TOISENLAINEN

ETUSIVU TEKIJÄT HYVÄNTEKEVÄISYYS

TOISENLAINEN-ELOKUVAN ENSI-ILTA 14.4.2016

VIRALLINEN TRAILERI

Toisenlainen-elokuvan virallinen traileri

TARINA

Toisenlainen tarina elämästä. Toisenlainen tarina rakkaudesta. Synkän alamaailman kurimuksessa päähenkilöt rakastuvat ja löytävät elämälleen tarkoituksen... Jos säilyvät hengissä kohdatakse uuden tulevaisuuden.

NÄYTTÉLIJÄT



PETRI AARNIO
LEO

JOHANNA KARE
MARI

Johanna on valokuvausta harrastava vahva ja intohimoinen monilahjakkuus, ehdottomasti parhaimmillaan päästäessään tunteensa valloilleen.

Elokuvassa hän on Mari, elämänmuutosta tekevä entinen narkkari. Matkallaan hän törmää palkkamurhaajaan, joka on hänen vastakohtansa, mutta Mari uskoo kaikissa ihmisissä hyvään.



PAULI AALTO-SETÄLÄ
POMO



NICKE LIGNELL
HAGMAN



PETER MUSTONEN
OZ



AAKE KALLIALA
EUROOPAN JOHTAJA



ARMAN ALIZAD
PAPAK



JUKKA HURME
ARMAN



MATTI JUSSILA
HENKIVARTIJA



FILIP MUHONEN
VLADIMIR



ANU HÄMÄLÄINEN
KATA



KAI PORTMAN
JOHN DOE



?
JOKERI



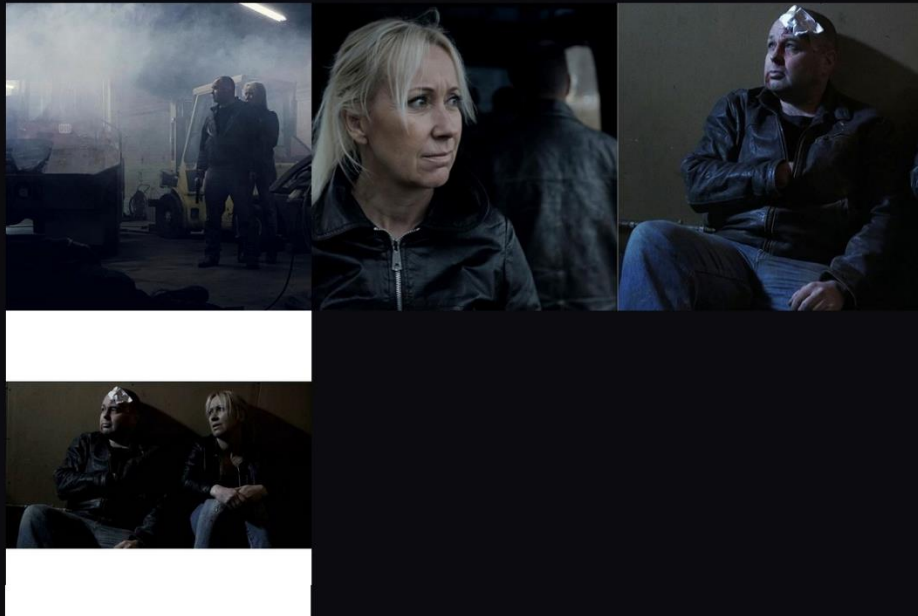
JOKERI

VAPAAEHTOISVOIMIN JA HYVÄLLÄ ASIALLA

Toisenlainen-elokuva on mahtava osoitus yhteisöllisyyden voimasta ja intohimosta elokuvaan. Ennakkoluulottomien sekä lahjakkaiden ihmisten ja yhteistyökumppaneiden avustuksella unelma ensi-illasta on toteutumassa.

Lue lisää

TOISENLAINEN INSTAGRAMISSA



KIITÄMME YHTEISTYÖSTÄ



SILJA LINE

KUUSAKOSKI RECYCLING

Hope YHDESSÄ & YHTEISESTI



alepa

A? Aalto-yliopisto

VITAL MED Helsingin Unikinikka

HELSINKI-MALMI AIRPORT EFHF75

H. Kuokkanen Peruuttiliike Oy

id HAIR

Everyday Beauty

So Stylish

PORT OF HELSINKI

Itäin pohjoinen optikkaliike OPTILOOK

MAILLIKLUBI Bar & Disco

HYVINKÄÄN KARTANO HOTELLI & RAHVINTOLA

MALLASAASHI

BeautyGrace

RÄNNIPARONI



RIITAN OLKKARI

BF-LENTO

Susanne Pitopalvelu

proairssoft & CO

ILMARI TEE PÄIVÄN TYÖT VARTISSA

MARINE MOTEL & RESTAURANT

lippu.fi

T2 TUOTANTO

INFO@T2TUOTANTO.COM

ALKUUN

SITE BY KATTI RASÄNEN